Leakage-Power-Aware Clock Period Minimization

Hua-Hsin Yeh[†], Shih-Hsu Huang[†], and Yow-Tyng Nieh[‡]

[†]Department of Electronic Engineering
Chung Yuan Christian University
Chung Li, Taiwan, R.O.C.

[‡]Information and Communications Research Labs
Industrial Technology Research Institute
Hsin Chu, Taiwan, R.O.C.

g9902603@cycu.edu.tw, shhuang@cycu.edu.tw, ytnieh@itri.org.tw

Abstract— In the design of nonzero clock skew circuits, an increase of the path delay may improve circuit speed and reduce leakage power. However, the impact of increasing path delay on the trade-off between circuit speed and leakage power has not been well studied. In this paper, we propose a two-step approach for leakage-power-aware clock period minimization. Compared with previous works, our approach has the following two significant contributions. First, our approach is the first leakage-power-aware clock skew scheduling that can guarantee working with the lower bound of the clock period. Second, our approach is also the first work that demonstrates the problem of minimizing the number of extra buffers is a polynomial-time problem. Benchmark data show that our approach achieves the best results in terms of the clock period and the leakage power.

Keywords—Clock Period Minimization, Leakage Power, Clock Skew Scheduling, Sequential Timing Optimization.

I. INTRODUCTION

By properly scheduling the clock arrival times of registers, the clock period of a nonzero clock skew circuit can be shorter than the longest path delay [1,2]. However, hold constraints often limit the smallest feasible clock period that clock skew scheduling can achieve. In fact, hold violations can be resolved by increasing the path delay [3]. Therefore, in recent years, several research efforts [4-8] have been devoted to study the combination of clock skew scheduling and delay insertion for further clock period reduction. Especially, in [5-8], the lower bound of the clock period can be achieved.

In addition to clock period minimization, low power is also a very important subject. Especially, due to technology scaling, leakage power has become a significant source of power consumption. It is known that gate downsizing (including higher threshold voltage assignment) is one of the most useful techniques for leakage power reduction. Thus, in [9-11], they use gate downsizing to reduce leakage power by slowing down non-critical paths. Moreover, in recent years, several leakage-power-aware clock skew scheduling approaches [12-14], which make use of clock skew scheduling to extend the timing slack of each logic gate for gate downsizing, have been proposed for further leakage power reduction.

From the above discussions, we know that, in the design of nonzero clock skew circuits, an increase of the path delay may improve circuit speed and reduce leakage power. However, to the best of our knowledge, the impact of increasing the path delay on the trade-off between circuit speed and leakage power has not been studied. With an analysis to previous works, we find that previous works can be classified into the following two independent groups.

(1) One group (i.e., previous works [5-8]) uses buffer insertion to implement delay insertion for resolving hold violations. This group can achieve the lower bound of the clock period that the combination of clock skew scheduling and delay insertion can achieve. However, in fact, many hold violations can be resolved by gate downsizing. Since this group only uses buffer insertion to resolve hold violation, this group often suffers from a large overhead on leakage power due to the insertion of extra buffers.
(2) The other group (i.e., previous works [12-14]) applies gate downsizing (including higher threshold voltage assignment) to each logic gate as possible for leakage power reduction. However, since not all hold violations can be resolved by gate downsizing, this group often does not work with the lower bound of the clock period.

Based on those observations, in this paper, we are motivated to study the combination of buffer insertion and gate downsizing during clock skew scheduling for achieving the lower bound of the clock period with the minimum leakage power. Our basic idea is to make use of gate downsizing as possible under the lower bound of the clock period. In other words, only for those hold violations that cannot be resolved by gate downsizing, we apply buffer insertion to them for achieving the lower bound of the clock period. As a result, we can derive a nonzero clock skew circuit that works with the lower bound of the clock period with the minimum leakage power.

Our design methodology includes two steps. At the first step, we use a linear programming (LP) approach to minimize the number of required inserted buffers for achieving the lower bound of the clock period with giving a higher priority to gate downsizing. Next, at the second step, we use a LP approach [12] to apply gate downsizing to all logic gates in the circuit (including those extra buffers inserted at the first step) as possible for minimizing the leakage power. Compared with previous works, experimental results consistently show that our two-step design methodology achieves the best results in terms of circuit speed, leakage power, and total power consumption.

