JOURNAL OF SEMICONDUCTOR TECHNOLOGY AND SCIENCE, VOL.17, NO.6, DECEMBER, 2017
https://doi.org/10.5573/JSTS.2017.17.6.862

ISSN(Print) 1598-1657
ISSN(Online) 2233-4866

Circuit Timing Analysis and Optimization under Flexible Flip-flop Timing Model

Jeongwoo Heo and Taewhan Kim

Abstract—**Most of static timing analyses today are based on conventional fixed flip-flop timing models, in which every flip-flop is assumed to have a fixed clock-to-Q delay. However, since the setup and hold skews affect the clock-to-Q delay in reality, many researchers have paid attention to the flexible flip-flop timing property in circuit timing analysis and optimization. Nevertheless, the existing works have failed in solving the problem of exactly formulating the sequences of timing dependency among the skews and delays over time. In this work, we solve the problem, specifically, (1) *proposing a mathematical formulation to solve the problem,* (2) *applying our solution to the analysis of a given circuit as well as* (3) *to the clock skew scheduling problems,* and (4) *proposing a speedup technique.* Through experiments with benchmark circuits, it is demonstrated that our method can rectify improper or unoptimized results drawn from the previous approach. It is also shown that our speedup technique is able to shorten run times significantly with very little loss of optimality.**

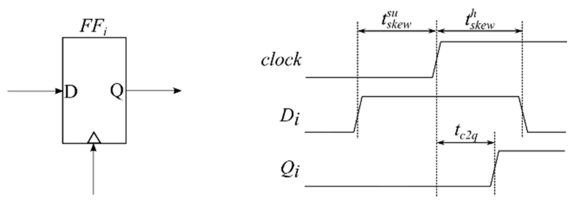
Index Terms—**Static timing analysis, flexible flip-flop, clock skew scheduling**

Manuscript received Apr. 6, 2017; accepted Sep. 19, 2017
The preliminary version of this work was presented in IEEE Computer Society Annual Symposium on VLSI (ISVLSI), July 2016 [1].
Department of Electrical and Computer Engineering, Seoul National University, Seoul, Korea
E-mail : {jwheo, tkim}@snucad.snu.ac.kr

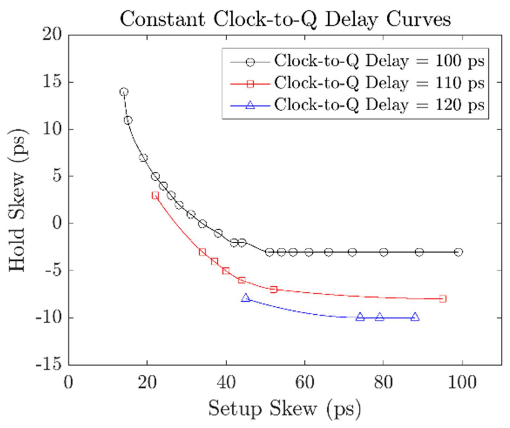
I. Introduction

Almost all static timing analysis techniques used today for evaluating timing performance of sequential circuits are based on setup and hold times constrained flip-flop timing model. This timing model entails that a clock-to-Q delay of a flip-flop is set to a constant value and the setup and hold times are measured at the points of 5~10% clock-to-Q delay increases with additional timing margin for various variations. However, it has been regarded that a clock-to-Q delay has nothing to do with the setup and hold skews of the flip-flop; consequently, the timing model simply sets the setup and hold times to be unnecessarily large in order to overly constrain timing at flip-flops, so that the actual clock-to-Q delay is always shorter than the (assumed) maximum clock-to-Q delay. Thus, this *fixed flip-flop timing model* is safe in that if there is no timing violation, it ensures that the circuit has really no setup and hold time violation at all. However, due to the pessimistic setup and hold time constraints independently imposed on flip-flops, the timing reports may not be so accurate.

Fig. 1 shows SPICE simulation results for the change of the values of a setup skew, a hold skew, and a clock-to-Q delay. Illustration for these notions is shown in Fig. 1(a). The three curves in Fig. 1(b) show the trade-off between setup and hold skews for the clock-to-Q delays of 100 ps, 110 ps, and 120 ps, respectively. These curves clearly reveal that timing analysis and optimization should take into account the interdependent relation between a setup skew, a hold skew, and a clock-to-Q delay, if we want to avoid unnecessary late-state engineering change order (ECO) or high cost design change before timing sign-off due to inaccurate analysis



(a) The values of the setup skew t^{su}_{skew} and the hold skew t^{h}_{skew} of FF_i determine the value of the clock-to-Q delay t_{c2q} of FF_i. t_{c2q} would affect the values of t^{su}_{skew} and t^{h}_{skew} of the flip-flops to be driven by FF_i.



(b) Curves showing the relationship between a setup skew and a hold skew for the clock-to-Q delays of 100 ps, 110 ps, and 120 ps. The flexibility of trading the setup skew with the hold skew increases as the clock-to-Q delay decreases.

Fig. 1. Illustration for the interdependent relation between a setup skew, a hold skew, and a clock-to-Q delay. The curves were produced by SPICE simulation with 45 nm NanGate Open Cell Library [2].

of the timing behavior of a circuit. In this paper, we refer to this property as *flexible flip-flop timing* to differentiate it from the concept of the conventional fixed flip-flop timing.

In the past, many researchers studied about characterization, analysis, and optimization methods of the flexible flip-flop timing model widely. Stojanovic and Oklobdzija [3] focused on the analysis of the relationship between a setup skew and a clock-to-Q delay, and minimized the sum of them. Jain and Blaauw [4] modeled the relation between a clock-to-Q delay and a setup skew in the form of an exponential expression to improve the analysis accuracy over that in [3]. Both of the works, however, did not consider the effect of a hold skew on a clock-to-Q delay. To overcome this shortcoming, Rao and Howick [5] firstly clarified the existence of the close interaction between a setup skew, a hold skew, and a clock-to-Q delay. Srivastava and Roychowdhury [6, 7] developed a methodology of fast characterization of the timing relation by employing Newton-Raphson solution and Eular curve tracking of state-transition equations. Salman *et al.* [8, 9] enhanced the accuracy of timing analysis by reflecting the interdependent relation of setup and hold times and a clock-to-Q delay. Later, Salman and Friedman [10] attempted to utilize the interdependent timing relation to tolerate delay variations, so that delay uncertainty be reduced. While those works focused on static timing analysis, Hatami *et al.* [11] demonstrated, through Monte Carlo simulation, that the flexible flip-flop timing model can significantly improve the accuracy in statistical static timing analysis.

In the perspective of analyzing and optimizing circuit timing, Chen *et al.* [12] addressed the clock period minimization problem and proposed a simple and accurate analytical model of a clock-to-Q delay surface. They also expressed an iterative timing analysis method, in which the clock-to-Q delay of a flip-flop is iteratively computed, starting from a valid clock-to-Q delay value, and reports a timing violation on a flip-flop if the clock-to-Q delay value of the flip-flop does not converge. One critical limitation is that if the initial delay value is improperly set, it may lead to a false negative result. They tried to find the minimum clock period of a circuit by repeatedly applying this analysis technique in a binary search framework. Kahng and Lee [13] divided the flexible flip-flop timing model based timing margin recovery problem into two optimization subproblems: (1) optimization between a setup time and a clock-to-Q delay and (2) optimization between a hold time and a clock-to-Q delay to enable linear programming formulations. However, the linearization of flexible flip-flop timing model causes non-trivial fitting errors. On the other hand, Seo *et al.* [14, 15] considered the flexible flip-flop timing in the useful clock skew scheduling problem. They proposed a stepwise clock skew scheduling technique, in which at each iteration, setup and hold slacks are systematically and incrementally relaxed by utilizing flexible flip-flop timing based analysis method in [12]. Though the approach is practical and fast, the result could be far from the optimal solution due to its ad hoc nature. Recently, Yang *et al.* [16] attempted to compute timing relation accurately only on the timing critical part of a circuit.

The contributions of this work are (a short version of this work is in [1]) (1) proposing the analysis method under flexible flip-flop timing model derived from a comprehensive interval analysis of a clock-to-Q delay, with a demonstration of applications to two examples, (2) formulating the problem of finding minimum clock period of a given circuit, which is one of the representative applications of our flexible flip-flop timing analysis, into a convex optimization problem, (3) formulating the clock skew scheduling problems for maximizing worst and total slacks into convex optimization problems, and (4) proposing a speedup technique.

