JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 27, 1513-1526 (2011)

Minimum Inserted Buffers for Clock Period Minimization*

SHIH-HSU HUANG+, GUAN-YU JHUO AND WEI-LUN HUANG
Department of Electronic Engineering
Chung Yuan Christian University
Chungli, 320 Taiwan
+*E-mail: shhuang@cycu.edu.tw*

It is well known that the combination of clock skew scheduling and delay insertion can achieve the lower bound of sequential timing optimization. Previous works focus on the minimization of required inserted delay. However, from the viewpoint of design closure, minimizing the number of inserted buffers is also very important. In this paper, we propose an MILP (mixed integer linear programming) approach to minimize the number of inserted buffers under the constraints on the lower bound of sequential timing optimization and the lower bound of required inserted delay. Note that our MILP approach guarantees obtaining the optimal solution. Experimental results consistently show that our MILP approach can greatly reduce the number of inserted buffers.

Keywords: high performance, sequential circuits, timing optimization, clock period, buffer insertion

1. INTRODUCTION

Among the various objectives in the development of a high-speed sequential circuit, clock period minimization is always one of the most important topics. There are two methods to resolve the timing violations caused by long paths: one is to apply logic optimization techniques [1, 2] for reducing the delays of long paths, and the other is to apply sequential timing optimization techniques [3, 4] for adjusting the timing slacks among the data paths. In general, logic optimization techniques are applied first. For those long paths whose delays are difficult to further reduce, sequential timing optimization is necessary. Since sequential timing optimization does not reduce the path delay, the lower bound of sequential timing optimization [5] is determined by setup constraints.

Clock skew scheduling [6-8] is a useful sequential timing optimization technique to improve the circuit speed. 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. However, due to the limitation of hold constraints, clock skew scheduling often cannot achieve the low bound of sequential timing optimization [5]. In fact, a hold violation means the previous data is not held long enough at the destination register to be properly clocked through. Therefore, the hold violations for achieving the lower bound of sequential timing optimization can be resolved by applying the delay insertion, which is referred to as the *padding* method [9]. As a result, recently, several approaches [10-14] have been paid to study the combination of clock skew scheduling and delay insertion for achieving the low

Received May 21, 2010; revised August 9 & November 1, 2010; accepted November 28, 2010.
Communicated by Yao-Wen Chang.

* A preliminary version, entitled "Minimum Buffer Insertions for Clock Period Minimization" [15], has appeared in the Proceedings of IEEE International Symposium on Computer, Communication, Control and Automation (3CA), 2010. This work was supported in part by the National Science Council of Taiwan, R.O.C., under grant No. NSC 99-2221-E-033-061-MY3.

bound of sequential timing optimization.[1]

Huang *et al*. [10] present the first work to study the combination of clock skew scheduling and delay insertion. Then, Huang *et al.* [12] propose the DIANA algorithm to deal with the combination of clock skew scheduling and delay insertion. The DIANA algorithm guarantees achieving the low bound of sequential timing optimization, but it only heuristically minimizes the required inserted delay. Furthermore, the time complexity of the DIANA algorithm is too high. Thus, Huang *et al*. [13] propose the RCA algorithm to substitute the DIANA algorithm. Note that both the RCA algorithm and the DIANA algorithm obtain the same result. However, compared with the DIANA algorithm, the RCA algorithm greatly reduces the run time.

Different from earlier heuristic approaches [10-13], Huang *et al*. [14] propose a linear programming approach to formally formulate the simultaneous application of clock skew scheduling and delay insertion. Note that their approach [14] not only achieves the lower bound of sequential timing optimization but also achieves the lower bound of required inserted delay. However, they [14] still do not consider the minimization of the number of inserted buffers (*i.e*., the number of wires that require delay insertions).

In fact, fewer inserted buffers are easier to achieve the design closure. Therefore, from the viewpoint of design closure, minimizing the number of inserted buffers should also be an important objective. However, to the best of our knowledge, no attention has been paid to the reduction of the number of inserted buffers for clock period minimization. Based on that observation, we present the first attempt to deal with this problem. Given the lower bound of sequential timing optimization and the lower bound of required inserted delay as the constraints, our objective is to minimize the number of inserted buffers.