Our approach has the following two significant contributions:

(1) To the best of our knowledge, our approach is the only leakage-power-aware clock skew scheduling that can guarantee working with the lower bound of the clock period.
(2) Different from previous work [8] that uses a mixed integer

linear programming (MILP) approach (i.e., NP-hard approach) to minimize the number of required inserted buffers, we use a LP approach (i.e., polynomial-time approach) to minimize the number of required inserted buffers. To the best of our knowledge, our work provides the first proof of showing that the minimization of the number of required inserted buffers can be solved in polynomial-time complexity. Moreover, it should be mentioned that previous work [8] does not consider gate downsizing. Therefore, in fact, we solve a more complex problem with a smaller time complexity.

II. PRELIMINARIES

A. Clocking Constraints

An edge-triggered circuit consists of registers and logic gates, with wires connecting them. A timing arc is defined as the signal propagation from a wire to the other wire through one logic gate. A data path $R_i \rightarrow R_j$ is defined as the combinational logic from register R_i to register R_j. A timing path is defined as the signal propagation from a register (called *begin port*) to another register (called *end port*). Note that a data path may consist of several timing paths.

Let the notation T_{Ri} denote the designated clock arrival time of register R_i. Let the notation $D_{Ri \rightarrow Rj(max)}$ and the notation $D_{Ri \rightarrow Rj(min)}$ denote the maximum delay and the minimum delay of data path $R_i \rightarrow R_j$, respectively. For each data path $R_i \rightarrow R_j$, there are two types of clocking constraints: the setup constraint corresponds to $T_{Ri}-T_{Rj} \leq P-D_{Ri \rightarrow Rj(max)}$, where P is the clock period, and the hold constraint corresponds to $T_{Rj}-T_{Ri} \leq D_{Ri \rightarrow Rj(min)}$. The minimum-period clock skew scheduling problem [1,2] is to find the smallest feasible clock period and the corresponding clock skew schedule (i.e., the clock arrival times of registers) that satisfies all the clocking constraints. Several graph-based algorithms [2] use the *constraint graph* G(V,E) to solve the minimum-period clock skew scheduling problem in polynomial-time complexity. In a constraint graph, each vertex $R_i \in V$ represents a register. A special vertex, called the *host*, is used for the synchronization of primary inputs and primary outputs. Each directed edge $e \in E$ represents a clocking constraint. For each data path $R_i \rightarrow R_j$, its setup constraint is modeled as an S-edge $e_s(R_j,R_i)$, which is from register R_j to register R_i and associated with a weight $P-D_{Ri \rightarrow Rj(max)}$, and its hold constraint is modeled as an H-edge $e_h(R_i,R_j)$, which is from register R_i to register R_j and associated with a weight $D_{Ri \rightarrow Rj(min)}$. A constraint graph works with clock period P if and only if it the summation of weights in each cycle is not negative when the clock period is P.

Take circuit *ex1* shown in Figure 1 as an example. In Figure 1, each logic gate is associated with a delay value. For example, the delay of logic gate A is 3. The delay of each register is assumed to be 0. Figure 2 gives the corresponding constraint graph. By applying the minimum-period clock skew scheduling, the smallest feasible clock period is 6 and a clock skew schedule in which T_{Host}=0, T_{R1}=1, T_{R2}=2, and T_{R3}=3 is obtained. Figure 3 gives the corresponding constraint graph when the clock period is 6. For the convenience of presentation, in Figure 3, we label T_{Ri} for each vertex R_i.

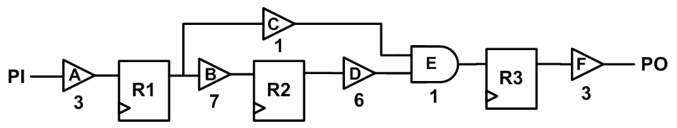

Figure 1: Circuit ex1.

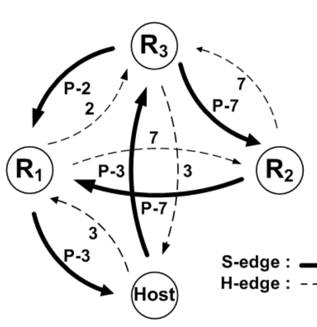

Figure 2: Constraint graph of circuit *ex1*.

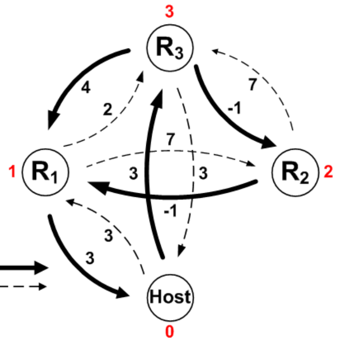

Figure 3. Constraint graph of circuit *ex1* when P = 6.