II. Preliminaries

1. Terminologies

Let us consider the circuit consisting of two flip-flops FF_i and FF_j shown in Fig. 2, and let $x(i)$, $x(j)$, $D_{i,j}^{max}$, and $D_{i,j}^{min}$ in the figure be clock arrival times at FF_i and FF_j and maximum and minimum propagation delays of the combinational logic from FF_i to FF_j, respectively. Then the static timing analysis terms are formally defined as follows:

- *Setup skew* $t_{skew}^{su}(j)$: The setup skew $t_{skew}^{su}(j)$ of FF_j is the time interval ending at the clock's active edge of the flip-flop during which the input data to FF_j is stable.
- *Hold skew* $t_{skew}^{h}(j)$: The hold skew $t_{skew}^{h}(j)$ of FF_j is the time interval starting at the clock's active edge of the flip-flop during which the input data to FF_j is stable.
- *Clock*-to-Q delay $t_{c2q}(j)$: The clock-to-Q delay $t_{c2q}(j)$ of FF_j refers to the time lapse between the clock's arrival edge of the flip-flop and the time to register the input data to the output of FF_j. If FF_j's setup skew $t_{skew}^{su}(j)$, hold skew $t_{skew}^{h}(j)$, and clock-to-Q delay surface $\mathcal{F}_j$ are given, $t_{c2q}(j)$ could be expressed as

$$t_{c2q}(j) = \mathcal{F}_j\left(t_{skew}^{su}(j), t_{skew}^{h}(j)\right). \tag{1}$$

- *Setup time* $t_{time}^{su}(j)$: The setup time value $t_{time}^{su}(j)$ of FF_j is a lower bound of the setup skew of FF_j set by a designer to make sure a successful latching at FF_j.
- *Hold time* $t_{time}^{h}(j)$: The hold time value $t_{time}^{h}(j)$ of FF_j is a lower bound of the hold skew of FF_j set by a designer to make sure a successful latching at FF_j,
- *Setup slack* $t_{slk}^{su}(j)$: The setup slack $t_{slk}^{su}(j)$ of FF_j is the extra setup skew available beyond the setup time $t_{time}^{su}(j)$ of FF_j with respect to the input data, i.e.,

$$t_{slk}^{su}(j) = t_{skew}^{su}(j) - t_{time}^{su}(j). \tag{2}$$

- *Hold slack* $t_{slk}^{h}(j)$: The hold slack $t_{slk}^{h}(j)$ of FF_j is the extra hold skew available beyond the hold time $t_{time}^{h}(j)$ of FF_j with respect to the input data, i.e.,

$$t_{slk}^{h}(j) = t_{skew}^{h}(j) - t_{time}^{h}(j). \tag{3}$$

- *Worst slack* t_{slk}^{w}: The worst slack t_{slk}^{w} of a circuit is the minimum value of the setup and hold slacks for all flip-flops in the circuit, i.e.,

$$t_{slk}^{w} = \min_{FF_j}\{t_{slk}^{su}(j), t_{slk}^{h}(j)\} \tag{4}$$

- *Total slack* t_{slk}^{tot}: The total slack t_{slk}^{tot} of a circuit is

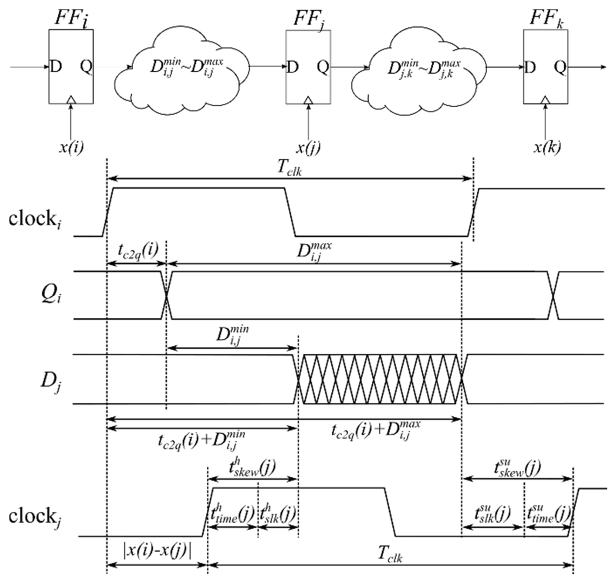


Fig. 2. A simple circuit and its timing diagram.

the summation of the minimum slack values all flip-flops have, i.e.,

$$t_{slk}^{tot} = \sum_{FF_j} \min\{t_{slk}^{su}(j), t_{slk}^{h}(j)\}. \tag{5}$$

The bottom of Fig. 2 shows pictorial representation of the terminologies with timing diagram.

2. Timing Analysis

A target circuit is first decomposed into the set of all timing paths and the maximum and the minimum signal propagation delays of each path are calculated. Then, timing validation at all flip-flops and primary inputs and outputs is checked. For simplicity, consider two fundamental timing paths: a data path and a clock path. A data path is a path that propagates logical signals. It starts from a primary input port or an output data pin of a flip-flop and ends at an input data pin of a flip-flop or a primary output port. A clock path, on the other hand, is a path that propagates clock signals. It starts from a clock source port and ends at each flip-flop's input clock pin. The delay of a clock path includes not only the delays of buffers or inverters but also interconnect delays.

In a synchronous digital system, there are two timing errors related to flip-flops or latches: a setup time violation and a hold time violation. A setup time violation occurs when the data signal arrives *too late* relative to the active clock transition, while a hold time violation occurs when it arrives *too soon*. To avoid the timing violations, designers introduce a *setup time constraint* $t_{skew}^{su}(j) \geq t_{time}^{su}(j)$ (or equivalently, $t_{slk}^{su}(j) \geq 0$) for each FF_j, and a *hold time constraint* $t_{skew}^{h}(j) \geq t_{time}^{h}(j)$ (or equivalently, $t_{slk}^{h}(j) \geq 0$) for each FF_j. For example, for the circuit in Fig. 2, the setup and hold skews of FF_j are computed by

$$t_{skew}^{su}(j) = \left(T_{clk} + x(j)\right) - \left(x(i) + t_{c2q}(i) + D_{i,j}\right), \tag{6a}$$
$$t_{skew}^{h}(j) = \left(x(i) + t_{c2q}(i) + D_{i,j}\right) - x(j), \tag{6b}$$

where $D_{i,j}^{min} \leq D_{i,j} \leq D_{i,j}^{max}$. The computed values are compared with the preset values of the setup time and the hold time of FF_j. If both of the skews are larger, the analyzer will ensure a correct operation at FF_j.

3. Clock-to-Q Delay Surface Modeling

One of the most important and basic tasks for flexible flip-flop timing based analysis and optimization is characterizing a clock-to-Q delay surface. A clock-to-Q delay surface is a three dimensional graph that describes the change of the clock-to-Q delay of a flip-flop for various pairs of setup and hold skew values. Its x, y, and z axes denote a flip-flop's setup skew, hold skew, and the corresponding clock-to-Q delay, respectively, as shown in Fig. 3.

For modeling a clock-to-Q delay surface, Chen *et al.* [12] tried to fit the SPICE simulation results into the following analytic function

$$t_{c2q} = a_0 + \frac{a_1}{t_{skew}^{su} - s_0} + \frac{a_2}{t_{skew}^{h} - h_0} \tag{7}$$

in which a_0, a_1, a_2, s_0, and h_0 are the fitting coefficients. It is a nonlinear fitting problem as the form of the function suggests, and it can be effective to use nonlinear least squares, which is used to fit a set of observations with a model consisting of unknown parameters. Contrary to Chen *et al.* [12], Kahng and Lee [13] suggested to exploit linearly approximated relations between setup skews and clock-to-Q delays for given hold skews, or hold skews and clock-to-Q delays for given setup skews. Conventionally this type of task is categorized as a linear fitting problem, and various objectives can be utilized for the problem. One possible

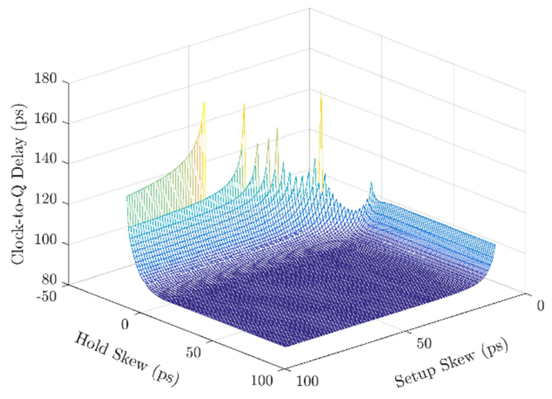


Fig. 3. View of a clock-to-Q delay surface. All data of $(t_{skew}^{su}, t_{skew}^{h}, t_{c2q})$ are extracted at every 1 ps of timing points through SPICE simulations with 45 nm NanGate Open Cell Library [2], where t_{skew}^{su} and t_{skew}^{h} are characterized over the range of 0~100 ps and -20~100 ps, respectively.

direction is linear least squares, which minimizes the sum of all residuals for given data points.

More accurate clock-to-Q delay surface modeling than that in Eq. (7) would also be possible. For instance, the latest convex regression techniques such as [17-20], and other various works of applied mathematics communities can be exploited for clock-to-Q delay surface modeling. It should be noted that our timing analyzer can use any existing clock-to-Q delay surface model as long as it is convex.

III. Clock-to-Q Delay Interval Analysis

Since the clock-to-Q delay at a certain clock cycle depends on the setup skew at the previous clock cycle and the hold skew at the current clock cycle, the clock-to-Q delay at cycle c of FF_j in Fig. 2 can be written as

$$\begin{aligned} t_{skew}^{su,(c-1)}(j) &= \left(T_{clk} + x(j)\right) - \\ &\quad \left(x(i) + t_{c2q}^{(c-1)}(i) + D_{i,j}^{max,(c-1)}\right), \end{aligned} \tag{8a}$$

$$t_{skew}^{h,(c)}(j) = \left(x(i) + t_{c2q}^{(c)}(i) + D_{i,j}^{min,(c)}\right) - x(j), \tag{8b}$$

$$t_{c2q}^{(c)}(j) = \mathcal{F}_j\left(t_{skew}^{su,(c-1)}(j), t_{skew}^{h,(c)}(j)\right). \tag{8c}$$

Let us assume that all logical operations at every clock cycle are independent to each other as being done in conventional static timing analysis. Then it is possible to have the case that $t_{c2q}^{(c)}$ is very large while $t_{c2q}^{(c+1)}$ is very small or vice versa; therefore, the concept of the convergence in the pointwise manner used in [12-15] would not result an accurate solution, and it is more natural to consider the convergence based on clock-to-Q delay intervals.

