Research Article

Post-CTS Delay Insertion

Jianchao Lu and Baris Taskin

Department of Electrical and Computer Engineering, Drexel University, Philadelphia, PA 19104, USA

Correspondence should be addressed to Jianchao Lu, jl597@drexel.edu and Baris Taskin, taskin@coe.drexel.edu

Received 29 May 2009; Revised 23 October 2009; Accepted 18 November 2009

Academic Editor: Gregory D. Peterson

Copyright © 2010 J. Lu and B. Taskin. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

A post-clock-tree-synthesis (post-CTS) optimization method is proposed that suggests delay insertion at the leaves of the clock tree in order to implement a limited version of clock skew scheduling. Delay insertion is limited on each clock tree branch simultaneous with a global monitoring of the total amount of delay insertion. The delay insertion for nonzero clock skew operation is performed only at the clock sinks in order to preserve the structure and the optimizations implemented in the clock tree synthesis stage. The methodology is implemented as a linear programming model amenable to two design objectives: fixing timing violations or optimizing the clock period. Experimental results show that the clock networks of the largest ISCAS'89 circuits can be corrected post-CTS to resolve the timing conflicts in approximately 90% of the circuits with minimal delay insertion ($0.159 \times$ clock period per clock path on average). It is also shown that the majority of the clock period improvement achievable through unrestricted clock skew scheduling are obtained through very limited insertion ($\approx 43\%$ average improvement through 10% of max insertion).

1. Introduction

One of the tools at the designers' expense during the design of high performance ASIC circuits is the manipulation of clock delays to compensate for the timing critical paths at the physical design stage. After power and timing aware physical design steps of floorplanning, placement, and clock tree synthesis steps, timing verification can still reveal a number of violated paths, which might need an overall redesign of the system or iterative physical design steps to be resolved. Post-clock-tree-synthesis (post-CTS) optimization can be used to resolve such violated paths or to improve the clock period. The two objectives are considered in this paper.

In particular, a practical delay insertion process to be performed on a synthesized clock tree is introduced. This process is devised to work with industry standard automation tools, such that, the clock distribution network (i.e., clock tree) and, the placement results are the inputs to the proposed methodology. The timing verification tools are used to detect the violations on the data paths. These violations are eliminated (i.e., fixed) by inserting small delay elements on the clock branches. For circuits where timing is satisfied (no timing violations), delay mismatch can be used to implement a limited version of clock skew scheduling in order to improve the operating clock frequency [1]. A systematic study of the effectiveness of the delay

insertion method in both fixing timing violations and improving circuit frequency is presented. The formulation and mathematical analysis of the post-CTS delay insertion on clock leaves are presented that

- (i) preserves the structure of the zero clock skew tree,
- (ii) limits the amount of insertion on each clock branch,
- (iii) limits the amount of insertion on the overall clock tree.

Existing delay insertion methods, including [2], only limit the amount of delay insertion on clock branches. Such a limitation per branch is not optimal as there are often paths that do not require any delay insertion. The available space on the chip can be utilized more efficiently by permitting higher levels of delay insertion on each branch while simultaneously monitoring the total amount of delay insertion (such that the available space is not overused). Existing clock skew scheduling methods, including [2], are implemented with continuous delay models and do not limit the delay insertion, which are not practical. Consequently, practical implementation of clock skew scheduling resorts to suboptimal, iteration-based delay insertion procedures which are rarely methodical. The post-CTS delay insertion proposed in this paper constitutes a methodical and practical implementation of clock skew scheduling.

(a) Circuit structure of the clock distribution network

(b) Equivalent graph of the clock tree that corresponds to the circuit in (a)

FIGURE 1: Tree structure of a clock distribution network.

This paper is organized as follows. In Section 2, the timing constraints are reviewed and a brief description of the clock tree is introduced. In Section 3, the motivation of this paper is explained. In Section 4, the proposed post-CTS optimization methodology is demonstrated. In Section 5, experimental results on a suite of ISCAS'89 benchmark circuits are presented. The paper is finalized in Section 6.

2. Technical Background

The timing constraints of a synchronous local data path are used as a part of the proposed mathematical framework to perform post-CTS delay insertion. In Section 2.1, the clock network design process is outlined as in relevance to this work. In Section 2.2, these timing constraints of a synchronous local data path are reviewed.