B. The Lower Bound of The Clock Period

Setup constraints give a lower bound on the clock period obtained by clock skew scheduling [2]. We use the notation P_{set} to denote this lower bound. Take circuit *ex1* shown in Figure 1 as an example. With an analysis to Figure 2, we find that the cycle consisting of $e_s(Host,R_3)$, $e_s(R_3,R_2)$, $e_s(R_2,R_1)$, and $e_s(R_1,Host)$ determines this lower bound, and thus we have $P_{set}(ex1)$ = 5. Due to the limitation of hold constraints, the minimum-period clock skew scheduling often does not achieve this lower bound. Using circuit *ex1* as an example, the minimum-period clock skew scheduling is 6 instead of 5. With an analysis to Figure 3, we find that $e_s(R_3,R_2)$, $e_s(R_2,R_1)$, and $e_h(R_1,R_3)$ form a critical cycle. Thus, if we can insert delay to the data path $R_1 \rightarrow R_3$, the clock period may be further reduced.

We say that the path delay difference of timing path *p* is D(*p*)-d(*p*), where D(*p*) and d(*p*) are the maximum delay and the minimum delay of timing path *p*, respectively. From [5-7], if a clock skew schedule has satisfied all the setup constraints, the largest path delay difference among all the timing paths gives a lower bound of the clock period for inserting delays to resolve all the hold violations without affecting the circuit speed. We use the notation P_{ins} to denote this lower bound. In circuit *ex1*, we have $P_{ins}(ex1)$ = 0.

Finally, we study the lower bound of the clock period for the combination of clock skew scheduling and delay insertion. We use the notation P_{LB} to denote this lower bound. From [5-8], we know P_{LB} = maximum(P_{set},P_{ins}). In circuit *ex1*, we have $P_{LB}(ex1)$ = 5.

C. Leakage Power Minimization

The basic idea of leakage-power-aware clock skew scheduling [12-14] is to leverage on borrowed time for leakage power reduction. Thus, in [12-14], the problem of gate downsizing is regarded as distributing available timing slacks (for delay insertion) to individual logic gates. After that, the assigned timing slack (i.e., the inserted delay) is converted to power saving on each logic gate by identifying the most efficient gate size (including threshold voltage assignment) for that gate. We use the notation C_A to denote the power-delay sensitivity of logic gate A (i.e., the power saving for each unit of the assigned timing slack of logic gate A). If the assigned timing slack of logic gate A is δ_A, then the power saving on logic gate A is $C_A \times \delta_A$. Thus, the objective of leakage-power-aware clock skew scheduling is to maximize $\sum_{\forall A \in GT} C_A \times \delta_A$, where the notation GT denotes the set that includes all logic gates in the circuit. In [12], Ni et al. have used a LP approach to formally formulate the problem of leakage power minimization under clock skew scheduling.

III. OBSEVATION AND MOTIVATION

A. Drawback of Buffer Insertion

For working with the lower bound of the clock period, previous works [5-8] use buffer insertion to resolve hold violations. However, extra buffers cause a large overhead on leakage power. In fact, most hold violations can be resolved by gate downsizing. If we can resolve hold violations by gate downsizing, the leakage power even can be reduced.

Let's use circuit *ex1* to demonstrate our observation. With an analysis to Figure 2, the summation of the weights in the cycle consisting of $e_s(R_3,R_2)$, $e_s(R_2,R_1)$, and $e_h(R_1,R_3)$ is 2P-12. Thus, if the clock period is 5 (i.e., the lower bound of the clock period), to ensure the summation of weights in this cycle is not negative, the increase of the minimum delay of data path $R_1 \rightarrow R_3$ should be at least 2 (in other words, $D_{R1\rightarrow R3(min)}$ should be increased from 2 to at least 4).

Previous works [5-7] focus on the minimization of required inserted delay, but they pay no attention to the minimization of the number of inserted buffers. Therefore, they [5-7] may obtain circuit ex1* as shown in Figure 4 for working with the lower bound of the clock period. We use the notation $<R_1,C>$ to denote the wire from register R_1 to logic gate C. In circuit ex1*, two extra buffers, whose delay values are 1, are inserted into the wire $<R_1,C>$ and the wire $<C,E>$, respectively. Note that these two extra buffers cause an overhead on leakage power.

Previous work [8] uses a MILP approach to minimize the number of inserted buffers for working with the lower bound of the clock period. Thus, previous work [8] can obtain circuit ex1' as shown in Figure 5, in which one extra buffer (whose delay value is 2) is inserted into the wire $<C,E>$. Although the number of required inserted buffer is minimized, this extra buffer still causes an overhead on leakage power.