Let $t_{c2q}^{min}(i)$ and $t_{c2q}^{max}(i)$ be the minimum and maximum clock-to-Q delays of FF_i, respectively. Then, the minimum setup skew $t_{skew}^{su,min}(j)$ and the minimum hold skew $t_{skew}^{h,min}(j)$ at FF_j are

$$\begin{aligned} t_{skew}^{su,min}(j) &= \left(T_{clk} + x(j)\right) - \\ &\quad \left(x(i) + t_{c2q}^{max}(i) + D_{i,j}^{max}\right), \end{aligned} \tag{9a}$$

$$t_{skew}^{h,min}(j) = \left(x(i) + t_{c2q}^{min}(i) + D_{i,j}^{min}\right) - x(j). \tag{9b}$$

The maximum setup skew $t_{skew}^{su,max}(j)$ and the maximum hold skew $t_{skew}^{h,max}(j)$ at FF_j are also expressed similarly. By reasoning the more delay is required to charge or discharge the internal capacitor of a flip-flop as the setup skew and the hold skew diminish, the minimum and the maximum clock-to-Q delays at FF_j can be expressed as

$$t_{c2q}^{min}(j) = \mathcal{F}_j\left(t_{skew}^{su,max}(j), t_{skew}^{h,max}(j)\right), \tag{10a}$$

$$t_{c2q}^{max}(j) = \mathcal{F}_j\left(t_{skew}^{su,min}(j), t_{skew}^{h,min}(j)\right), \tag{10b}$$

where $\mathcal{F}_j$ is the clock-to-Q delay surface of FF_j. By the same token, the procedure for deriving the interval of the clock-to-Q delay at FF_k of the circuit in Fig. 2 can be expressed as

$$\begin{aligned} t_{skew}^{su,min}(k) &= \left(T_{clk} + x(k)\right) - \\ &\quad \left(x(j) + t_{c2q}^{max}(j) + D_{j,k}^{max}\right), \end{aligned} \tag{11a}$$

$$t_{skew}^{h,min}(k) = \left(x(j) + t_{c2q}^{min}(j) + D_{j,k}^{min}\right) - x(k), \tag{11b}$$

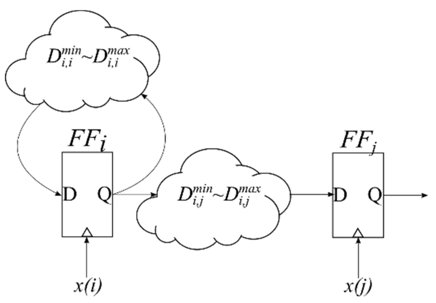
$$t_{c2q}^{min}(k) = \mathcal{F}_k\left(t_{skew}^{su,max}(k), t_{skew}^{h,max}(k)\right), \tag{11c}$$

$$t_{c2q}^{max}(k) = \mathcal{F}_k\left(t_{skew}^{su,min}(k), t_{skew}^{h,min}(k)\right). \tag{11d}$$

By repeating this process for all flip-flops, the intervals of their clock-to-Q delays as well as the timing relations between their parameters being involved can be obtained easily.

Now let us find out the impacts of the analysis methods, i.e., the conventional pointwise convergence based analysis used in [12-15] and our clock-to-Q delay interval based analysis, with two example circuits. For clarity, we denote clock-to-Q delays obtained from pointwise convergence based analysis with an asterisk.

Case 1: The circuit of Fig. 2: Let us suppose that FF_i has a specific value of a clock-to-Q delay, i.e., $t_{c2q}(i)$, initially. From the clock-to-Q delay of FF_i, previous works compute the setup skew and the hold skew of FF_j, from which the clock-to-Q delay of FF_j, i.e., $t_{c2q}^{*}(j)$, is determined and marked with a red dot in Fig. 5(a). Likewise, the worst skew scenario of a timing path to FF_k together with $t_{c2q}^{*}(j)$ will provide the clock-to-Q delay of FF_k, i.e., $t_{c2q}^{*}(k)$, as illustrated in Fig. 5(b). However, FF_j can have a smaller clock-to-Q delay than $t_{c2q}^{*}(j)$ in reality because of the independence between the two clock-to-Q delays used in the calculation of the setup skew and the hold skew of FF_j; in other words, all points in the blue region of Fig. 5(a) can be a pair of the



$D_{i,i}^{min}$~$D_{i,i}^{max}$
FF_i
D Q
$D_{i,j}^{min}$~$D_{i,j}^{max}$
FF_j
D Q
x(i)
x(j)

Fig. 4. A simple circuit.

setup skew and the hold skew of FF_j. The minimum setup skew $t_{skew}^{su,min}(k)$ and the minimum hold skew $t_{skew}^{h,min}(k)$ at FF_k are then calculated like Eq. (11), and we can easily deduce that the gap between $t_{c2q}^{*}(j)$ and $t_{c2q}^{min}(j)$ causes reduction of the minimum hold skew of FF_k in Fig. 5(b). (Note that the clock-to-Q delay value increases as the point gets closer to the separation curve.) As a result, $t_{c2q}^{max}(k)$ would be located below the curve, and therefore, against the expectation based on the previous thought, the circuit may run into the danger of falling to the timing failure region.

Case 2: The circuit of Fig. 4: Let us suppose that the circuit of Fig. 4 has a simple feedback loop. Unlike the previous case, through the iteration, the location of the clock-to-Q delay of FF_i, i.e., $t_{c2q}^{*}(i)$, may be far from the blue box bounded by the minimum and the maximum of the setup skews and the hold skews of FF_i, as shown in Fig. 5(c). If $t_{c2q}^{*}(i) \geq t_{c2q}^{max}(i)$, the minimum setup skew of FF_j calculated from $t_{c2q}^{max}(i)$ will be larger than that from $t_{c2q}^{*}(i)$ as shown in Fig. 5(d) according to Eq. (9a). Similarly, the minimum hold skew of FF_j obtained from $t_{c2q}^{min}(i)$ is smaller than that from $t_{c2q}^{*}(i)$, due to $t_{c2q}^{*}(i) \geq t_{c2q}^{min}(i)$ if $t_{c2q}^{*}(i) \geq t_{c2q}^{max}(i)$, as shown in the same figure. As a result, contrary to the case of the circuit in Fig. 2, there can be a room for further timing optimization, e.g. increasing the clock frequency.

IV. Applications of Clock-to-Q Delay Interval Analysis under Flexible Flip-flop Timing Model

In this section, we apply our clock-to-Q delay interval analysis presented in section III to two representative applications of flexible flip-flop timing model: (1) finding minimum clock period of a given circuit design, (2) clock skew scheduling for improving worst slack, clock period, and total slack. Note that (1) is an analysis problem while (2) is an optimization problem. In subsection IV-1, we add two additional constraints, i.e., an interacting constraint and a clock-to-Q delay constraint, and then formulate the problem into a convex programming based on the constraints, in subsection IV-2. In addition, we apply our concept to clock skew scheduling problems in subsection IV-3. Finally, we explain our speedup technique for those applications in subsection IV-4.

1. Additional Constraints

(1) *Formulating interacting constraints of setup and hold times*: Setup and hold times are treated independently under fixed flip-flop timing model. However, in flexible flip-flop timing model based analysis, we should consider the interaction of them through clock-to-Q delay surface $\mathcal{F}(t_{skew}^{su}, t_{skew}^{h})$. A flip-flop is able to latch its data with a certain pair of setup and hold skews, so there exists a boundary curve

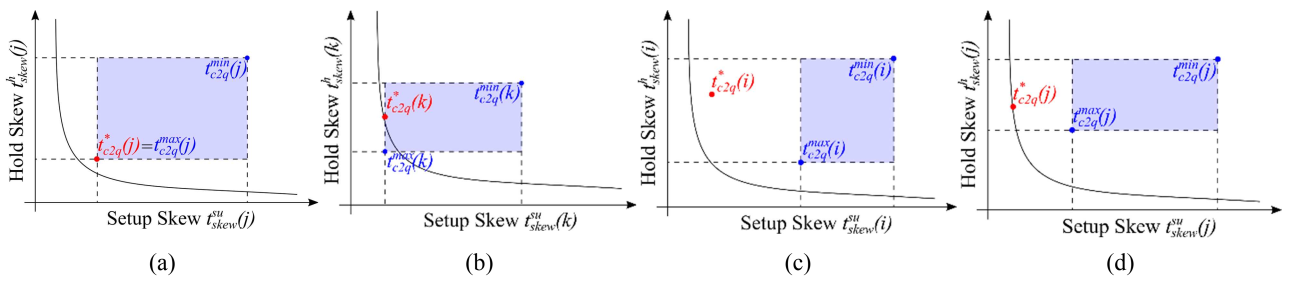


Fig. 5. (a)-(b) Timing analyses of FF_j and FF_k of the circuit in Fig. 2. (c)-(d) Timing analyses of FF_i and FF_j of the circuit in Fig. 4. In all figures, the outcomes based on the previous notion and ours are in red and blue, respectively. Each plot represents the top view of the clock-to-Q delay surface of the corresponding flip-flop, and each curve represents the boundary between success and failure regions of latching. Therefore, the points located above the curve are safe while the others are in danger.

among the pairs. To represent this curve and include it in our formulation, we create the function $\mathcal{G}(t_{skew}^{su}, t_{skew}^{h})$ and let $\mathcal{G}$ has negative values in the feasible region while positive values on the opposite side. Then the setup and hold times we can feasibly set with no timing violations are in the region including the curve that satisfies

$$\mathcal{G}(t_{skew}^{su}, t_{skew}^{h}) \leq 0, \tag{12}$$

as shown in Fig. 6. Note that since there is abundant choice of selecting setup and hold times, i.e., all points satisfying Eq. (12), the exploration of alternatives will enable the timing analysis and optimization to be more flexible. One simple method to create the function $\mathcal{G}$ is to set

$$\mathcal{G}(t_{skew}^{su}, t_{skew}^{h}) = \mathcal{F}(t_{skew}^{su}, t_{skew}^{h}) - t_{c2q}^{upr}, \tag{13}$$

with an upper bound of a clock-to-Q delay value t_{c2q}^{upr}.