2.1. Clock Network Design. Clock network design (also called clock tree synthesis) [3] is an essential step in the physical design flow of integrated circuits. During the clock network design step, the interconnect topology of the clock distribution network is designed based on the placement and routing information. The clock distribution network

FIGURE 2: Setup and hold constraints.

is frequently organized as a rooted tree structure [4, 5], as illustrated in Figure 1. A circuit schematic of a clock distribution network is shown in Figure 1(a). An abstract graphical representation of the tree structure is shown in Figure 1(b). The clock signal is distributed from the source to every register in the circuit through a sequence of *buffers* and *interconnect wires*. Such minimal or zero clock skew can be achieved by different routing strategies [6–9], buffered clock tree synthesis, symmetric n -ary trees [10] (most notably H-trees), using deskew buffers [11] or a distributed series of buffers connected as a mesh [12].

In this work, a generic tree implementation as shown in Figure 1 is considered. The proposed optimization methodology is performed *post-CTS*, thus, the synthesis of the clock tree and sizing of the buffers are considered complete. Consequently, any clock tree synthesis methodology or tool can be used for the clock tree synthesis process.

2.2. Static Timing Constraints. As shown in Figure 2, minimum and maximum propagation delays on the combinational path from register R_i to register R_f are denoted by D_{PMin}^{if} and D_{PMax}^{if} , respectively. The clock arriving time of a register R_i is denoted by t_i ; whereas the setup and hold times are denoted by S_i and H_i , respectively. The clock arriving time t_i represents the clock signal delay from the source to register R_i at that branch. The clock period is denoted by T . The clock to output delay of each register is D_{CQ}^i . The timing analysis of a synchronous circuit is performed by satisfying the *setup* timing constraints for each local data path:

$$\text{Setup: } t_i + D_{CQ}^i + D_{PMax}^{if} \leq t_f + T - S_f, \quad (1)$$

$$\text{Hold: } t_i + D_{CQ}^i + D_{PMin}^{if} \geq t_f + H_f. \quad (2)$$

For zero clock skew systems, clock delays t_i and t_f are identical

$$t_i = t_f \implies t_i - t_f = 0. \quad (3)$$

This equality of clock delays to registers simplifies the timing constraints. Further assuming that the internal register delays can be neglected ($D_{CQ} = S = H = 0$), a limitation on the clock period T is derived from (1)

$$\text{Setup: } D_{PMax}^{if} \leq T. \quad (4)$$

The setup constraint must be satisfied on all timing paths, leading to the following inequality:

$$\max_{\forall(i,j)} (D_{PMax}^{ij}) = T_{zs} \leq T. \quad (5)$$

FIGURE 3: Post-CTS delay insertion examples on a sample binary clock tree with N sinks.

Thus, if the circuit operates at any clock period less than the largest maximum data propagation time, a *timing violation* occurs [13]. Finding a clock period T for a zero clock skew circuit is always possible, making it convenient to design zero clock skew systems. Consequently, the application of zero clock skew schemes has been central to the design of fully synchronous digital circuits for decades [4]. The minimum clock period at zero skew T_{zs} is defined at the equality condition for inequality (5) and is used in the formulations as the basis metric to measure the improvement through clock skew scheduling.

3. Motivation

The proposed methodology of delay insertion at the leaves of the clock tree is a limited version of *clock skew scheduling*. Clock skew scheduling permits the modification of the clock delays to be different from each other, leading to a nonzero clock skew system

$$t_i \neq t_f. \quad (6)$$

The clock arrival time t_i might be less or greater than t_f , causing more time to the path between R_i and R_f , or the paths leading to R_i . The advantages of clock skew scheduling are well known and documented extensively in the literature [1]. The minimum clock period of a circuit with zero clock skew is the largest logic path delay in that circuit (5) and with clock skew scheduling, the minimum clock period can be improved on average by 30% [1] through rearranging the *slack time* between the long and short paths. Clock skew scheduling improvements are described for unbounded amount of clock delays. In other words, t_i and t_f are modeled as continuous variables thereby allowing clock tree delays to have different values. In practice, clock delays can be changed by only a limited amount. This limitation is due to the size and discreteness of the delay values available. Consequently, the limited amount of delay insertion problem presented in this work is a practical implementation of clock skew scheduling. The proposed methodology introduces simultaneous limitations on delay insertion per branch and per clock tree. The simultaneous

limitations are proposed in order to more accurately reflect the practical limitations of an integrated circuit; that, the integrated circuit has a limited amount of area for delay buffering, which can be unevenly distributed between each clock branch. The limitation per clock tree is representative of the available space. The limitation per branch is to prevent exorbitant delay insertion on one branch. In this paper, the delay insertion method is explored with two purposes: (1) to fix the timing violations and (2) to optimize the circuit frequency with a very limited amount of delay insertion.