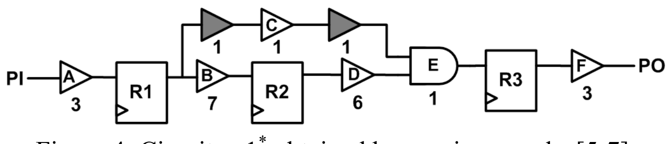

Figure 4: Circuit ex1* obtained by previous works [5-7].

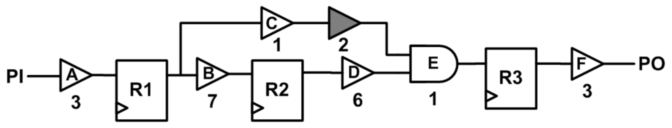

Figure 5: Circuit ex1' obtained by previous work [8].

In fact, in this example, we need not to insert any extra buffer. We can use gate downsizing to increase the delay of logic gate C from 1 to 3. As a result, we not only resolve the hold violation but also reduce the leakage power.

For further reducing the leakage power, the increase of the delay of logic gate C can be larger. Actually, even the delay of logic gate C is increased to be 8, all the clocking constraints are still satisfied. Figure 6 gives circuit ex1" that works with the lower bound of the clock period under the clock skew schedule in which T_{Host}=0, T_{R1}=-2, T_{R2}=0, and T_{R3}=2. In next section, we will present an approach to obtain circuit ex1".

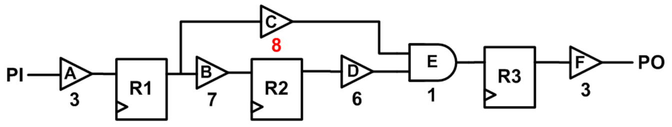

Figure 6: Circuit ex1" obtained by our approach.

B. Limitation of Gate Downsizing

It should be mentioned that not all hold violations can be resolved by gate downsizing. Since previous leakage-power-aware clock skew scheduling approaches [12-14] only use gate downsizing, they often does not work with the lower bound of the clock period. We use circuit *ex2* shown in Figure 7 as an example to demonstrate our observation.

Figure 8 gives the corresponding constraint graph of circuit ex2. From Figure 8, we have $P_{set}(ex2)$ = 5. Since $P_{ins}(ex2)$ = 0, we have $P_{LB}(ex2)$ = $P_{set}(ex2)$ = 5. However, by applying the minimum-period clock skew scheduling, the smallest feasible clock period is 7 instead of 5. Figure 9 gives the corresponding constraint graph when the clock period is 7. With an analysis, we find that $e_h(R_1,R_3)$ and $e_s(R_3,R_1)$ form a critical cycle. To achieve the lower bound of the clock period, we should increase the minimum delay of data path $R_1 \rightarrow R_3$.

In circuit ex2, the timing path $R_1 \rightarrow C \rightarrow R_3$ determines the minimum delay of data path $R_1 \rightarrow R_3$. Note that this timing path has only one logic gate (i.e., logic gate C). With an analysis, we find that we cannot increase the delay of logic gate C for achieving the lower bound of the clock period. The reason is below. Logic gate C is also in the timing path $R_1 \rightarrow B \rightarrow C \rightarrow R_3$ that determines the maximum delay of data path $R_1 \rightarrow R_3$. Since data path $R_1 \rightarrow R_3$ is critical to the setup constraint, we cannot increase the delay of logic gate C. Therefore, in this example, since previous leakage-power-aware clock skew scheduling approaches [12-14] only use gate downsizing, they cannot work with the lower bound of the clock period.

However, in this example, previous work [8] can obtain circuit ex2' as shown in Figure 10 to work with the lower bound of the clock period. In ex2', one extra buffer (whose delay value is 2) is inserted into the wire <R_1,C>.

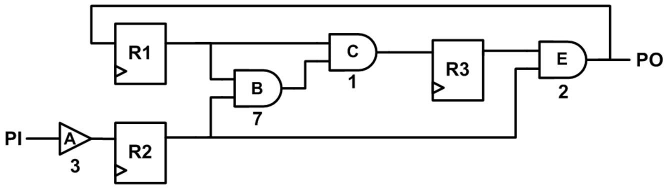

Figure 7: Circuit *ex2*.