(2) Formulating clock-to-Q delay constraints: The clock-to-Q delay constraint at FF_j is exactly the boundary constrained by Eq. (10), which can be rewritten as:

$$\mathcal{F}_j\left(t_{skew}^{su,min}(j), t_{skew}^{h,min}(j)\right) \leq t_{c2q}^{max}(j), \tag{14a}$$

$$t_{c2q}^{min}(j) \leq \mathcal{F}_j\left(t_{skew}^{su,max}(j), t_{skew}^{h,max}(j)\right). \tag{14b}$$

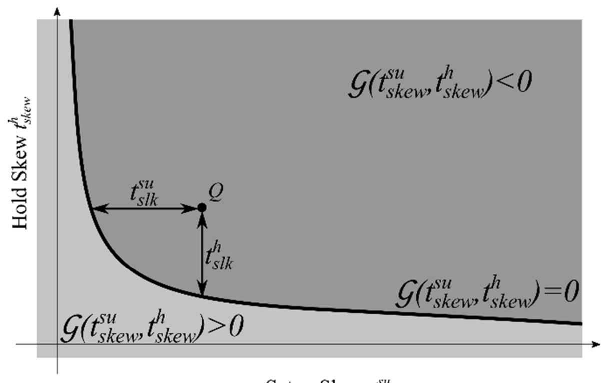


Fig. 6. Description of the boundary curve $\mathcal{G}(t_{skew}^{su}, t_{skew}^{h}) = 0$. The upper right part, represented by $\mathcal{G}(t_{skew}^{su}, t_{skew}^{h}) < 0$, is safe for setting setup and hold times, while setting them in the lower left region, i.e., $\mathcal{G}(t_{skew}^{su}, t_{skew}^{h}) > 0$, is infeasible due to the failure of a latching. If the minimum setup and hold skews of a flip-flop are located at point Q, the setup and hold slacks of that flip-flop are the distances from Q to the curve $\mathcal{G}(t_{skew}^{su}, t_{skew}^{h}) = 0$ along x and y axes, respectively.

2. Analysis: Finding Minimum Clock Period

The problem in this subsection is to find a minimum clock period at which a given circuit can operate with no timing violations for given clock arrival times. To consider the combination of setup and hold time constraints, let us assume that N flip-flops $FF_{(i,1)}, \ldots, FF_{(i,N)}$ have combinational logic paths to FF_j, as shown in Fig. 7. Each path from $FF_{(i,n)}$ to FF_j $(n = 1, \cdots, N)$ has the minimum setup and hold skews

$$t_{skew}^{su,min}(i,n,j) = \left(T_{clk} + x(j)\right) - \left(x(i,n) + t_{c2q}^{max}(i,n) + D_{(i,n,j)}^{max}\right), \tag{15a}$$

$$t_{skew}^{h,min}(i,n,j) = \left(x(i,n) + t_{c2q}^{min}(i,n) + D_{(i,n,j)}^{min}\right) - x(j), \tag{15b}$$

where $t_{skew}^{su,min}(i,n,j)$ and $t_{skew}^{h,min}(i,n,j)$ are the minimum setup and hold skews of the path from $FF_{(i,n)}$ to FF_j, respectively. The maximum setup and hold skews of each path can also be calculated similarly. Since all paths are independent to each other in static timing analysis, the minimum setup and hold skews of FF_j, i.e., $t_{skew}^{su,min}(j)$ and $t_{skew}^{h,min}(j)$, should be less than all the minimum setup and hold skews, and the maximum setup and hold skews of FF_j, i.e., $t_{skew}^{su,max}(j)$ and $t_{skew}^{h,max}(j)$, should also be greater than all the maximum setup and hold skews; therefore, it can be expressed as

$$t_{skew}^{su,min}(j) \leq \min_{n=1,\cdots,N}\{t_{skew}^{su,min}(i,n,j)\}, \tag{16a}$$

$$t_{skew}^{h,min}(j) \leq \min_{n=1,\cdots,N}\{t_{skew}^{h,min}(i,n,j)\}, \tag{16b}$$

$$t_{skew}^{su,max}(j) \geq \max_{n=1,\cdots,N}\{t_{skew}^{su,max}(i,n,j)\}, \tag{16c}$$

$$t_{skew}^{h,max}(j) \geq \max_{n=1,\cdots,N}\{t_{skew}^{h,max}(i,n,j)\}. \tag{16d}$$

In addition, the following timing analysis constraints,

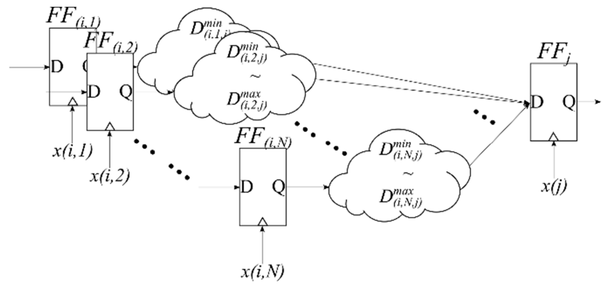


Fig. 7. A circuit illustrating our derivation of formulation for the problem in subsection IV-2. For clarity, only three paths are shown in the figure.

i.e., the interacting constraints of setup and hold times and the clock-to-Q delay constraints, should satisfy

$$\mathcal{G}_j\left(t_{skew}^{su}(j),\, t_{skew}^{h}(j)\right) \le 0, \tag{17a}$$

$$t_{c2q}^{min}(j) \le \mathcal{F}j\left(t_{skew}^{su}(j),\, t_{skew}^{h}(j)\right) \le t_{c2q}^{max}(j) \tag{17b}$$

for each flip-flop FF_j.

At the same time, it is necessary to check if the solution space constrained by the inequalities with functions $\mathcal{G}$ and $\mathcal{F}$ in Eq. (17) is convex for reliable and efficient exploration. The following three assumptions support the convexity:

- *Assumption 1*. $\mathcal{G}$ for every flip-flop is a convex and monotonic non-increasing function.
- *Assumption 2*. $\mathcal{F}$ for every flip-flop is a convex and monotonic non-increasing function.
- *Assumption 3*. The lower bound of clock-to-Q delay for every flip-flop equals to the minimum possible clock-to-Q delay of the flip-flop.

Assumption 1 is acceptable since $\mathcal{G}$ is defined by a designer. *Assumption 2* is also strongly supported in the light of the shape of a clock-to-Q delay surface, i.e., Fig. 3, and by the qualitative analysis of the internal operation of flip-flops. Finally, *Assumption 3* does also make sense in that clock-to-Q delay decreases very rapidly as setup and hold skews increase, ultimately approaching the minimum clock-to-Q delay, as shown in Fig. 3.

As a result, the problem can be transformed into a form of convex programming expressed as

$$\text{minimize } T_{clk} \tag{18}$$

$$\begin{aligned}\text{subject to } & t_{skew}^{su,min}(j) \le \left(T_{clk} + x(j)\right) \\ & -\left(x(i) + t_{c2q}^{max}(i) + D_{i,j}^{max}\right) \\ & \forall \text{ flip-flop pairs } (i,j), \\ & \mathcal{F}_j\left(t_{skew}^{su,min}(j),\, t_{skew}^{h,min}(j)\right) \le t_{c2q}^{max}(j) \forall FF_j, \\ & \mathcal{G}_j\left(t_{skew}^{su,min}(j),\, t_{skew}^{h,min}(j)\right) \le 0\ \forall FF_j.\end{aligned}$$

Here T_{clk}, $t_{skew}^{su,min}$, and t_{c2q}^{max} are optimization variables while the others are constants obtained from analyzing the target circuit and characterizing flip-flops in the optimization problem of Eq. (18). Note that $t_{skew}^{h,min}$ can be computed using clock arrival times already known as *Assumption 3* before solving Eq. (18); therefore, we can exclude all optimization variables related to $t_{skew}^{h,min}$ and t_{c2q}^{min}.

3. Optimization: Clock Skew Scheduling

Clock skew scheduling is another representative timing optimization problem which exploits the adjustment of clock arrival times of flip-flops in order to relax the worst and total slacks or increase the clock frequency of a circuit. Relaxation of setup and hold slacks by exploiting clock skew scheduling, in particular, is important for a number of reasons [14, 15]. One reason is that it could resolve setup and hold time violations with little effort to remove them using late-state ECO knobs, e.g., trial-and-error of gate sizing, threshold voltage swapping, etc. Another reason is that it enables a circuit to be highly tolerant to process, voltage, and temperature variations which are frequently occurred in recent nano-scale high speed designs.

In spite of its importance, there are few previous works handled the integration of clock skew scheduling and flexible flip-flop timing model, and among them, to the best of our knowledge, no work has considered the intervals of clock-to-Q delays of flip-flops yet. We formulate the problem in a form of convex programming with the three assumptions presented in subsection IV-2. Basic constraints are identical to those in Eq. (18), but we require additional constraints for expressing slacks and adjustment ranges of clock arrival times.