In this paper, we present an MILP (mixed integer linear programming) approach to formally formulate our problem. Two MILP programs are proposed in this paper. In the first MILP program, we assume that the number of buffer types is unlimited (*i.e*., any delay value can be implemented by a single buffer). In the second MILP program, we assume that the number of buffer types is limited (*i.e*., a delay value may need to be implemented by a combination of buffers). Note that our MILP approach guarantees minimizing the number of inserted buffers under the constraints on the lower bound of sequential timing optimization and the lower bound of required inserted delay. Compared with previous work [14], experimental results show that our MILP approach can greatly reduce the number of inserted buffers.

The rest of this paper is organized as follows. Section 2 provides the background materials. Section 3 presents our motivation. In section 4, we propose an MILP program under the assumption of an unlimited buffer types. In section 5, we propose an MILP program under the assumption of a limited buffer types. We demonstrate our experimental results in section 6. Finally, some concluding remarks are given in section 7.

2. PRELIMINARIES

In a sequential circuit, a data path $R_i \rightarrow R_j$ defined as the combinational logic from register R_i to register R_j. For each data path $R_i \rightarrow R_j$, there are two types of clocking constraints: the setup constraint and the hold constraint. Let T_{ci} denote the clock arrival time

[1] Note that the approach of Taskin *et al*. [11] does not guarantee achieving the low bound of sequential timing optimization.

of register R_i. To prevent the data reaching a register too late relative to the following clock pulse, the clock skew must satisfy the following setup constraint: $T_{ci} - T_{cj} \leq P - \max(R_i, R_j)$, where P is the clock period and $\max(R_i, R_j)$ denotes the maximum delay from register R_i to register R_j. To prevent the same clock pulse triggering the same data into two adjacent registers, the clock skew must satisfy the following hold constraint: $T_{cj} - T_{ci} \leq \min(R_i, R_j)$, where $\min(R_i, R_j)$ denotes the minimum delay from register R_i to register R_j. A sequential circuit works with clock period P, if there is a clock skew schedule (*i.e.*, a solution of clock arrival times of registers) that satisfies all the clocking constraints.

Conventional clock skew scheduling approaches [6-8] use the constraint graph to represent the clock constraints. In the constraint graph, each vertex denotes a register and each directed edge denotes a data path. A special vertex, called the host (abbreviated as H), is used for synchronization with the primary inputs and the primary outputs. There are two types of directed edges: S-edges and H-edges. The setup constraint of data path $R_i \rightarrow R_j$ 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-$\max(R_i, R_j)$; and the hold constraint of data path $R_i \rightarrow R_j$ 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 $\min(R_i, R_j)$. Then, a sequential circuit works with clock period P if only and if there is no negative cycle in the corresponding constraint graph.

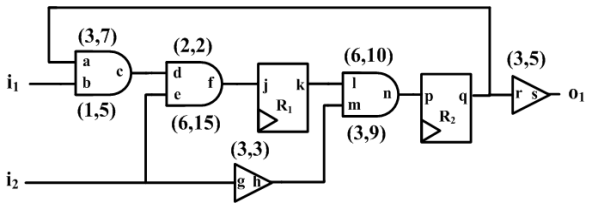


Fig. 1. Example circuit ex.

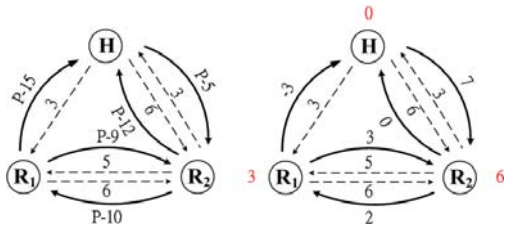
Let's use the edge-triggered circuit ex shown in Fig. 1 as an example. This circuit has two registers, including register R_1 and register R_2. This circuit has five logic gates. In Fig. 1, each timing arc of logic gate is associated with its minimum delay and its maximum delay. For example, the minimum delay and the maximum delay of the timing arc from pin a to pin c are 3 and 7, respectively; the minimum delay and the maximum delay of the timing arc from pin b to pin c are 1 and 5, respectively; and so on. Besides, following the assumptions of previous works [10-14], in this paper, we assume that the wire delay is 0.[2] In other words, the delay of the wire from pin d to pin c is 0; the delay of the wire from pin f to pin j is 0; and so on.