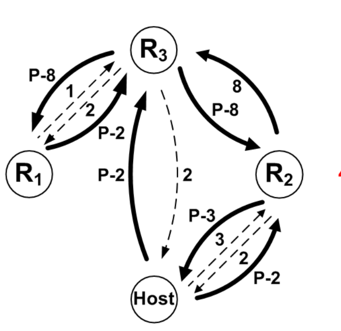

Figure 8: Constraint graph of circuit *ex2*.

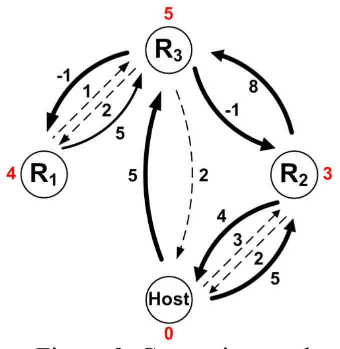

Figure 9. Constraint graph of circuit *ex2* when P = 7.

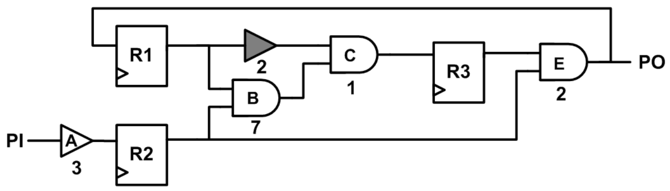

Figure 10: Circuit ex2' obtained by previous work [8].

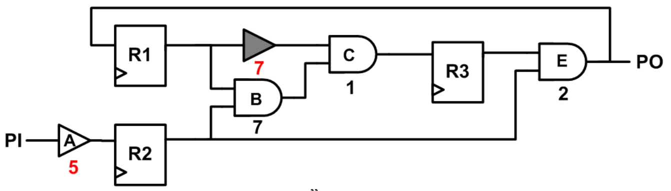

Figure 11: Circuit ex2" obtained by our approach.

For further reducing the leakage power, the delay of this extra buffer can be increased from 2 to 7 and the delay of logic gate A can be increased from 3 to 5. Figure 11 gives circuit ex2" that works with the lower bound of the clock period under the clock skew schedule in which T_{Host}=0, T_{R1}=0, T_{R2}=0, and T_{R3}=3. Suppose that the power-delay sensitivity of each logic gate is 1. Then, compared with circuit ex2', the power saving of circuit ex2" achieves 7, i.e., (7-2)×1+(5-3)×1= 7. In next section, we will present an approach to obtain circuit ex2".

C. Our Motivation

From the discussions on circuit ex1 and circuit ex2, we have the following two observations.

(1) Many hold violations can be resolved by gate downsizing. Since previous works [5-8] only use buffer insertion to resolve hold violations, they suffer from a large overhead on leakage power.

(2) Not all hold violations can be resolved by gate downsizing. Since previous works [12-14] only use gate downsizing, they often does not work with the lower bound of the clock period.

Based on those observations, we are motivated to study the combination of buffer insertion and gate downsizing during clock skew scheduling.

IV. THE PROPOSED APPROACH

This section proposes a two-step methodology for leakage-power-aware clock period minimization. At the first step, we use a LP approach to minimize the number of required inserted buffers for achieving the lower bound of the clock period. Then, at the second step, we use a LP approach [12] to apply gate downsizing to all logic gates for leakage power minimization. Section IV-A addresses the first step. Section IV-B addresses the second step. Section IV-C gives examples.

A. Minimization of the Number of Buffers

Our basic idea is below: only for those hold violations that cannot be resolved by gate downsizing, we apply buffer insertion for achieving the lower bound of the clock period. Compared with previous work [8] that also attempts to reduce the number of required inserted buffers, our approach has the following two advantages. First, our approach considers gate downsizing, while previous work [8] does not. Therefore, the number of our required inserted buffers is much less than previous work [8]. Second, our approach is a LP approach, while previous work [8] is a MILP approach. Thus, our approach has a smaller time complexity.