3.1. Challenge 1: Timing Violations. As the minimum feature size of VLSI circuits continues to shrink, process variations have become significantly worse [14]. The delay variations on clock network branches, for instance, correspond to 10% of their nominal value for deep sub-micron technologies [15]. This trend for global skew mismatches in recent microprocessors has been well documented [16]. Furthermore, the increasing functionality and speed of operation require a smaller clock period, which further complicate the timing closure of integrated circuits. Physical design tools are optimized to satisfy timing in presence of variations and the increasing clock frequencies. However, in practice, timing violations remain that require engineering change order (ECO) changes, such as the post-CTS methodology described in this paper.

3.2. Challenge 2: Clock Period Optimization. Consider the sample clock tree with N sinks shown in Figure 3. The clock tree is a balanced binary tree synthesized for a zero clock skew operation (without the delay buffers). The multidomain clock skew scheduling methodology [2] suggests the definition of multiple clock domains and the limitation of clock skew on each clock domain to a fixed percentage of the (zero clock skew) clock period T_{zs} . Consider that a single clock domain is selected for simplicity and the clock delay variation limit is set to 10% of the zero clock skew clock period. Such a limitation means a maximum skew of $10\% \times T_{zs}$ to be observed on the clock tree. In the *worst* case, the proposed delay insertion will be performed on $N - 1$ of

the N clock branches. This is such, as maximum insertion on each of the N branches would result in zero clock skew, which can be achieved with zero delay insertion as well. In this worst case, the total amount of insertion corresponds to a total delay insertion of $10\% \times T_{zs} \times (N - 1)$. It is more advantageous to use the insertion area corresponding to a total of $10\% \times T_{zs} \times (N - 1)$ time units as follows.

Instead of constraining the amount of insertion on each clock branch to a smaller number (e.g., 10%) that guarantees the overall insertion limitation in the worst case, the limitations on each branch are held more flexible. The adherence to the overall delay insertion budget is maintained with a general constraint that controls all of the branches at the same time. In other words, the sum of all delay insertion on each branch is limited to the same amount of $10\% \times T_{zs} \times (N - 1)$; however, the limitation on each branch is raised to a higher amount. Under the proposed scheme, some clock branches can be allocated more than the $10\% \times T_{zs}$ delay insertion whereas only a fraction of the clock branches can have a high (or maximum) delay insertion. Assume that this fraction is selected to be 0.5, thus, in the delay insertion process depicted in Figure 3(b), the maximum delay insertion on each branch is raised to $20\% \times T_{zs}$ (from $10\% \times T_{zs}$) but only half of the clock branches are allowed to accommodate the maximum delay insertion. For a high number of registers N , the overall delay insertion is approximately the same, that is, $10\% \times T_{zs} \times (N - 1) \approx 20\% \times T_{zs} \times N/2$.

4. Proposed Methodology

The traditional design flow with clock skew scheduling and the design flow with proposed method are illustrated in Figures 4(a) and 4(b), respectively. The proposed methodology analyzes each branch of a presynthesized clock tree (post-CTS) to explore the possibility of additional delay insertion only on the clock leaves. The proposed additional insertion is performed only to take place at the clock leaves, which are the sinks of the clock tree topology. Such delay insertion is advantageous in preserving most of the automated optimizations during the clock tree synthesis stage. It also requires less effort in order to fix the timing violations after verification since the new CTS step might not be necessary in the flow in Figure 4(b).

In the rest of the discussion and in experimentation, a zero skew clock tree is considered as the output of the clock tree synthesis step, and thus, the input to the proposed post-CTS delay insertion process. This simplification reflects the mainstream practice in clock tree synthesis in minimizing the clock skew subject to the system resource constraints (e.g., power, area, etc.). Nonetheless, the generality of the proposed discussion still holds for an arbitrary clock tree and slight modifications can be performed to handle any arbitrary tree.

In Section 4.1, a linear programming formulation is presented to fix timing violations. In Section 4.2, the mathematical framework proposed for clock period minimization is presented. In Section 4.3, discussions are presented based on presented formulations.

FIGURE 4: Integrated Circuit Design flow.

4.1. Formulation 1: Delay Insertion to Fix Timing Violations.

In mainstream IC design, automated placement and CTS tools are used to compute the physical implementation of the circuit. The logic and memory elements are placed with timing, congestion and, power driven objectives. The clock tree is implemented in one of the forms described in Section 2.1 to deliver identical delays to each synchronous component. Despite aggressive optimizations, however, clock network is still subject to random (and systematic) variations. These variations cause small shifts in the clock delays, leading to skew mismatches and potentially timing violations. A delay insertion method is proposed to be performed after the clock tree synthesis step (post-CTS) to fix the timing violations. The problem definition is

Given a pre-computed placement, and a synthesized clock tree of an IC (thus, given the clock delay t_i of each branch, clock period T_{zs} , local data path propagation time $[D_{PMin}, D_{PMax}]$ of each local data path, internal register delays S , H and D_{CQ} of each register), compute the minimum amount of delay Δ_i to be inserted on each clock tree branch in order to eliminate timing violations, considering upper bounds for delay insertion per branch and total delay insertion.