With an analysis to Fig. 1, we know that $\min(R_1, R_2)$ is 6, $\max(R_1, R_2)$ is 10, $\min(R_2, R_1)$ is 5, $\max(R_2, R_1)$ is 9, and so on. Fig. 2 (a) displays the corresponding constraint graph, in which S-edges are drawn in solid lines and H-edges are drawn in dashed lines. By applying clock skew scheduling, the smallest feasible is 12 and the clock skew schedule is $T_{host} = 0$, $T_{c1} = 3$ and $T_{c2} = 6$. Fig. 2 (b) gives the corresponding constraint graph when clock period is 12. Note that this constraint graph has no negative cycle. However, there is a critical cycle $R_1 \rightarrow H \rightarrow R_1$, in which the summation of weights is 0. Thus, both the

[2] In the previous works [10-14] and our approach, a wire corresponds to a timing arc from an output pin of a logic gate to an input pin of another logic gate. Actually, the wire delay is not limited to 0. If the wire delay is a constant (*i.e.*, the delay of each wire is fixed), the previous works [10-14] and our approach still can work.

setup constraint and the hold constraint of the data path from host to R_1 are critical when clock period is 12.

Actually, the lower bound of sequential timing optimization is only determined by setup constraints [5]. In this example, the lower bound of sequential timing optimization should be 10. However, due to the hold constraint of the data path from host to R_1, clock skew scheduling cannot achieve the lower bound of sequential timing optimization. Therefore, unless we insert delay into the data path from host to register R_1 (for resolving the hold violation), the lower bound of sequential timing cannot be achieved.



(a) (b)

Fig. 2. Constraint graph.

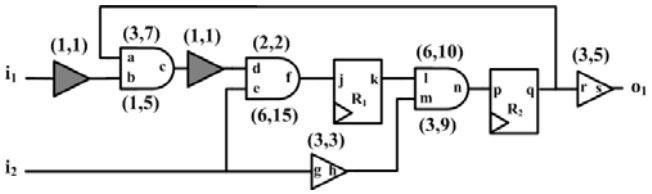


Fig. 3. Padded circuit ex'.

Huang *et al*. [14] propose a linear programming approach for the simultaneous application of clock skew scheduling and delay insertion in order to further reduce the clock period. Their approach not only achieves the lower bound of sequential timing optimization but also achieves the lower bound of required inserted delay. Consider the example circuit ex. From [14], we know the lower bound of required inserted delay is 2. Fig. 3 gives a padded circuit ex', which is a feasible solution of the linear program proposed by Huang *et al*. [14]. In the padded circuit ex', the required inserted delay has been minimized: an extra buffer, whose delay is 1, is added into the wire from pin i_1 to pin *b*; and another extra buffer, whose delay is 1, is added into the wire from pin *c* to pin *d*. The padded circuit ex' works with clock period 10 (*i.e*., the lower bound of sequential timing optimization) under the clock skew schedule $T_{host} = 0$, $T_{c1} = 5$, and $T_{c2} = 5$. Note that, in the padded circuit ex', the number of inserted buffers (*i.e*., the number of wires that require delay insertions) is 2.

3. MOTIVATION

Huang *et al*. [14] does not consider the minimization of the number of inserted buffers (*i.e*., the number of wires that require delay insertion). However, fewer inserted buffers

are easier to achieve the design closure. Thus, from the viewpoint of design closure, minimizing the number of inserted buffers is also very important.

Take the circuit ex for example. Under the lower bound of sequential timing optimization and the lower bound of required inserted delay, we can find another padded circuit ex" (different from circuit ex') in which the number of inserted buffers is only 1. Consider the padded circuit ex" displayed in Fig. 4. An extra buffer, whose delay is 2, is added into the wire from pin i_1 to pin b. The padded circuit ex" works with clock period 10 (*i.e.*, the lower bound of sequential timing optimization) under the clock skew schedule $T_{host} = 0$, $T_{c1} = 5$, and $T_{c2} = 5$. However, in the padded circuit ex", the number of inserted buffers is only 1.

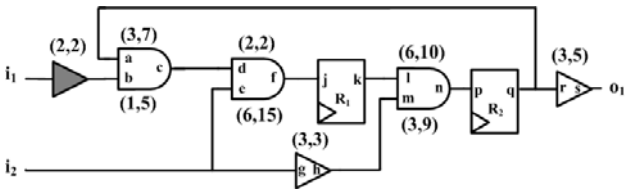


Fig. 4. Padded circuit ex".

Based on the above observation, we are motivated to minimize the number of inserted buffers (under the lower bound of sequential timing optimization and the lower bound of required inserted delay).