(1) Formulating worst and total slack constraints: Worst slack is the minimum value among all setup and hold slacks, thus it can simply be formulated as

$$t_{slk}^{w} \le t_{slk}^{su}(j) = t_{skew}^{su,min}(j) - t_{time}^{su}(j)\ \forall FF_j \tag{19a}$$

$$t_{slk}^{w} \le t_{slk}^{h}(j) = t_{skew}^{h,min}(j) - t_{time}^{h}(j)\ \ \forall FF_j \tag{19b}$$

Eq. (19) includes $t_{time}^{su}(j)$ and $t_{time}^{h}(j)$, which can be easily calculated from $t_{skew}^{h}(j)$ and $t_{skew}^{su}(j)$, respectively. For an example of the nonlinear analytical model in [12], $\mathcal{G}_j\left(t_{skew}^{su}(j),\, t_{skew}^{h}(j)\right) = a_0 + a_1/(t_{skew}^{su} - s_0) + a_2/(t_{skew}^{h} - h_0) - t_{c2q}^{upr}(j)$ if $\mathcal{G}_j$ is assumed to be a constant clock-to-Q delay curve; thus $t_{time}^{su}(j)$ and $t_{time}^{h}(j)$ corresponding to $t_{skew}^{su,min}(j)$ and $t_{skew}^{h,min}(j)$ are obtained by

$$t_{time}^{su}(j) = \frac{a_1}{t_{c2q}^{upr}(j) - a_0 - \frac{a_2}{t_{skew}^{h,min}(j) - h_0}} + s_0, \tag{20a}$$

$$t_{time}^{h}(j) = \frac{a_2}{t_{c2q}^{upr}(j) - a_0 - \frac{a_1}{t_{skew}^{su,min}(j) - s_0}} + h_0. \tag{20b}$$

Total slack is the summation of minimum slack values all flip-flops have, and the constraint is formulated as

$$t_{slk}^{tot} \leq \sum_{FF_j} \min\{t_{slk}^{su}(j), t_{slk}^{h}(j)\}, \tag{21}$$

where $t_{slk}^{su}(j)$ and $t_{slk}^{h}(j)$ are that same as that in Eq. (19).

For formulating convex programming, the solution space constrained by all constrains should be a convex set. In the case of the constraints of Eq. (19), $t_{time}^{su}(j)$ and $t_{time}^{h}(j)$ are convex functions of $t_{skew}^{h,min}(j)$ and $t_{skew}^{su,min}(j)$, respectively, due to *Assumption 2* and the property that $f: \mathbb{R}^n \to \mathbb{R}$ is convex if and only if the function $g: \mathbb{R} \to \mathbb{R}$, $g(t) = f(x + tv)$, $\operatorname{dom} g = \{t | x + tv \in \operatorname{dom} f\}$ is convex in t for any $x \in \operatorname{dom} f$, $v \in \mathbb{R}^n$. Hence, the right hand sides of Eq. (19) are concave, and consequently, the solution space is a convex set. For the case of a total slack constraint, the right hand side of Eq. (21) is also concave since minimum of concave functions is concave and summation operator preserves its convexity or concavity; thus the solution space is again a convex set. It should be noted that we do not need to care about the clock-to-Q delay surface model for Eq. (19) as long as the boundary function $\mathcal{G}_j$ and itself are convex.

(2) Formulating boundaries of clock arrival times: Clock signal arrival times cannot be increased or decreased unlimitedly because of the limitation of the amount of available resources, e.g., metal wires, buffers, inverters, etc. The boundary of each clock arrival time x thus should be expressed as

$$\gamma_1(j) \leq x(j) \leq \gamma_2(j) \;\; \forall FF_j \tag{22}$$

where $\gamma_1(j)$ and $\gamma_2(j)$ are the lower and the upper bounds of the adjustment amount of clock arrival time of FF_j set by a designer.

In summary, clock skew scheduling problem for maximizing worst slack based on flexible flip-flop timing model and our clock-to-Q delay interval analysis is expressed as

$$\text{maximize } t_{slk}^{w} \tag{23}$$

$$\begin{aligned}
\text{subject to } & t_{skew}^{su,min}(j) \leq (T_{clk} + x(j)) \\
& -(x(i) + t_{c2q}^{max}(i) + D_{i,j}^{max}) \\
& \forall \text{ flip-flop pairs } (i,j), \\
& t_{skew}^{h,min}(j) \leq x(i) + t_{c2q}^{min}(i) + D_{i,j}^{min} - x(j) \\
& \forall \text{ flip-flop pairs } (i,j), \\
& \mathcal{F}_j\left(t_{skew}^{su,min}(j), t_{skew}^{h,min}(j)\right) \leq t_{c2q}^{max}(j) \;\; \forall FF_j, \\
& t_{slk}^{w} \leq t_{skew}^{su,min}(j) - \mathcal{H}_j^{su}\left(t_{skew}^{h,min}(j)\right) \;\; \forall FF_j, \\
& t_{slk}^{w} \leq t_{skew}^{h,min}(j) - \mathcal{H}_j^{h}\left(t_{skew}^{su,min}(j)\right) \;\; \forall FF_j, \\
& \gamma_1(j) \leq x(j) \leq \gamma_2(j) \;\; \forall FF_j, \\
& t_{slk}^{w} \geq 0,
\end{aligned}$$

where $\mathcal{H}_j^{su}$ and $\mathcal{H}_j^{h}$ are equations of $t_{time}^{su}(j)$ and $t_{time}^{h}(j)$ derived from Eq. (20), and optimization variables in Eq. (23) are t_{slk}^{w}, $t_{skew}^{su,min}$, $t_{skew}^{h,min}$, t_{c2q}^{max}, and x. Note that we can omit interacting constraints of setup and hold times in this formulation since the forth and the fifth constraints can replace them perfectly with the last constraint. All t_{c2q}^{min} are also vanished due to Assumption 3. In addition, Eq. (23) can be exploited for minimizing clock period by substituting T_{clk} for the objective function and including it in the optimization variables; furthermore, total slack maximization is also possible by replacing the objective with the right hand side of Eq. (21).

4. Scalable Speedup Technique

To solve Eq. (18) and Eq. (23) efficiently, we can exploit various algorithms, e.g., the interior-point polynomial time method [21], which is known as one of the most efficient techniques in theory and practice to solve many convex optimization problems including linear programming, second order cone programming, semidefinite programming, etc. However, as a circuit size grows and the number of flip-flops explodes, the interior-point method will suffer from the run time issue as well as the capacity problem due to the drastic increase of the number of variables and constraints in the formulation. For these reasons, it is required to develop an alternative viable speedup technique that is best suited for the context of our solution with negligible decrease of analysis quality.

It is trivial that all timing paths cannot be critical simultaneously, and not surprisingly, a large part of information on the timing paths is redundant and can be excluded from our problem formulation. Therefore, we can exploit this property as a means of speeding up. In some aspect, our speedup technique is similar to that in [16] which is a sort of criticality-dependency aware timing analysis. The work in [16] collected timing-risky flip-flops and checked the criticality for all fan-out flip-flops by applying breadth first search traversal. We update the technique in [16] as Algorithm 1 based on our observation.

Algorithm 1 shows our proposed algorithm for speeding up of solving clock period minimization problem Eq. (18), which classifies the types of all flip-flops and data paths. Meaning of each flip-flop's type is as follows:

- $TYPE_N$-0: It means that there is no need to include the flexible timing property of the flip-flop in Eq. (18). The maximum clock-to-Q delay of that flip-flop will be set as $\alpha \cdot t_{c2q}^{min}$ for reduction of the problem size.
- $TYPE_N$-1: It means that the flexible timing property of the flip-flop should be fully or partially considered in Eq. (18).

The algorithm required two additional input parameters: a set of maximum allowable clock-to-Q delays t_{c2q}^{upr}, which is related to interacting constraints of setup and hold times, and a user-defined parameter α. We first assume the clock period of the circuit as the minimum possible value T_{clk}^{l}. After that, we calculate $t_{skew}^{h,min}$ for all flip-flops and compute $t_{skew}^{su,cor}$ using them. Note that the minimum hold skew for each flip-flop is independent to the clock period, and $\mathcal{H}$ is the function which finds the corresponding setup skew for a given hold skew and a target clock-to-Q delay on a clock-to-Q delay surface. We then calculate $t_{skew}^{su,wst}$ assuming that the maximum clock-to-Q delays of all precedent flip-flops are equal to $\alpha \cdot t_{c2q}^{min}$. If $t_{skew}^{su,wst} \geq t_{skew}^{su,cor}$, the minimum hold skew of the flip-flop is large enough, so that there is no need to involve its flexible flip-flop timing property. At last, we propagate the effects of $TYPE_N$-1 flip-flops in the way of work list algorithm using the queue Q.