Note that, typically the last stage buffers of a clock tree drive more than one register. The presented formulation can be easily changed to reflect this requirement. For simplicity of presentation, each leaf buffer is selected to drive only one synchronous component.

TABLE 1: LP model for post-CTS delay insertion method.

Minimize inserted delay	
min	$\sum_{i=0}^{N-1} \Delta_i$
s.t.	$t_i + \Delta_i + D_{CQ}^i + D_{PMax}^{if} \leq t_f + \Delta_f + T_{zs} - S^f$
	$t_i + \Delta_i + D_{CQ}^i + D_{PMin}^{if} \geq t_f + \Delta_f + H^f$
	$\Delta_i \leq k_1 T_{zs}$
	$\sum_{i=0}^{N-1} \Delta_i \leq k_2 T_{zs} N$

The mathematical formulation for this problem is derived as an Linear Programming (LP) form. After post-CTS delay insertion, (1) and (2) can be written as

$$\text{Setup : } t_i + \Delta_i + D_{CQ}^i + D_{PMax}^{if} \leq t_f + \Delta_f + T_{zs} - S^f, \quad (7)$$

$$\text{Hold : } t_i + \Delta_i + D_{CQ}^i + D_{PMin}^{if} \geq t_f + \Delta_f + H^f, \quad (8)$$

where the added term Δ_i is the delay element on clock tree branch driving R_i . We also assume two practical limitations on the delay insertion process. First, we assume that the amount of delay to be inserted on a clock tree branch has an upper bound proportional to the overall clock period T_{zs} :

$$\Delta_i \leq k_1 T_{zs}, \quad (9)$$

where k_1 is a design parameter. Second, we assume that the total amount of delay to be inserted ($\sum_{i=0}^N \Delta_i$) has an upper bound proportional to the clock period T_{zs} and the number of registers N in the circuit:

$$\sum_{i=0}^{N-1} \Delta_i \leq k_2 T_{zs} N, \quad (10)$$

where k_2 is a design parameter. In a practical implementation, k_1 and k_2 can be determined by evaluating the physical design information such as the area utilization, the number of clock tree levels, and the power dissipation budget.

The LP model is shown in Table 1. The objective is to minimize the total amount delay insertion. The first two set of constraints are the setup and hold time constraints, respectively, defined for each local data path. The third set of constraints is the delay insertion upper bounds given in (9) defined for each clock branch. The fourth constraint is the total delay insertion bound given in (10). In this formulation, the clock arriving time t_i of each clock branch and the clock period are known. The value of delay insertion Δ_i necessary to fix the timing violations is obtained by solving the formulation.

The LP problem formulation guarantees minimum delay insertion. For instance, if no delay insertion is necessary, Δ_i evaluates to zero. For some circuits, the LP might return infeasibility which means either the timing violations cannot be resolved with the proposed delay insertion upper bounds or the circuit has reconvergent paths which are pathological cases [17] that cannot be solved with clock delay manipulation. Otherwise, the minimal amount of delay to be

FIGURE 5: A sample reconvergent path system. Clock delays t_d and t_c satisfy the timing of paths $R_d \rightarrow R_1$, $R_1 \rightarrow R_2$, $R_2 \rightarrow R_c$. However, the timing of one or both of paths $R_d \rightarrow R_3$, $R_3 \rightarrow R_c$ is violated.

inserted on each branch is returned as a continuous variable. A more detailed integer linear programming problem (ILP) formulation can also be devised to model the discrete values of delay to be inserted for a higher practical purpose.

The prescribed topology of a simple reconvergent path is shown in Figure 5. For *some* such systems, timing violations cannot be resolved by manipulating clock delay values as the timing of both branches depends on the clock delays at the divergent R_d and convergent R_c registers. As presented in [17], in such cases, *delay insertion into the logic network or reduction of clock frequency* is necessary.

4.2. Formulation 2: Clock Period Optimization. The post-CTS delay insertion methodology proposed for clock period optimization targets the objective of clock period minimization while preserving the original clock tree. The problem definition is

Given a pre-computed placement and a synthesized clock tree of an IC (thus, given the clock delay t_i of each branch, propagation time $[D_{PMin}^{ij}, D_{PMax}^{ij}]$ of each local data path, internal register delays S , H and D_{CQ} of each register), compute the amount of delay Δ_i to be inserted on each clock tree branch leaf in order to optimize the clock period, considering upper bounds for delay insertion per branch and total delay insertion.