4. THE PROPOSED APPROACH

In this section, we propose an MILP approach to minimize the number of inserted buffers under the constraints on the lower bound of sequential timing optimization and the lower bound of required inserted delay. Note that the lower bound of sequential timing optimization and the lower bound of required inserted delay can be obtained by the previous work [14]. Therefore, our approach can be used as the post-processing approach for the previous work [14].

First, we define the following notations.

(1) The real-value variable $X_{u,v}$ denotes the inserted delay of each wire from pin u to pin v (also denoted as $u \rightarrow v$).
(2) The notations EA_u and LA_u denote the earliest data arrival time and latest data arrival time of pin u, respectively.
(3) The constant value $D_{ARC_{u,v}(\min)}$ is the minimum delay of the timing arc (of logic gate) from pin u to pin v.
(4) The constant value $D_{ARC_{u,v}(\max)}$ is the maximum delay of the timing arc (of logic gate) from pin u to pin v.
(5) The real-value variable T_{ci} denotes the clock arrival time of register R_i.
(6) The constant value P_{LB} denotes the lower bound of sequential timing optimization.
(7) The constant value C denotes the lower bound of required inserted delay.
(8) The notation W denotes the set which includes all the wires.

(9) The notation $I_{u,v}$ is a binary variable. If the wire $u \rightarrow v$ requires delay insertion, then $I_{u,v} = 1$; otherwise, $I_{u,v} = 0$.

Next, we introduce the objective function and the constraints. Our goal is to minimize the number of inserted buffers. Therefore, we describe the objective functions below:

$$\text{Minimize}\{\sum_{(u\rightarrow v)\in W} I_{u,v}\}.$$

It should be non-negative for the inserted delay of each wire form pin u to pin v. Thus, we have the following constraint:

$$0 \le X_{u,v}. \tag{1}$$

According to the hold constraint, for each timing arc from pin u to pin v, the earliest data arrival time of pin v should be not later than the earliest data arrival time of pin u plus the minimum delay of the timing arc from pin u to pin v. Thus, we have the following constraint:

$$EA_v \le EA_u + D_{ARC_{u,v(\min)}}. \tag{2}$$

According to the setup constraint, for each timing arc from pin u to pin v, the latest data arrival time of pin v should be not earlier than the latest data arrival time of pin u plus the maximum delay of the timing arc from pin u to pin v. Thus, we have the following constraint:

$$LA_u + D_{ARC_{u,v(\min)}} \le LA_v. \tag{3}$$

According to the hold constraint, for each wire from pin u to pin v, the earliest data arrival time of pin v should be not later than the earliest data arrival time of pin u plus the inserted delay of this wire. Consequently, we have the following constraint:

$$EA_v \le EA_u + X_{u,v}. \tag{4}$$

According to the setup constraint, for each wire from pin u to pin v, the latest data arrival time of pin v should be not earlier than the latest data arrival time of pin u plus the inserted delay of this wire. Consequently, we have the following constraint:

$$LA_u + X_{u,v} \le LA_v. \tag{5}$$

Consider data path $R_i \rightarrow R_j$. Suppose u is the starting pin of the data path. Then, the earliest data arrival time of pin u should be equal to the clock arrival time of register R_i. Thus, we have the following constraint:

$$EA_u = T_{ci}. \tag{6}$$

Consider data path $R_i \rightarrow R_j$. Suppose u is the starting pin of the data path. Then, the latest data arrival time of pin u should also be equal to the clock arrival time of register R_i. Thus, we have the following constraint:

$$LA_u = T_{ci}. \tag{7}$$

Due to the hold constraint, suppose v is the ending pin of data path $R_i \rightarrow R_j$, the earliest data arrival time of pin v should be not earlier than the clock arrival time of R_j. Therefore, we have the following constraint:

$$T_{cj} \leq EA_v. \tag{8}$$

Due to the setup constraint, suppose v is the ending pin of data path $R_i \rightarrow R_j$, the latest data arrival time of pin v should be not later than the clock arrival time of next clock pulse of register R_j. Therefore, we have the following constraint:

$$LA_v \leq T_{cj} + P_{LB}. \tag{9}$$

The summation of inserted delay should be equal to the lower bound of required inserted dely. As a result, we have the following constraint:

$$\sum_{(u \rightarrow v) \in W} X_{u,v} = C. \tag{10}$$

For each wire from pin u to pin v, if the inserted delay $X_{u,v}$ is 0, then the binary variable $I_{u,v}$ should be 0; otherwise, the binary variable $I_{u,v}$ should be 1. Since the inserted delay is impossible greater than the lower bound of required inserted delay, we have the following constraint:

$$C \times I_{u,v} \geq X_{u,v}. \tag{11}$$

Let's use the edge-triggered circuit ex shown in Fig. 1 as an example. Our objective function is to minimize ($I_{i1,b} + I_{q,a} + I_{c,d} + I_{i2,e} + I_{i2,g} + I_{f,j} + I_{k,l} + I_{h,m} + I_{n,p} + I_{q,r} + I_{s,o1}$).

Due to Eq. (1), we have the following constraints: $X_{i1,b} \geq 0$; $X_{q,a} \geq 0$; $X_{c,d} \geq 0$; $X_{i2,e} \geq 0$; $X_{i2,g} \geq 0$; $X_{f,j} \geq 0$; $X_{k,l} \geq 0$; $X_{h,m} \geq 0$; $X_{n,p} \geq 0$; $X_{q,r} \geq 0$; and $X_{s,o1} \geq 0$.

Due to Eqs. (2) and (3), we have the following constraints: $EA_a + 3 \geq EA_c$; $LA_a + 7 \leq LA_c$; $EA_b + 1 \geq EA_c$; $LA_b + 5 \leq LA_c$; $EA_d + 2 \geq EA_f$; $LA_d + 2 \leq LA_f$; $EA_e + 6 \geq EA_f$; $LA_e + 15 \leq LA_f$; $EA_g + 3 \geq EA_h$; $LA_g + 3 \leq LA_h$; $EA_l + 6 \geq EA_n$; $LA_l + 10 \leq LA_n$; $EA_m + 3 \geq EA_n$; $LA_m + 9 \leq LA_n$; $EA_r + 3 \geq EA_s$; and $LA_r + 5 \leq LA_s$.

Due to Eqs. (4) and (5), we have the following constraints: $EA_{i1} + X_{i1,b} \geq EA_b$; $LA_{i1} + X_{i1,b} \leq LA_b$; $EA_q + X_{q,a} \geq EA_a$; $LA_q + X_{q,a} \leq LA_a$; $EA_c + X_{c,d} \geq EA_d$; $LA_c + X_{c,d} \leq LA_d$; $EA_{i2} + X_{i2,e} \geq EA_e$; $LA_{i2} + X_{i2,e} \leq LA_e$; $EA_{i2} + X_{i2,g} \geq EA_g$; $LA_{i2} + X_{i2,g} \leq LA_g$; $EA_f + X_{f,j} \geq EA_j$; $LA_f + X_{f,j} \leq LA_j$; $EA_k + X_{k,l} \geq EA_l$; $LA_k + X_{k,l} \leq LA_l$; $EA_h + X_{h,m} \geq EA_m$; $LA_h + X_{h,m} \leq LA_m$; $EA_n + X_{n,p} \geq EA_p$; $LA_n + X_{n,p} \leq LA_p$; $EA_q + X_{q,r} \geq EA_r$; $LAq + X_{q,r} \leq LA_r$; $EA_s + X_{s,o1} \geq EA_{o1}$; and $LA_s + X_{s,o1} \leq LA_{o1}$.

Due to Eqs. (6) and (7), we have the following constraints: $EA_{i1} = T_{host}$; $LA_{i1} = T_{host}$; $EA_{i2} = T_{host}$; $LA_{i2} = T_{host}$; $EA_k = T_{c1}$; $LA_k = T_{c1}$; $EA_q = T_{c2}$; and $LA_q = T_{c2}$.

Due to Eq. (8), we have the following constraints: $EA_j \geq T_{c1}$; $EA_p \geq T_{c2}$; and $EA_{o1} \geq T_{host}$.

Due to Eq. (9), we have the following constraints: $T_{c1} + 10 \geq LA_j$; $T_{c2} + 10 \geq LA_p$; and $T_{host} + 10 \geq LA_{o1}$.

Due to Eq. (10), we have the following constraint: $X_{i1,b} + X_{q,a} + X_{c,d} + X_{i2,e} + X_{i2,g} + X_{f,j}$

$+ X_{k,l} + X_{h,m} + X_{n,p} + X_{q,r} + X_{s,o1} = 2$.