Connections around a $TYPE_N$-1 flip-flop can be classified into four types by the types of their fan-in and

Algorithm 1. Preprocessing for speeding up of Eq. (18)

```
input   Timing information of a given circuit
        Clock-to-Q delay surface model
        Maximum allowable clock-to-Q delays $t_{c2q}^{upr}$
        User-defined parameter $\alpha$
output  Classification results for all flip-flops
/* initialization */
$T_{clk}^{l} \leftarrow \max_{(i,j)}\{s_0(j) + (x(i) + t_{c2q}^{min}(i) + D_{i,j}^{max}) - x(j)\}$
initialize an empty queue $Q$
/* rough estimation */
for each flip-flop $FF_j$ do
    $t_{skew}^{h,min}(j) \leftarrow \min_{FF_i \in P_j}\{x(i) + t_{c2q}^{min}(i) + D_{i,j}^{min}\} - x(j)$
    $t_{skew}^{su,cor}(j) \leftarrow \mathcal{H}\left(t_{skew}^{h,min}(j), \alpha \times t_{c2q}^{min}(j)\right)$
    $t_{skew}^{su,wst}(j) \leftarrow \left(T_{clk}^{l} + x(j)\right)$
            $- \max_{FF_i \in P_j}\{x(i) + \alpha \times t_{c2q}^{min}(i) + D_{i,j}^{max}\}$
    if $t_{skew}^{su,wst}(j) < t_{skew}^{su,cor}(j)$ then
        $type_N(j), t_{c2q}^{max}(j) \leftarrow TYPE_N\text{-}1, t_{c2q}^{upr}(j)$
        $Q.enqueue(j)$
    else
        $type_N(j), t_{c2q}^{max}(j) \leftarrow TYPE_N\text{-}0, \alpha \times t_{c2q}^{min}$
    end
end
/* work list algorithm */
while $Q.empty() = False$ do
    $i \leftarrow Q.dequeue()$
    for each flip-flop $FF_j$ driven by $FF_i$ do
        if $type_N(j) = TYPE_N\text{-}0$ then
            $t_{skew}^{su,wst}(j) \leftarrow \min\left\{t_{skew}^{su,wst}(j), \left(T_{clk}^{l} + x(j)\right) -\right.$
                $\left.\left(x(i) + t_{c2q}^{max}(i) + D_{i,j}^{max}\right)\right\}$
            if $t_{skew}^{su,wst}(j) < t_{skew}^{su,cor}(j)$ then
                $type_N(j), t_{c2q}^{max}(j) \leftarrow TYPE_N\text{-}1, t_{c2q}^{upr}(j)$
                $Q.enqueue(j)$
            end
        end
    end
end
return $type_N$
```

fan-out flip-flops. When it is connected to no $TYPE_N$-1 fan-out flip-flops, it is separated from other flip-flops, and there is no need to include its t_{c2q}^{max}. Hence it is enough to consider only the constraints related to the minimum setup skew and the interacting constraints of the setup and hold times for it. For other cases, it is not guaranteed that t_{c2q}^{max} values of all $TYPE_N$-1 fan-out flip-flops are less than or equal to $\alpha \cdot t_{c2q}^{min}$, and hence we need to reserve them as optimization variables in Eq. (18). Remaining flip-flops except $TYPE_N$-1 flip-flops, i.e., $TYPE_N$-0 flip-flops, can be excluded from solving Eq. (18), and as a result, we could expect the reduction of the number of variables and constraints and total run time.

Algorithm 1 can also be extended to the clock skew scheduling problem with slight modification in the same token. In the case of Eq. (23), unlike Eq. (18), x values are included in optimization variables, and therefore, we cannot obtain the exact minimum hold skew for each flip-flop and should handle it as an interval. For example, in the circuit in Fig. 7, we define n_1 and n_2 as

$$n_1 \coloneqq \text{argmax}_n\{\gamma_2(i,n) + \alpha \cdot t_{c2q}^{min}(i,n) + D_{(i,n,j)}^{max}\}, \tag{24a}$$

$$n_2 \coloneqq \text{argmin}_n\{\gamma_1(i,n) + t_{c2q}^{min}(i,n) + D_{(i,n,j)}^{min}\}, \tag{24b}$$

in order to analyze and use the maximum clock-to-Q delay of each flip-flop. Then the candidate minimum setup and hold skews of FF_j when $x(j) = 0$ can be expressed as a rectangle whose width and height are $\gamma_2(i,n_1) - \gamma_1(i,n_1)$ and $\gamma_2(i,n_2) - \gamma_1(i,n_2)$, respectively. Since an increment of $x(j)$ causes an increase of $t_{skew}^{su}(j)$ and a decrease of $t_{skew}^{h}(j)$ of the center point of the rectangle simultaneously, the region would shift if $x(j)$ varies from $\gamma_1(j)$ to $\gamma_2(j)$. Consequently, all candidate pairs of $t_{skew}^{su,min}(j)$ and $t_{skew}^{h,min}(j)$ can be represented as a hexagonal region, and it is enough to check clock-to-Q delays at two vertices of the region due to the convexity of a clock-to-Q delay surface. In Algorithm 1, we can use this concept instead of $t_{skew}^{su,cor}$ and $t_{skew}^{su,wst}$ for classifying the types of all flip-flops.

V. Experimental Results

All the experiments were run in MATLAB environments, and our flexible flip-flop timing model based formulations were solved by using CVX, a package for specifying and solving convex programs [22, 23], with SDPT3 4.0, a MATLAB software for semidefinite-quadratic-linear programming [24, 25]. All implementations were performed on Linux machine with 8 cores of 3.50 GHz CPU and 16 GB memory and targeted on ISCAS'89 benchmark circuits, b19 of ITC'99 benchmark circuits, and des (perf_opt), pci, usb_funct, and vga_lcd of OpenCores benchmark circuits. 45 nm NanGate Open Cell Library [2] and SPICE simulations using Synopsys HSPICE were used to sample clock-to-Q delays. Synopsys Design Compiler, IC Compiler, and PrimeTime were used for synthesizing, placement & routing, and extracting timing information of the circuits, respectively.

1. Analysis: Finding Minimum Clock Period

To compare the results of the conventional flexible flip-flop timing model based clock period minimization methods proposed in [12] and our clock-to-Q delay interval based analysis (without speedup technique) fairly, we implemented them in MATLAB environments. In addition, we assumed all clock arrival times had already been scheduled under the conventional fixed flip-flop timing model based method [26], since useful clock skews are frequently utilized in circuit design flow for the purpose of optimization. The results are shown in Table 1.

Table 1 summarizes minimum clock periods obtained by ITA (proposed method in [12]) and ours with run times. The first and second columns, Ckt. and #FFs, represent the name and the number of flip-flops of each benchmark circuit. For some of benchmark circuits, there were discrepancies of minimum clock periods T_{clk}^{min} between ITA and ours. For example, ITA determined that the minimum clock period of s38417 is 2130.43 ps; however, our solution reported that it could be reduced further to 2121.32 ps. As it can be seen from Table 1, all minimum clock periods were reduced by ours. In terms of run time, our method took a little more time than ITA for small circuits, e.g., s27, s386, s420, s641, etc.; on the other hand, for large circuits, e.g., s35932, s38584, b19, pci, etc., ours took less time due to its stability nature originated from convexity. Note that M/E in the table represents out-of-memory error.

2. Optimization: Clock Skew Scheduling

Experiments of clock skew scheduling for worst slack maximization were started with initializing reference clock periods using conventional fixed flip-flop timing model based clock skew scheduling scheme [26]. For each benchmark circuit, we compared worst slack values under the reference clock period obtained by (1) applying clock skew scheduling utilizing flexible flip-flop timing (CSS-FT) proposed in [15] to the result of [26]

Table 1. Comparison of minimum clock periods and run times assuming clock arrival times have already been scheduled under the conventional fixed flip-flop timing model [26]

Ckt.	#FFs	run time (s)		T_{slk}^{min} (ps)		
		ITA [12]	Ours	ITA [12]	Ours	Impr.
s27	3	0.02	0.07	439.75	439.00	0.75
s298	14	0.05	0.08	905.16	904.09	1.08
s344	15	0.06	0.09	909.43	908.37	1.06
s349	15	0.06	0.09	913.94	912.93	1.01
s382	21	0.08	0.09	881.94	880.80	1.14
s386	6	0.01	0.09	903.01	902.96	0.06
s400	21	0.18	0.12	930.18	929.19	0.99
s420	16	0.05	0.09	941.68	938.73	2.95
s444	21	0.08	0.09	919.16	918.27	0.89
s510	6	0.03	0.08	1041.59	1040.46	1.13
s526	21	0.09	0.09	944.33	943.38	0.95
s641	14	0.02	0.09	3259.33	3258.76	0.57
s713	14	0.03	0.09	3456.06	3455.48	0.58
s820	5	0.01	0.08	1316.92	1316.88	0.04
s832	5	0.02	0.07	1246.86	1245.90	0.95
s838	32	0.17	0.10	1029.96	1028.17	1.79
s953	29	0.07	0.10	1197.13	1197.01	0.13
s1196	18	0.02	0.08	1284.22	1283.61	0.61
s1238	18	0.02	0.08	1228.26	1227.51	0.75
s1423	74	0.36	0.17	2983.67	2974.80	8.86
s1488	6	0.01	0.08	1279.83	1279.68	0.14
s5378	162	0.30	0.22	1190.28	1190.20	0.08
s9234	132	0.28	0.22	2439.80	2430.73	9.07
s13207	214	0.38	0.23	2498.22	2489.42	8.80
s15850	128	0.34	0.20	2845.97	2837.20	8.77
s35932	1728	2.25	0.96	1502.32	1501.83	0.48
s38417	1462	2.13	1.60	2130.43	2121.32	9.11
s38584	1159	2.05	0.80	2388.52	2379.63	8.88
b19	1876	6.05	2.01	5999.09	5989.90	0.02
des	1984	2.46	1.29	2293.53	2293.50	0.02
pci	3271	4.16	2.25	2400.64	2400.04	0.60
usb_funct	1735	0.46	0.72	1297.53	1296.86	0.67
vga_lcd	17055	69.47	M/E	3743.12	M/E	N/A

additionally, (2) solving our flexible flip-flop timing model based convex problems described in subsection IV-3 with CVX directly. All clock arrival times were assumed to be in the range from -150 ps to +150 ps. We also applied the same definitions of setup and hold slacks used in [15] and subsection IV-3 for calculating them. Results of the experiments are shown in columns 3-5 of Table 2. Columns 3 and 4 of the table show worst slacks of CSS-FT and ours, respectively. As shown in the table, all worst slacks obtained by our method were greater than or equal to those obtained by CSS-FT since ours considered it globally while CSS-FT did it locally. Compared with the results of CSS-FT for benchmark circuits, e.g., s386, s838, s953, s1488, etc., ours ameliorated worst slacks by 10 ~30 ps. In the case of vga_lcd, our method and t_{slk}^{w} estimation process could not be performed due to out-of-memory errors originated from the size of the circuit.