The mathematical formulation for this problem is derived as a LP problem similar to the formulation in Section 4.1. One difference is that the objective of this LP is clock period minimization so the clock period T is not a known parameter. The resulting LP formulation is presented in Table 3.

The LP formulation guarantees minimum clock period operation with the amount of delay insertion specified by parameters k_1 and k_2 . The LP formulation always returns a feasible result, which in worst case is the zero clock skew clock period T_{zs} (i.e., if no improvements are possible through the specified amount of delay insertion). For higher amounts of delay insertion that are allowed, lower minimum clock periods are expected (not guaranteed). In the experiments presented in the next section, the consequences of the level of permitted delay insertion (i.e., k_1 and k_2) on the clock period improvement are analyzed experimentally to observe these trends.

TABLE 2: Post-CTS Delay insertion results for a suite of ISCAS'89 benchmark circuits.

Circuit info			Clock		Violation data			Insertion data	
Circuit	N	$\#paths$	T	$\#p^{vio}$	$\%p^{vio}$	$\sum vio$	$\sum \Delta$	$(\sum \Delta)/N$	Metric
<i>s1196</i>	18	20	20.8	1	5.0%	0.5	10.8	0.6	2.8%
<i>s1423</i>	74	1471	92.2	77	5.2%	1070.7	1531.8	20.7	22.5%
<i>s1488</i>	6	15	32.2	12	80.0%	80.8	61.2	10.2	31.7%
<i>s6669</i>	239	2138	128.6	68	3.2%	1630.2	6811.5	28.5	22.1%
<i>s9234</i>	228	247	75.8	86	34.8%	2204.3	N/A	N/A	N/A
<i>s13207</i>	669	3068	85.6	81	2.6%	1020.5	6279.2	9.4	11.0%
<i>s15850</i>	597	14257	116.0	406	2.8%	5025.4	11581.8	19.4	16.7%
<i>s15850.1</i>	534	10830	81.2	774	7.1%	8580.2	5767.2	10.8	13.3%
<i>s35932</i>	1728	4187	34.2	794	19.0%	6224.5	11232.0	6.5	19.0%
<i>s38417</i>	1636	28082	69.0	2206	7.9%	18811.2	12270.0	7.5	10.9%
<i>s38584</i>	1452	15545	94.2	178	1.1%	1837.1	34412.4	23.7	25.2%
<i>Average</i>					15.3%		8314.6	10.9	15.9%

4.3. Discussion. As discussed earlier, the delay insertion of the proposed work is performed at the post-CTS stage. In order to implement the delay insertion practically, some blocks of reserved white space must be allocated on the chip area. These blocks of white space should be reserved during the floorplanning stage. Depending on the size and the number of cells in the design, designers have to define the utilization area of the chip at the floorplanning stage. If the size requirement is not very strict, a low utilization factor of a chip can be defined in floorplanning so that the delay insertion space at post-CTS stage will be abundant to lead to a better result for fixing the timing violations or optimizing the frequency. If there is not enough space to insert post-CTS delay, a re-design of the layout (floorplanning) might be necessary.

5. Experimental Results

Proposed post-CTS optimization methods are used in experiments on a suite of ISCAS'89 benchmark circuits. A single phase clock signal with a 50% duty cycle is selected for synchronization. The internal register delays (i.e., setup, hold, clock-to-output times) are assumed negligible. The clock network is built experimentally as a zero skew clock tree. The experiments are performed on a 3.2 GHz Intel Xeon processor with a 16 GB RAM. The simplex optimizer of the GNU LP solver GLPK (version 4.31) [18] is used to solve the LP problems. The timing information for ISCAS'89 circuits is generated with a pre-determined algorithm, in which the fanout, size, and type of logic gates are considered. In the floorplanning stage, the utilization factor is chosen to be on the order of 40%.

5.1. Experiment 1: Fixing Timing Violations. In order to fix the timing violations with minimum delay insertion, the formulation in Table 1 is applied in the experiments. It is assumed that the clock period is selected as the largest data propagation delay in the circuit (which is typical in an ASIC design, see (5)), the clock delay t_i to each register is arbitrarily

selected to be $4T$ with a 10% variation which simulates the variation on the skew. Upper bounds of post-CTS clock delay insertion are set to be $0.8T$ on each branch ($k_1 = 0.8$) with a total delay insertion of $0.4TN$ ($k_2 = 0.4$). In a real application, all experimental assumptions can be easily changed according to automated tool results.