Due to Eq. (11), we have the following constraints: $2 \times I_{i1,b} \geq X_{i1,b}$; $2 \times I_{q,a} \geq X_{q,a}$; $2 \times I_{c,d} \geq X_{c,d}$; $2 \times I_{i2,e} \geq X_{i2,e}$; $2 \times I_{i2,g} \geq X_{i2,g}$; $2 \times I_{f,j} \geq X_{f,j}$; $2 \times I_{k,l} \geq X_{k,l}$; $2 \times I_{h,m} \geq X_{h,m}$; $2 \times I_{n,p} \geq X_{n,p}$; $2 \times I_{q,r} \geq X_{q,r}$; and $2 \times I_{s,o1} \geq X_{s,o1}$.

After solving the MILP formulation, we have $I_{i1,b} = 1$, $X_{i1,b} = 2$, $I_{q,a} = X_{q,a} = I_{c,d} = X_{c,d} = I_{i2,e} = X_{i2,e} = I_{i2,g} = X_{i2,g} = I_{f,j} = X_{f,j} = I_{k,l} = X_{k,l} = I_{h,m} = X_{h,m} = I_{n,p} = X_{n,p} = I_{q,r} = X_{q,r} = I_{s,o1} = X_{s,o1} = 0$, $T_{host} = 0$, $T_{c1} = 5$, and $T_{c2} = 5$.[3] The corresponding padded circuit ex" is displayed in Fig. 4. The number of inserted buffers is only 1. An extra buffer, whose delay is 2, is added into the wire from pin i_1 to pin b. The padded circuit ex" works with clock period 10 (*i.e.*, the lower bound of sequential timing optimization) under the clock skew schedule $T_{host} = 0$, $T_{c1} = 5$, and $T_{c2} = 5$. Note that, in the padded circuit ex", the number of inserted buffers is minimized.

5. MODIFIED MILP APPROACH

So far, we assume that the number of buffer types is unlimited (*i.e.*, any delay value can be implemented by a single buffer). However, in the cell library, the number of buffer types is often limited. As a result, a delay value may need to be implemented by a combination of buffers. In this section, we modify the MILP approach presented in section 4 to deal with this problem.

Since the number of buffer types is often limited, the delay insertion may require a combination of buffers (*i.e.*, not a single buffer). Since the binary variable $I_{u,v}$ only represents the wire $u \rightarrow v$ requires delay insertion or not, we cannot use the binary variable $I_{u,v}$ to count the total number of buffers inserted into the wire $u \rightarrow v$. Therefore, in the modified MILP approach, the binary variable $I_{u,v}$ is not used. Instead, we should add the following two new notations for the modified MILP approach.

(1) The cell library has n buffer types, including b_1, b_2, …, and b_n. The delays of buffers b_1, b_2, …, and b_n are d_1, d_2, …, and d_n, respectively.[4]
(2) The integer variable $Y_{u,v,i}$ denotes the number of buffer type b_i inserted into the wire $u \rightarrow v$.

Since the integer variable $Y_{u,v,i}$ denotes the number of the type b_i buffers inserted into the wire $u \rightarrow v$, the term $\sum_{i=1}^{n} Y_{u,v,i}$ represents the total number of all the buffers inserted into the wire $u \rightarrow v$. Therefore, we can use the term $\sum_{(u \rightarrow v) \in W} \sum_{i=1}^{n} Y_{u,v,i}$ to represent the number of inserted buffers. As a result, the objective function should be modified below:

---

[3] In [14], Huang *et al*. propose a heuristic approach to reduce the solution space. In this example, by applying the heuristic approach [14], we can have $I_{q,a} = X_{q,a} = I_{i2,e} = X_{i2,e} = I_{i2,g} = X_{i2,g} = I_{k,l} = X_{k,l} = I_{h,m} = X_{h,m} = I_{n,p} = X_{n,p} = I_{q,r} = X_{q,r} = I_{s,o1} = X_{s,o1} = 0$ in advance. As a result, the CPU time for solving the MILP formulation can be greatly reduced. It should be mentioned that the heuristic approach [14] does not guarantee minimizing the required inserted delay. Fortunately, in this example, the heuristic approach [14] achieves the lower bound of required inserted delay.