The worst slack value of a circuit linked directly to the minimum clock period of it. Basically, a minimum clock period means that there are no timing violations at all in the circuit under the timing condition, i.e. $t_{slk}^{w} \geq 0$; therefore, comparing minimum clock periods obtained by changing clock arrival times can also be one criterion for assessing the performance of clock skew scheduling, and we proceeded the experiments additionally. Since [12] cannot modify clock arrival times and [15] only works under given clock period with initial clock arrival times, both of them cannot be compared with ours directly; thus we estimated the differences between minimum clock periods obtained by fixed flip-flop timing model based clock skew scheduling scheme [26] and ours. Results are shown in columns 6-8 of Table 2. The table shows that our flexible flip-flop timing model based clock skew scheduling reduced minimum clock periods by up to 75 ps over fixed flip-flop timing based approach.

Columns 9-11 of Table 2 summarizes the total slacks of CSS-FT and our approach. For all benchmark circuits, ours improved total slacks by up to 56 ns over those of CSS-FT. While our methods are effective in comparison with other conventional methods, complexity or run times of them would be one of the most critical issues. For example, CSS-FT with conventional fixed flip-flop timing model based clock skew scheduling took 0.87+3.10=3.97 seconds for relaxing timing constraints of pci in Table 3, but ours took 21.85 seconds for resolving them, without any speedup techniques, which is about 5.5x slower. In addition, for all kinds of experiments, proposed methods could not handle vga_lcd due to its size. Thus it is needed to refine our proposed solution to overcome them by exploiting scalable speedup technique.

3. Scalable Speedup Technique

For verifying the effectiveness of our speedup technique presented in subsection IV-4, we compared analysis qualities and run times obtained with and without the technique. We set the same experimental environments used in subsections V-1 and V-2. For speeding up of finding solutions, we set $\alpha = 1.05$, i.e., we excluded some variables and constraints of the flip-flops whose maximum clock-to-Q delay is less than $1.05 \times t_{c2q}^{min}$ from our formulations. Results are shown in Fig. 8.

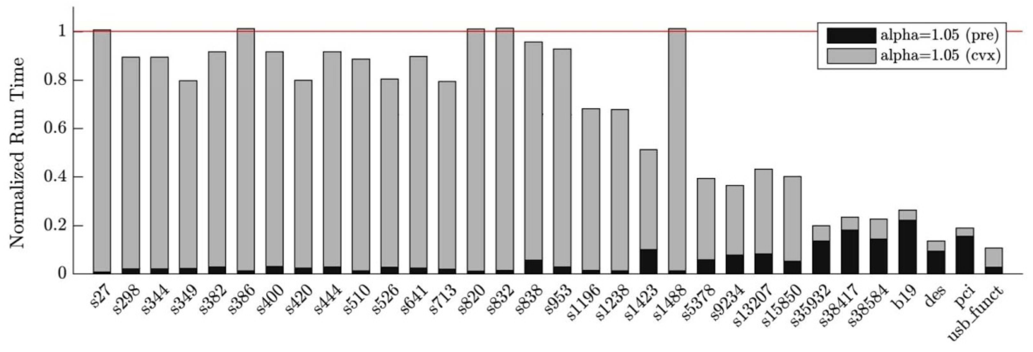
Table 2. Comparison of worst slacks, minimum clock periods, and total slacks obtained by previous researches and ours

Ckt.	#FFs	t_{slk}^{w} (ps)			T_{clk}^{min} (ps)			t_{slk}^{tot} (ps)		
		CSS-FT [15]	Ours	Impr.	CSS [26]	Ours	Impr.	CSS-FT [15]	Ours	Impr.
s27	3	73.26	74.34	1.08	503.28	438.96	64.32	382.74	891.44	8.80
s298	14	75.00	75.11	0.11	969.01	903.97	65.04	3156.44	3873.69	717.25
s344	15	75.05	75.05	0.00	973.75	908.37	64.99	3309.81	3748.24	438.42
s349	15	74.97	75.16	0.19	977.87	912.79	65.08	3161.56	3700.39	538.83
s382	21	75.13	75.18	0.04	945.86	880.75	65.10	3032.47	3679.62	647.15
s386	6	19.11	36.71	17.60	922.05	872.85	49.21	563.70	795.68	231.98
s400	21	74.93	75.11	0.18	994.12	929.07	65.04	3619.72	4590.68	970.96
s420	16	75.01	75.10	0.08	1005.68	938.21	67.47	2717.76	3238.24	520.48
s444	21	74.67	75.10	0.43	982.87	917.83	65.04	3589.79	4473.75	883.96
s510	6	75.12	75.12	0.00	1105.51	1040.46	65.05	1040.57	1171.39	130.82
s526	21	74.57	75.09	0.52	1008.08	943.05	65.03	3788.39	4970.82	1182.43
s641	14	16.03	17.48	1.45	3728.24	3245.27	32.97	3810.11	4660.41	850.30
s713	14	16.03	17.48	1.45	3474.97	3442.00	39.55	3422.62	4257.65	835.03
s820	5	16.08	23.39	7.31	1335.84	1296.29	39.55	291.45	458.45	166.00
s832	5	74.46	74.78	0.32	1310.48	1245.77	64.71	557.54	646.78	89.24
s838	32	39.70	66.23	26.53	1093.90	1026.87	67.03	4757.35	5937.16	1179.81
s953	29	34.75	56.12	21.37	1231.80	1166.78	65.92	4723.87	5780.30	1056.43
s1196	18	6.48	17.48	11.00	1303.13	1270.16	32.98	4076.08	5208.91	1132.84
s1238	18	6.50	17.49	10.99	1247.05	1214.06	32.99	3781.89	5038.83	1256.94
s1423	74	75.50	75.51	0.00	3050.64	2975.14	75.50	21544.53	23745.83	2201.30
s1488	6	34.84	55.63	20.79	1314.58	1249.77	64.80	541.63	668.43	126.79
s5378	162	37.27	44.19	6.91	1209.19	1167.32	41.87	36732.06	47125.46	10393.40
s9234	132	75.48	75.54	0.06	2506.72	2431.19	75.53	35781.64	41272.52	5490.88
s13207	214	75.19	75.22	0.04	2565.16	2489.94	75.22	56039.09	64436.94	8397.85
s15850	128	73.93	75.23	1.30	2912.96	2837.72	75.23	31611.01	34006.96	2395.95
s35932	1728	18.90	18.90	0.00	1521.31	1502.31	19.00	559817.35	615888.62	56071.27
s38417	1462	75.60	75.61	0.01	2197.36	2121.75	75.61	391052.57	423355.40	32302.84
s38584	1159	75.31	75.49	0.18	2455.41	2379.93	75.49	334020.99	366855.19	32834.19
b19	1876	75.52	75.52	0.00	6065.95	5990.43	75.52	708851.21	736437.16	27585.95
des	1984	18.99	19.00	0.00	2312.49	2293.49	19.00	400060.43	440894.99	40834.56
pci	3271	19.00	19.00	0.00	2419.56	2400.56	19.00	410575.41	441053.41	30477.99
usb_funct	1735	5.44	17.47	11.03	1316.39	1283.41	32.98	29011.55	38375.92	9365.36
vga_lcd	17055	N/A	M/E	N/A	3762.12	M/E	N/A	N/A	M/E	N/A

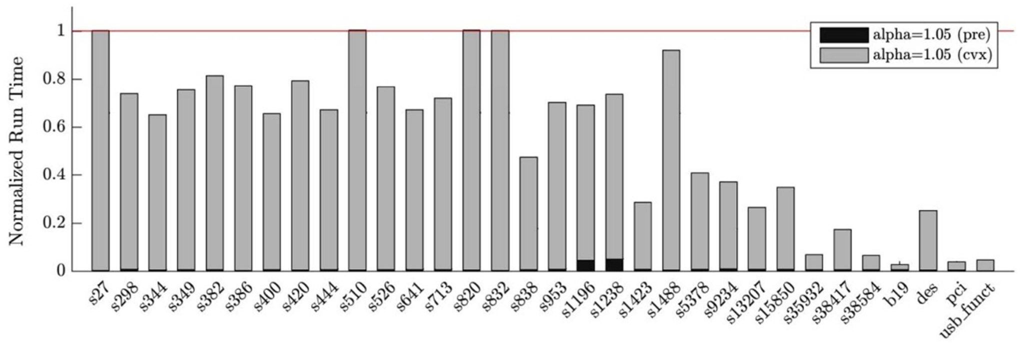
Table 3. Comparison of run times of CSS [26], CSS-FT [15], and ours for maximizing worst slack for some benchmark circuits. Note that run times of CSS-FT denote additional processing times only