The results are presented on Table 2. In Table 2, circuit information, zero clock skew operation frequency, timing violation data, and post-CTS delay insertion data are presented. The numbers of registers and paths are shown in columns marked N and $\#paths$, respectively. The clock period T is selected as the largest data propagation delay in the circuit as derived in (5) to be functional for zero clock skew operation. The number of paths returning timing violation are shown in $\#p^{vio}$. The percentage of paths with timing violation are shown in $\%p^{vio}$. The total amount of violation (on all paths) is shown in column $\sum vio$. Post-CTS inserted delay information is presented in the last three columns, $\sum \Delta$ is the total inserted delay, $(\sum \Delta)/N$ is the average inserted delay per branch (register), and *metric* is a measure of the delay inserted per clock period, that is, $(\sum \Delta)/(NT)$. The metric is used as an arbitrary measure of inserted delay density, as the delay values increase with an increasing clock period regardless of the circuit complexity.

It is observed in Table 2 that post-CTS delay insertion on the clock network is applicable to all circuits except for *s9234* on the selected suite of circuits. Due to the 10% variations in delays—which are randomly generated in experimentation—timing violations occur on 15.3% of the paths but as many as 80% of all the paths (*s1488*) and as low as one (1) path (*s1196*) for a given circuit. The upper bounds of clock delay insertion, set by $k_1 = 0.8$ and $k_2 = 0.4$, enable us to fix the timing violations in most of the circuits by minimal delay insertion. The selected metric for delay insertion density

$$\text{Metric: } \frac{(\sum \Delta)}{(NT)} \quad (11)$$

has an average of 15.9%, which is reasonably small for a practical implementation.

FIGURE 6: Clock period optimization results.

TABLE 3: LP model for post-CTS delay insertion method.

Minimize inserted buffer delay	
min	T
s.t.	$t_i + \Delta_i + D_{CQ}^i + D_{PMax}^{if} \leq t_f + \Delta_f + T - S^j$ $t_i + \Delta_i + D_{CQ}^i + D_{PMin}^{if} \geq t_f + \Delta_f + H^f$ $\Delta_i \leq k_1 T_{zs}$ $\sum_{i=0}^{N-1} \Delta_i \leq k_2 T_{zs} N$

The timing violations for benchmark circuit s9234 cannot be resolved with post-CTS delay insertion because of *reconvergent paths* [17]. Although not observed in our experiments, the maximum delay insertion bounds on each clock branch $\Delta_i \leq k_1 T$ and the total delay insertion constraint $\sum \Delta < k_2 TN$ can also be limiting. For such circuits, designers can choose to follow typical procedures of performing iterative runs of placement, routing (or synthesis) to satisfy the specified timing budget. When such practices are costly, frequency specification can be relaxed to have the IC operate at a lower speed.

5.2. Experiment 2: Clock Period Optimization. In order to optimize the clock period, the formulation in Table 3 is applied in the experiments. In these experiments, upper bound of delay insertion on each branch of the clock tree is set to $k_1 \times T_{zs}$ and the upper bound of delay insertion on the clock tree is set to $k_2 \times T_{zs} \times N$, where N is number of

leaves in the clock tree and T_{zs} is the minimum clock period at zero clock skew. In the experiments, k_2 is set equal to one half of k_1 ($k_2 = 0.5k_1$), which suggests that the amount of delay insertion allowed on each tree branch is $k_1 \times T_{zs}$, while the total amount of delay insertion allowed on the tree is $0.5k_1 \times T_{zs} \times N$. As described in Section 4.2, such a correlation between k_1 and k_2 is used to have both constraints be binding as opposed to permitting excessive delay insertion for impractical clock period improvements. Additionally, this experimental setup enables a direct comparison with the previous work in [2] by providing the methodologies with identical total delay insertion resources. The comparison of results with the previous work in [2] is presented in Table 4. A “single”-domain application of the multidomain clock skew scheduling algorithm proposed in [2] is replicated in experimentation with skew scheduling ranges of 5% and 10% (0% case in [2] is the obvious zero clock skew case and needs not to be considered). In Table 4, the clock periods computed with both methodologies are presented as well as the *progress* of the improvement in clock period minimization. For instance, an improvement progress of 0% would indicate a zero clock skew operation while an improvement progress of 100% would indicate a design that is scheduled to operate at the minimum possible clock period with unlimited insertion. It is observed for both delay insertion bounds of 5% and 10% that the proposed post-CTS methodology consistently outperforms the multidomain clock skew scheduling methodology. On average, the proposed methodology is 2X and 1.6X better than [2] for skew scheduling ranges of 5% and 10%, respectively. As described in Section 3, the superiority of the proposed

TABLE 4: Clock period optimization with respect to the maximum possible improvement.