We introduce the notations used in our LP approach. The real-value variable T_{Ri} denotes the clock arrival time of register R_i. The real-value variables E_A and L_A denote the data earliest arrival time and the data latest arrival time, respectively. Since register R_i may serve as both the begin port and the end port, in order to distinguish them, when register R_i serves as the begin port, we use the notations E_{Ri} and L_{Ri}; when register R_i serves as the end port, we use the notations $E_{Ri'}$ and $L_{Ri'}$. The real-value variable δ_A denotes the amount of increased delay of logic gate A (due to gate downsizing). The real-value variable $X_{A,B}$ denotes the amount of the delay inserted into the wire <A,B> due to buffer insertion. The constant D_A denotes the original delay of logic gate A. The constant P_{LB} denotes the lower bound of the clock period. The set W denotes a set that includes all the wires in the circuit. The constant INF denotes a very large constant value that approximates infinity, i.e., INF→∞. Actually, in theory, with an analysis to our LP approach, we can derive the lower bound on the constant INF (for guaranteeing that the circuit obtained by our LP approach can work with the lower bound of the clock period) below: the multiplication of the constant INF and the constant ρ should be greater than the number of wires in the circuit, where the constant ρ should be less than or equal to the smallest required inserted delay among all delay insertions in the solutions of previous works [5-7]. Due to the limit on the number of pages, here we omit the formal proof.

To minimize the number of required inserted buffers, our basic idea is to let the cost of each buffer insertion be very tremendous. We define the real-value variable $Y_{A,B}$ to reflect the cost: if no buffer is inserted into the wire <A,B>, we let the value of variable $Y_{A,B}$ be only 1; on the other word, if a buffer is inserted into the wire <A,B>, we let the value of variable $Y_{A,B}$ be the multiplication of $X_{A,B}$ and INF (note that, from the definition of the constant ρ, if the value of variable $X_{A,B}$ is nonzero, the value of variable $X_{A,B}$ is not less than ρ; thus, the value of variable $Y_{A,B}$, i.e., the multiplication of $X_{A,B}$ and INF, is greater than the number of wires in the circuit). Then, our objective function can become to ***minimize*** $\sum_{\forall <A,B> \in W} Y_{A,B}$.

The constraints in our LP approach are elaborated below. Since the amount of delay increase should be non-negative, for each wire <A,B>, we have the constraint**:** $\boldsymbol{X_{A,B} \geq 0}$, and for each logic gate A, we have the constraint: $\boldsymbol{\delta_A \geq 0}$.

For each timing arc from logic gate A to logic gate B, since the latest data arrival time of logic gate B should not be less than the summation of the latest data arrival time of logic gate A, the original delay of logic gate A, the amount of the increased delay of logic gate A, and the amount of the delay inserted into the wire <A,B>, we have the constraint: $\boldsymbol{L_B \geq L_A + D_A + X_{A,B} + \delta_A}$.
since the earliest data arrival time of logic gate B should not be greater than the summation of the earliest data arrival time of logic gate A, the original delay of logic gate A, the amount of the increased delay of logic gate A, and the amount of the delay inserted into the wire <A,B>, we have the constraint: $\boldsymbol{E_B \leq E_A + D_A + X_{A,B} + \delta_A}$.

For each register R_i that serves as the begin port of any timing path, the data arrival time of register R_i should be the clock arrival time of register R_i; thus, we have the constraints: $\boldsymbol{L_{Ri} = T_{Ri}}$ and $\boldsymbol{E_{Ri} = T_{Ri}}$.

For each register R_i that serves as the end port of any timing path, since the latest data arrival time of register R_i should not be later than the next clock arrival time of register R_i, we have the constraint: $\boldsymbol{L_{Ri'} \leq P_{LB} + T_{Ri}}$.
since the earliest data arrival time of register R_i should not be earlier than the clock arrival time of register R_i, we have the constraint: $\boldsymbol{E_{Ri'} \geq T_{Ri}}$.

According to the definition of variable $Y_{A,B}$, we have the constraints: $\boldsymbol{Y_{A,B} \geq 1}$ and $\boldsymbol{Y_{A,B} \geq X_{A,B} \times INF}$.

B. Leakage Power Minimization

At the second step, we use a LP approach to increase the delays of logic gates as possible for leakage power minimization. In fact, the framework of our LP approach is the same as that of previous work [12]. The only difference is that we use the circuit obtained at the first step as the input circuit (i.e., those extra buffers inserted at the first step are included).

We use the constant C_A to denote the power-delay sensitivity of logic gate A. Then, our objective function is to ***maximize*** $\sum_{\forall A \in GT} C_A \times \delta_A$, and the constraints in our LP approach are the same as previous work [12].

C. Examples

Take circuit ex1 for illustration. At the first step, we find that no buffer insertion is needed. Thus, the input circuit of step 2 is still circuit ex1. Then, at the second step, we obtain circuit ex1" (displayed in Figure 6).