Ckt.	#FFs	average run time (s)		
		CSS [26]	CSS-FT [15]	Ours
s27	3	0.05	0.02	0.10
s349	15	0.06	0.04	0.17
s386	6	0.05	0.05	0.12
s526	21	0.06	0.04	0.19
s713	14	0.08	0.07	0.15
s832	5	0.05	0.04	0.12
s838	32	0.08	0.13	0.30
s953	29	0.06	0.06	0.23
s1238	18	0.07	0.04	0.15
s1423	74	0.11	0.22	0.61
s1488	6	0.05	0.02	0.11
s5378	162	0.11	0.20	0.72
s9234	132	0.09	0.21	0.77
s13207	214	0.12	0.29	0.96
s15850	128	0.07	0.16	0.69
s35932	1728	0.19	1.48	7.36
s38417	1462	0.65	2.09	11.80
s38584	1159	0.31	1.20	6.99
b19	1876	0.86	3.27	14.81
des	1984	0.44	1.35	12.08
pci	3271	0.87	3.10	21.85
usb_funct	1735	0.10	0.43	6.36

Fig. 8 shows the normalized run times of our speeding up solutions for the problem of finding minimum clock period and the clock skew scheduling problem for worst slack maximization, respectively. All run times are normalized by the run times of our original solutions; therefore, the value of 0.10 of usb_funct in Fig. 8(a), for example, means that we reduced the run time by 90%. As it can be seen from the figure, run times were reduced by 33% and 44% on average. Run times for small circuits like s27 and s832 were slightly increased due to the preprocessing step; on the other hand, for large size circuits, they were especially reduced considerably since the number of critical flip-flops and data paths were not proportional to the size of a given circuit. For instance, they were reduced by 74% and 97% for analyzing and optimizing b19, respectively. Furthermore, the problems of vga_lcd were solved within 4.81 seconds and 12.22 seconds, which are 14.44× and 5.11× faster than those of ITA and CSS-FT, respectively, with no out-of-memory errors. Meanwhile, the averages of the differences of the



(a) Comparison of run times of finding minimum clock periods.



(b) Comparison of run times of clock skew scheduling for maximizing worst slack with clock arrival times in the range from -50 ps to 50 ps

Fig. 8. Effectiveness of our speedup technique for the problems. $\alpha = 1.05$ is used for speeding up and all run times are normalized by run times of ours without the speedup technique. *pre* and *cvx* in the charts represent preprocessing and solving with convex optimization solver, respectively, and target clock periods for (b) was obtained by conventional fixed flip-flop timing model based clock skew scheduling scheme [26].

results obtained with and without proposed speedup technique were only 1.20 ps and 1.40 ps, respectively. Therefore, it can be concluded that our technique can efficiently remove unnecessary variables and constraints from our original formulations.

V. CONCLUSION

In this work, we proposed our clock-to-Q delay interval analysis and developed its mathematical formulation for applicability to representative timing analysis and optimization problems, i.e., finding minimum clock period and clock skew scheduling, with the scalable speedup technique. Experimental results with benchmark circuits demonstrated improvement of optimization quality in terms of clock period, worst slack, and total slack, as well as discrepancies between results from previous works and our proposed concepts. Furthermore, it was shown that our speedup technique can significantly shorten run times of solving the problems with very little loss of optimality.

ACKNOWLEDGMENTS

This research was supported by NRF grant funded by the MSIP (2015R1A2A2A01004178) Korea, and Brain Korea 21 Plus Project in 2017.

REFERENCES

[1] J. Heo and T. Kim, “Timing analysis and optimization based on flexible flip-flop timing model,” *VLSI, IEEE Computer Society Annual Symposium on*, pp. 42-46, Jul., 2016.

[2] “Nangate open cell library,” http://www.nangate.com, 2009.

[3] V. Stojanovic and V. G. Oklobdzija, “Comparative analysis of master slave latches and flip-flops for high-performance and low-power systmes,” *Solid-State Circuits, IEEE Journal of*, Vol. 34, No. 4, pp. 536-548, Apr., 1999.

[4] A. M. Jain and D. Blaauw, “Modeling flip flop delay dependencies in timing analysis,” *ACM*

Timing Issues (TAU) Workshops, pp. 67-73, Feb., 2004.

[5] G. Rao and K. Howick, "Apparatus for optimized constraint characterization with degradation options and associated methods," *U. S. Patent No. 6,584,598*, Jun., 2003.

[6] S. Srivastava and J. Roychowdhury, "Interdependent latch setup/hold time characterization via euler-newton curve tracing on state-transition equations," *ACM/IEEE Design Automation Conference*, pp. 136-141, Jun., 2007.

[7] S. Srivastava and J. Roychowdhury, "Independent and interdependent latch setup/hold time characterization via newton-raphson solution and eular curve tracking of state-transition equations," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactiosn on*, Vol. 27, No. 5, pp. 817-830, May, 2008.

[8] E. Salman, E. G. Friedman, A. Dasdan, F. Taraporevala, and K. Kucukcakar, "Pessimism reduction in static timing analysis using interdependent setup and hold times," *Quality Electronic Design, IEEE International Symposium on*, pp. 159-164, Mar., 2006.

[9] E. Salman, A. Dasdan, F. Taraporevala, K. Kucukcakar, and E. G. Friedman, "Exploiting setup-hold-time interdependence in static timing analysis," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, Vol. 26, No. 6, pp. 1114-1125, Jun., 2007.

[10] E. Salman and E. G. Friedman, "Utilizing interdependent timing constraints to enhance robustness in synchronous circuits," *Microelectronics Journal*, Vol. 43, No. 2, pp. 119-127, Feb., 2012.

[11] S. Hatami, H. Abrishami, and M. Pedram, "Statistical timing analysis of flip-flops considering codependent setup and hold times," *VLSI, ACM Great Lakes symposium on*, pp. 101-106, May, 2008.

[12] N. Chen, B. Li, and U. Schlichtmann, "Iterative timing analysis based on nonlinear and interdpendent flipflop modelling," *IET Circuits, Devices & Systems*, Vol. 6, No. 5, pp. 330-337, Sep., 2012.

[13] A. B. Kahng and H. Lee, "Timing margin recovery with flexible flip-flop timing model," *Quality Electronic Design, IEEE International Symposium on*, pp. 496-503, Mar., 2014.

[14] H. Seo, J. Heo, and T. Kim, "Clock skew optimization for maximizing time margin by utilizing flexible flip-flop timing," *Quality Electronic Design, IEEE International Symposium on*, pp. 35-39, Mar., 2015.

[15] H. Seo, J. Heo, and T. Kim, "Post-silicon tuning based on flexible flip-flop timing," *JSTS: Journal of Semiconductor Technology and Science*, Vol. 16, No. 1, pp. 11-22, Feb., 2016.

[16] Y.-M. Yang, K. H. Tam, and I. H.-R. Jiang, "Criticality-dependency-aware timing characterization and analysis," ACM/EDAC/IEEE *Design Automation Conference*, p. 167, Jun., 2015.

[17] O. L. Mangasarian, J. B. Rosen, and M. E. Thompson, "Global minimization via piecewise-linear underestimation," *Global Optimization, Journal of,* Vol. 32, No. 1, pp. 1-9, May, 2005.

[18] A. Magnani and S. P. Boyd, "Convex piecewise-linear fitting," *Optimization and Engineering*, Vol. 10, No. 1, pp. 1-17, Mar., 2009.

[19] E. Lim and P. W. Glynn, "Consistency of multidimensional convex regression," *Operations Research*, Vol. 60, No. 1, pp. 196-208, Feb., 2012.

[20] L. A. Hannah and D. B. Dunson, "Multivariate convex regression with adaptive partitioning," *Machine Learning Research, Journal of*, Vol. 14, No. 1, pp. 3261-3294, Nov., 2013.

[21] Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming*, SIAM, Philadelphia, 1994.

[22] CVX Research Inc., "Cvx: matlab software for disciplined convex programming, version 2.0," http://cvxr.com/cvx, Aug., 2012.

[23] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent advances in learning and control-lecture notes in control and information sciences* (V. Blondel, S. Boyd, and H. Kimura, eds.), pp.95-110, Springer-Verlag London Limited, 2008.

[24] K. C. Toh, M. J. Todd, and R. H. Tutuncu, "Sdpt3 − a matlab software package for semidefinite programming," *Optimization Methods and Software*, Vol. 11, No. 1-4, pp. 545-581, 1999.

[25] R. H. Tutuncu, K. C. Toh, and M. J. Todd, "Solving semidefinite quadratic-linear programs using sdpt3," *Mathematical Programming*, Vol. 95,

No. 2, pp. 189-217, Feb., 2003.

[26] J. P. Fishburn, "Clock skew optimization," *Computers, IEEE Transactions on*, Vol. 39, No. 7, pp. 945-951, Jul., 1990.

Jeongwoo Heo received the B.S. degree in electrical and computer engineering from Seoul National University, Korea, in 2014. He is currently pursuing the Ph.D. degree in electrical and computer engineering at Seoul National University. His current research interests include timing analysis methodology of high performance ICs.

Taewhan Kim received B.S. degree in computer science and statistics and M.S. degree in computer science from Seoul National University, and Ph.D. degree in computer science from University of Illinois at Urbana-Champaign. He is currently a Professor with School of Electrical and Computer Engineering, Seoul National University. He has published over 200 technical papers in international journals and conferences in the area of design methodology of integrated circuits and systems. Dr. Kim is serving on Associate Editor of IEEE Transactions of Computer-Aided Design of Integrated Circuits and Systems, Associate Editor of INTEGRATION – VLSI journal, and Co-Editor-in-Chief of International Journal of Computing Science and Engineering.