Circuit info	Clock period with total delay buffering $5\%T_{zs}N$				Clock period with total delay buffering $10\%T_{zs}N$			
	[2]	Improvement	Proposed	Improvement	[2]	Improvement	Proposed	Improvement
<i>s1196</i>	17.36	34%	14.24	66%	16.32	45%	12.16	86%
<i>s1423</i>	87.59	31%	82.98	62%	82.98	62%	77.40	100%
<i>s1488</i>	30.59	50%	29.09	97%	29.09	97%	29.00	100%
<i>s6669</i>	122.17	34%	115.74	68%	115.74	68%	109.80	100%
<i>s9234</i>	299.41	7%	295.62	13%	295.62	13%	288.04	27%
<i>s13207</i>	81.32	15%	77.04	30%	77.04	30%	68.48	60%
<i>s15850</i>	110.20	18%	104.4	36%	104.40	36%	92.8	72%
<i>s15850.1</i>	77.14	17%	73.08	34%	73.08	34%	65.18	67%
<i>s35932</i>	33.99	2%	33.77	3%	33.77	3%	33.35	6%
<i>s38417</i>	65.55	13%	62.10	26%	62.10	26%	55.20	51%
<i>s38584</i>	89.49	16%	84.78	32%	84.78	32%	75.36	65%
<i>Average</i>		22%		43%		41%		67%

TABLE 5: Post-CTS limited delay insertion clock periods for a suite of ISCAS'89 benchmark circuits.

Circuit info	Clock period									
	$k_1 = 0$	$k_1 = 0.1$	$k_1 = 0.2$	$k_1 = 0.3$	$k_1 = 0.4$	$k_1 = 0.5$	$k_1 = 0.6$	$k_1 = 0.7$	$k_1 = 0.8$	CSS ($k_1 = \infty$)
<i>s1196</i>	20.80	14.24	12.16	10.80	10.80	10.80	10.80	10.80	10.80	10.80
<i>s1423</i>	92.20	82.98	77.40	77.40	77.40	77.40	77.40	77.40	77.40	77.40
<i>s1488</i>	32.20	29.09	29.00	29.00	29.00	29.00	29.00	29.00	29.00	29.00
<i>s6669</i>	128.60	115.74	109.80	109.80	109.80	109.80	109.80	109.80	109.80	109.80
<i>s9234</i>	303.20	295.62	288.04	280.46	272.88	265.30	257.72	250.14	246.00	246.00
<i>s13207</i>	85.60	77.04	68.48	60.16	57.10	57.10	57.10	57.10	57.10	57.10
<i>s15850</i>	116.00	104.40	92.80	83.60	83.60	83.60	83.60	83.60	83.60	83.60
<i>s15850.1</i>	81.20	73.08	65.18	61.12	57.40	57.40	57.40	57.40	57.40	57.40
<i>s35932</i>	34.20	33.77	33.35	32.92	32.49	32.06	31.64	31.21	30.78	20.4
<i>s38417</i>	69.00	62.10	55.20	48.30	42.20	42.20	42.20	42.20	42.20	42.20
<i>s38584</i>	94.20	84.78	75.36	66.77	65.20	65.20	65.20	65.20	65.20	65.20

methodology is due to the flexibility of bounds on each clock branch and global monitoring of overall delay insertion.

Next, the parameters k_1 and k_2 are gradually increased to observe the change in clock period optimization through various levels of delay insertion.

In Table 5, the clock period improvements for varying delay insertion bounds between $k_1 = 0$ and $k_1 = 0.8$ are presented. The last column in Table 5 presents the unbounded clock skew scheduling result, that is, an upper bound of $k_1 = k_2 = \infty$. It is confirmed with experimentation that with increasing k_1 and k_2 , the clock period is monotonously improved. An important observation is the delay insertion bounds at which significant progress is obtained in clock period minimization. For most of the circuits, the majority of the clock period improvement are achieved with delay insertion with an upper bound of 10% to 20% times T_{zs} on each branch. As demonstrated here, delay insertion budgets for clock period minimization can be devised more accurately so as to not waste design resources for relatively smaller improvements to be achieved for additional delay insertion over a certain bound.

In Figure 6(a), the minimum clock period with varying bounds of delay insertion is normalized with respect to the zero clock skew minimum clock period T_{zs} . In Figure 6(b), the clock period improvements with varying levels of insertion are presented as a percentage of maximum possible clock period improvement. In Figure 6(b), each curve starts from $k_1 = k_2 = 0$, which implies no delay insertion, thus no improvement in clock period minimization. The value “1” in the figure implies the maximum level of improvement (e.g., 100%) in clock period optimization is achieved. It is observed that nine (9) of the eleven (11) circuits can be optimized to more than 90% of the optimal solution with the post-CTS method using only a limit amount of delay insertion corresponding to $k_1 = 0.3$. Numerically, with k_1 set to 0.2 and k_2 set to 0.1, nine (9) out of eleven (11) circuits exhibit more than 20% of clock period improvement and seven (7) of them have improvements of over 50%. The average improvement in the clock period minimization is 67% for the selected suite of circuits, demonstrating the high level of improvement with limited delay insertion.