Take circuit ex2 for illustration. At the first step, we find that one extra buffer (whose delay value is 2) should be inserted. Thus, the input circuit of step 2 is circuit ex2'. Then, at the second step, we obtain circuit ex2" (displayed in Figure 11)

V. EXPERIMENTAL RESULTS

We use Extended-LINGO Release 13.0 as the mathematical solver for our approach and previous works. The circuits in ISCAS'89 benchmark suite are targeted to TSMC 65 nm cell library to test the effectiveness of our approach. Table 1 tabulates the smallest feasible clock periods of different approaches. The column *Long* denotes the longest path delay. The column *[1]* denotes the minimum-period clock skew scheduling [1]. The column *[12]* denotes the leakage-power-aware clock skew scheduling [12] (that only uses gate downsizing). The column *[6]* denotes previous work [6]. The column *[8]* denotes previous work [8]. Note that our approach, previous work [6], and previous work [8] can always achieve the lower bound of the clock period, while previous work [12] often does not achieve the lower bound of the clock period.

Table 2 gives the required inserted buffers for achieving the lower bound of the clock period. Compared with [6], our average reduction achieves 80.31%; compared with [8], our average reduction achieves 78.62%.

Table 3 gives the leakage power under the lower bound of the clock period. For those circuits that previous work [12] cannot achieve the lower bound of the clock period, their leakage power is denoted as NA in the column *[12]*. Our approach can always achieve the smallest leakage power. Compared with [6], our average reduction achieves 16.61%; compared with [8], our average reduction achieves 16.52%.

Table 4 gives the total power consumption (including leakage power, switching power, and short-circuit power) under the lower bound of the clock period. Our approach can always achieve the smallest total power consumption. Compared with [6], our average reduction achieves 12.96%; compared with [8], our average reduction achieves 12.83%.

Finally, in Table 5, we report the CPU times. The CPU time of our approach is much smaller than that of [8]. Therefore, compared with [8], our approach can achieve a much better result with a much smaller CPU time. The CPU time of our approach is slightly larger than those of [12] and [6]. The reason is that our approach uses two steps in order to achieve the lower bound of the clock period with the minimum leakage power.

VI. CONCLUSIONS

In this paper, we propose a two-step design methodology for the combination of buffer insertion, gate downsizing, and clock skew scheduling in order to achieve the lower bound of the clock period with the minimum leakage power. Our approach is the first leakage-power-aware clock skew scheduling that can guarantee working with the lower bound of the clock period. Compared with previous works, benchmark data show that our approach achieves the best results in terms of the clock period, the leakage power, and the total power consumption.

ACKNOWLEDGMENT

This work was supported in part by the National Science Council of Taiwan, R.O.C., under grant number NSC 99-2221-E-033-061-MY3.

REFERENCES

[1] J.P. Fishburn, "Clock Skew Optimization", IEEE Trans. on Computers, vol. 39, no. 7, pp. 945—951, 1990.

[2] N. Maheshwari and S.S. Sapatnekar, "Timing Analysis and Optimization of Sequential Circuits", Kluwer Academic Publishers, 1999.

[3] N.V. Shenoy, R.K. Brayton, and A.L. Sangiovanni-Vincentelli, "Minimum Padding to Satisfy Short Path Constraints", Proc. of IEEE ICCAD, pp. 156—161, 1993.

[4] B. Taskin and I.S. Kourtev, "Delay Insertion Method in Clock Skew Scheduling", IEEE Trans. on CAD, vol. 25, no.4, pp. 651—663, 2006.

[5] S.H. Huang and Y.T. Nieh, "Synthesis of Nonzero Clock Skew Circuits", IEEE Trans. on CADs, vol. 25, no.6, pp. 961—976, 2006.

[6] S.H. Huang, C.H. Cheng, C.M. Chang, and Y.T. Nieh, "Clock Period Minimization with Minimum Delay Insertion", Proc. of IEEE DAC, pp. 970—975, 2007.

[7] W.P. Tu, S.H. Huang, and C.H. Cheng, "Co-Synthesis of Data Paths and Clock Control Paths for Minimum-Period Clock Gating", Proc. of IEEE DATE, pp. 1831—1836, 2013.

[8] S.H. Huang, G.Y. Ghuo, and W.L. Huang, "Minimum Inserted Buffers for Clock Period Minimization", Journal of Information Science and Engineering, vol. 27, no. 5, pp. 1513—1526, 2011.

[9] M. Ketkar and S.S. Sapatnekar, "Standby Power Optimization via Transistor Sizing and Dual Threshold Voltage Assignment", Proc. of IEEE ICCAD, pp. 375—378, 2002.