[4] Note that any delay value should be able to be implemented by a combination of buffers. To guarantee any delay value can be implemented by a combination of buffers, a possible method is to let the delay of one buffer type be the minimum resolution of delay value. As a result, at least, we can use the buffers in this type to implement any delay value.

$$\text{Minimize}\{\sum_{(u\to v)\in W}\sum_{i=1}^{n} Y_{u,v,i}\}.$$

Since the binary variable $I_{u,v}$ is not used in the modified MILP approach, Eq. (11) is also not used in the modified MILP approach. But we should add two new constraints for the modified MILP approach.[5] We elaborate these two new constraints below.

The number of buffer type b_i should be non-negative for the number of buffers in each wire form pin u to pin v. Thus, we have the following constraint:

$$0 \le Y_{u,v,i}. \tag{12}$$

The term $Y_{u,v,i} \times d_i$ corresponds to the delay summation of all the type b_i buffers inserted into the wire $u \to v$. Further, the term $\sum_{i=1}^{n} Y_{u,v,i} \times d_i$ is the delay summation of all the buffers inserted into the wire $u \to v$. Thus, we have the following constraint:

$$X_{u,v} = \sum_{i=1}^{n} Y_{u,v,i} \times d_i. \tag{13}$$

Let's use the edge-triggered circuit ex shown in Fig. 1 as an example. Suppose that the cell library has n buffer types, including b_1, b_2 and b_3, and the delays of buffers b_1, b_2 and b_3 are 0.1, 0.5, and 2.0, respectively.[6] Our objective function becomes to minimize ($Y_{i1,b,1} + Y_{i1,b,2} + Y_{i1,b,3} + Y_{q,a,1} + Y_{q,a,2} + Y_{q,a,3} + Y_{c,d,1} + Y_{c,d,2} + Y_{c,d,3} + Y_{i2,e,1} + Y_{i2,e,2} + Y_{i2,e,3} + Y_{i2,g,1} + Y_{i2,g,2} + Y_{i2,g,3} + Y_{f,j,1} + Y_{f,j,2} + Y_{f,j,3} + Y_{k,l,1} + Y_{k,l,2} + Y_{k,l,3} + Y_{h,m,1} + Y_{h,m,2} + Y_{h,m,3} + Y_{n,p,1} + Y_{n,p,2} + Y_{n,p,3} + Y_{q,r,1} + Y_{q,r,2} + Y_{q,r,3} + Y_{s,o1,1} + Y_{s,o1,2} + Y_{s,o1,3}$).

In addition to the constraints presented in section 4,[7] we should add some new constraints. Due to Eq. (12), we have the following constraints: $Y_{i1,b,1} \ge 0$; $Y_{i1,b,2} \ge 0$; $Y_{i1,b,3} \ge 0$; $Y_{q,a,1} \ge 0$; $Y_{q,a,2} \ge 0$; $Y_{q,a,3} \ge 0$; $Y_{c,d,1} \ge 0$; $Y_{c,d,2} \ge 0$; $Y_{c,d,3} \ge 0$; $Y_{i2,e,1} \ge 0$; $Y_{i2,e,2} \ge 0$; $Y_{i2,e,3} \ge 0$; $Y_{i2,g,1} \ge 0$; $Y_{i2,g,2} \ge 0$; $Y_{i2,g,3} \ge 0$; $Y_{f,j,1} \ge 0$; $Y_{f,j,2} \ge 0$; $Y_{f,j,3} \ge 0$; $Y_{k,l,1} \ge 0$; $Y_{k,l,2} \ge 0$; $Y_{k,l,3} \ge 0$; $Y_{h,m,1} \ge 0$; $Y_{h,m,2} \ge 0$; $Y_{h,m,3} \ge 0$; $Y_{n,p,1} \ge 0$; $Y_{n,p,2} \ge 0$; $Y_{n,p,3} \ge 0$; $Y_{q,r,1} \ge 0$; $Y_{q,r,2} \ge 0$; $Y_{q,r,3} \ge 0$; $Y_{s,o1,1} \ge 0$; $Y_{s,o1,2} \ge 0$; and $Y_{s,o1,3} \ge 0$.