6. Conclusions

The post-CTS clock delay insertion method has the motivation of observing the limited amount of delay insertion space on an integrated circuit and utilizing this area more efficiently by simultaneously limiting the delay insertion per branch and the clock tree. The proposed method is analyzed for the objectives of fixing timing violations and clock period optimization. The proposed method has the following advantages.

- (i) The proposed method is performed post-CTS, which requires lower efforts to fix timing violations after verification.
- (ii) The proposed method starts off with the CTS results and only permits minimal delay insertion, which keeps the clock delays easy to realize.

A first set of experiments is performed to observe the advantages of limited delay insertion for circuits with timing violations. In experimentation with the generated ISCAS'89 clock networks, it is found that a 10% variation in the delay values of the clock network can result in 15.3% of the timing paths to fail the timing requirements. By applying the proposed method, timing violations are successfully resolved in ten (10) of the eleven (11) experimented circuits. A second set of experiments is performed to observe the advantages of limited insertion for circuits where no timing violations exist. By applying the clock period optimization method, the clock period can be improved by an average of 43% with a very limited amount of delay insertion. In practice, post-CTS delay insertion method can be used by designers to find quick solutions to timing violation or clock period optimization problems without having to go through lengthy synthesis-placement-routing iterations.

References

- [1] I. S. Kourtev, B. Taskin, and E. G. Friedman, *Timing Optimization through Clock Skew Scheduling*, Springer, New York, NY, USA, 2009.
- [2] K. Ravindran, A. Kuehlmann, and E. Sentovich, "Multi-domain clock skew scheduling," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD '03)*, pp. 801–808, San Jose, Calif, USA, November 2003.
- [3] Q. K. Wu, *High-Speed Clock Network Design*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.
- [4] E. G. Friedman, *Clock Distribution Networks in VLSI Circuits and Systems*, IEEE Press, New York, NY, USA, 1995.
- [5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, Cambridge, Mass, USA, 2nd edition, 2001.
- [6] M. A. B. Jackson, A. Srinivasan, and E. S. Kuh, "Clock routing for high-performance ICs," in *Proceedings of the ACM/IEEE Design Automation Conference (DAC '90)*, pp. 573–579, Orlando, Fla, USA, June 1990.
- [7] R.-S. Tsay, "An exact zero-skew clock routing algorithm," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 2, pp. 242–249, 1993.
- [8] N.-C. Chou and C.-K. Cheng, "On general zero-skew clock net construction," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 3, no. 1, pp. 141–146, 1995.
- [9] N. Ito, H. Sugiyama, and T. Konno, "ChipPRISM: clock routing and timing analysis for high-performance CMOS VLSI chips," *Fujitsu Scientific and Technical Journal*, vol. 31, no. 2, pp. 180–187, 1995.
- [10] N. Gaddis and J. Lotz, "A 64-b quad-issue CMOS RISC microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 11, pp. 1697–1702, 1996.
- [11] S. Rusu and G. Singer, "The first IA-64 microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 11, pp. 1539–1544, 2000.
- [12] W. J. Bowhill, S. L. Bell, B. J. Benschnieder, et al., "Circuit implementation of a 300-MHz 64-bit second-generation CMOS alpha CPU," *Digital Technical Journal*, vol. 7, no. 1, pp. 100–118, 1995.
- [13] W.-K. Chen, Ed., *The VLSI Handbook*, CRC Press, Boca Raton, Fla, USA, 1st edition, 1999.
- [14] S. R. Nassif, "Modeling and analysis of manufacturing variations," in *Proceedings of the IEEE Custom Integrated Circuits Conference (CICC '01)*, pp. 223–228, San Diego, Calif, USA, May 2001.
- [15] A. B. Kahng, "A roadmap and vision for physical design," in *Proceedings of the IEEE International Symposium on Physical Design (ISPD '02)*, pp. 112–117, Del Mar, Calif, USA, April 2002.
- [16] A. V. Mule, E. N. Glytsis, T. K. Gaylord, and J. D. Meindl, "Electrical and optical clock distribution networks for gigascale microprocessors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 10, no. 5, pp. 582–594, 2002.
- [17] B. Taskin and I. S. Kourtev, "Delay insertion method in clock skew scheduling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 4, pp. 651–663, 2006.
- [18] Free Software Foundation (FSF), "GLPK (GNU Linear Programming Kit)," 2008, <http://www.gnu.org/software/glpk/glpk.html>.