[10] S. Shah, A. Srivastava, D. Sharma, D. Sylvester, D. Blaauw, and V. Zolotov, "Discrete Vt Assignment and Gate Sizing Using a Selfsnapping Continuous Formulation, Proc. of IEEE ICCAD, pp. 704—711, 2005.

[11] Y. Liu and J. Hu, "A New Algorithm for Simultaneous Gate Sizing and Threshold Voltage Assignment", IEEE Trans. on CAD, vol. 29, no. 2, pp. 223—234, 2010.

[12] M. Ni and S.O. Memik, "Leakage Power-Aware Clock Skew Scheduling: Converting Stolen Time into Leakage Power Reduction", Proc. of IEEE DAC, pp. 610—613, 2008.

[13] M. Tie, H. Dong, T. Wang, and X. Chen, "Dual-Vth Leakage Reduction with Fast Clock Skew Scheduling Enhancement", Proc. of IEEE DATE, pp. 520—525, 2010.

[14] L. Li, J. Sun, Y. Lu, H. Zhou, and X. Zeng, "Low Power Discrete Voltage Assignment under Clock Skew Scheduling", Proc. of IEEE ASP-DAC, pp. 515—520, 2011.

Table 1: Comparisons on the smallest feasible clock periods.

Circuit	Clock Period (ns)					
	Long	[1]	[12]	[6]	[8]	Ours
S1269	1.48	1.16	0.73	0.71	0.71	0.71
S3271	1.08	0.81	0.80	0.56	0.56	0.56
S3384	2.73	2.36	1.29	1.10	1.10	1.10
S4863	2.64	2.54	2.50	1.46	1.46	1.46
S6669	3.57	3.56	1.25	1.09	1.09	1.09
S13207	1.06	0.85	0.74	0.74	0.74	0.74
S15850	0.98	0.89	0.70	0.67	0.67	0.67
S35932	1.23	1.07	1.07	0.99	0.99	0.99
S38417	1.54	1.31	1.26	1.26	1.26	1.26

Table 2: Comparisons on required inserted buffers.

Circuit	Number of Required Buffers			Our Reduction	
	[6]	[8]	Ours	vs [6]	vs [8]
S1269	33	26	1	96.97%	96.15%
S3271	45	39	16	64.44%	58.97%
S3384	38	35	9	76.32%	74.29%
S4863	188	154	17	90.96%	88.96%
S6669	622	493	4	99.36%	99.19%
S13207	43	6	0	100.00%	100.00%
S15850	19	10	1	94.74%	90.00%
S35932	288	288	288	0.00%	0.00%
S38417	2	1	0	100.00%	100.00%

Table 3: Comparisons on the leakage power.

Circuit	Leakage Power (nW)				Our Reduction	
	[12]	[6]	[8]	Ours	vs [6]	vs [8]
S1269	NA	3128	3126	2479	20.75%	20.70%
S3271	NA	8325	8323	6894	17.19%	17.17%
S3384	NA	8474	8473	6852	19.14%	19.13%
S4863	NA	12702	12691	9701	23.63%	23.56%
S6669	NA	16370	16330	12652	22.71%	22.52%
S13207	18777	20242	20230	18777	7.24%	7.18%
S15850	NA	18098	18010	16368	9.56%	9.12%
S35932	NA	68767	68767	56678	17.58%	17.58%
S38417	90955	103047	103046	90955	11.73%	11.73%

Table 4: Comparisons on the total power consumption.

Circuit	Total Power Consumption (mW)				Our Reduction	
	[12]	[6]	[8]	Ours	vs [6]	vs [8]
S1269	NA	3.13	3.13	2.61	16.61%	16.61%
S3271	NA	8.77	8.76	7.55	13.91%	13.81%
S3384	NA	8.72	8.71	7.50	13.99%	13.89%
S4863	NA	12.96	12.92	10.59	18.29%	18.03%
S6669	NA	17.04	16.92	13.83	18.84%	18.26%
S13207	24.10	25.56	25.52	24.10	5.71%	5.56%
S15850	NA	21.91	21.90	20.40	6.89%	6.85%
S35932	NA	84.93	84.93	72.79	14.29%	14.29%
S38417	106.79	116.27	116.27	106.79	8.15%	8.15%

Table 5: Comparisons on the CPU times.

Circuit	CPU Time (s)			
	[12]	[6]	[8]	Ours
S1269	1	1	63	2
S3271	1	2	69	2
S3384	2	3	104	6
S4863	2	4	283	8
S6669	5	10	351	18
S13207	4	12	784	17
S15850	4	10	691	18
S35932	16	46	803	66
S38417	14	24	176	36