Due to Eq. (13), we have the following constraints: $0.1 \times Y_{i1,b,1} + 0.5 \times Y_{i1,b,2} + 2 \times Y_{i1,b,3} = X_{i1,b}$; $0.1 \times Y_{q,a,1} + 0.5 \times Y_{q,a,2} + 2 \times Y_{q,a,3} = X_{q,a}$; $0.1 \times Y_{c,d,1} + 0.5 \times Y_{c,d,2} + 2 \times Y_{c,d,3} = X_{c,d}$; $0.1 \times Y_{i2,e,1} + 0.5 \times Y_{i2,e,2} + 2 \times Y_{i2,e,3} = X_{i2,e}$; $0.1 \times Y_{i2,g,1} + 0.5 \times Y_{i2,g,2} + 2 \times Y_{i2,g,3} = X_{i2,g}$; $0.1 \times Y_{f,j,1} + 0.5 \times Y_{f,j,2} + 2 \times Y_{f,j,3} = X_{f,j}$; $0.1 \times Y_{k,l,1} + 0.5 \times Y_{k,l,2} + 2 \times Y_{k,l,3} = X_{k,l}$; $0.1 \times Y_{h,m,1} + 0.5 \times Y_{h,m,2} + 2 \times Y_{h,m,3} = X_{h,m}$; $0.1 \times Y_{n,p,1} + 0.5 \times Y_{n,p,2} + 2 \times Y_{n,p,3} = X_{n,p}$; $0.1 \times Y_{q,r,1} + 0.5 \times Y_{q,r,2} + 2 \times Y_{q,r,3} = X_{q,r}$; $0.1 \times Y_{s,o1,1} + 0.5 \times Y_{s,o1,2} + 2 \times Y_{s,o1,3} = X_{s,o1}$.

After solving the MILP formulation, we have $Y_{i1,b,3} = 1$, $X_{i1,b} = 2$, $Y_{i1,b,1} = Y_{i1,b,2} = Y_{q,a,1} = Y_{q,a,2} = Y_{q,a,3} = Y_{c,d,1} = Y_{c,d,2} = Y_{c,d,3} = Y_{i2,e,1} = Y_{i2,e,2} = Y_{i2,e,3} = Y_{i2,g,1} = Y_{i2,g,2} = Y_{i2,g,3} = Y_{f,j,1} = Y_{f,j,2} = Y_{f,j,3} = Y_{k,l,1} = Y_{k,l,2} = Y_{k,l,3} = Y_{h,m,1} = Y_{h,m,2} = Y_{h,m,3} = Y_{n,p,1} = Y_{n,p,2} = Y_{n,p,3} = Y_{q,r,1} = Y_{q,r,2} = Y_{q,r,3} = Y_{s,o1,1} = Y_{s,o1,2} = Y_{s,o1,3} = X_{q,a} = X_{c,d} = X_{i2,e} = X_{i2,g} = X_{f,j} = X_{k,l} = X_{h,m} = X_{n,p} = X_{q,r} = X_{s,o1} = 0$, $T_{host} = 0$, $T_{c1} = 5$, and $T_{c2} = 5$. The corresponding padded circuit ex" is displayed in Fig. 4. The number of inserted buffers is only 1. An extra buffer, whose buffer type is

[5] In other words, the constraints in the modified MILP approach are Eqs. (1)-(10), (12), and (13).
[6] In this example, we assume that the minimum resolution of delay value is 0.1. Since the delay of buffer type b_1 is 0.1, we can use the buffers in type b_1 to implement any delay value.
[7] Note that the constraints derived from Eq. (11) can be removed.

b_3 (*i.e.*, delay is 2), is added into the wire from pin i_1 to pin b. The padded circuit ex" works with clock period 10 (*i.e.*, the lower bound of sequential timing optimization) under the clock skew schedule $T_{host} = 0$, $T_{c1} = 5$, and $T_{c2} = 5$. Note that, in the padded circuit ex", the number of inserted buffers is minimized.

6. EXPERIMENTAL RESULTS

We use Extended LINGO Release 10.0 as the MILP solver. The platform is Windows XP with 8GB RAM. We use ISCAS'89 benchmark circuits, which are targeted to UMC 0.18μm cell library, to test the effectiveness of our MILP approach. Table 1 tabulates the characteristics of benchmark circuits. The columns *Registers*, *Pins*, *Wires*, *Clock Period*, and *Inserted Delay* describe the numbers of register, the numbers of pins, the number of wires, the lower bound of sequential timing optimization, and the lower bound of required inserted delay, respectively.

Table 1. Characteristics of benchmark circuits.

Circuit	Registers	Pins	Wires	Clock Period (ns)	Inserted Delay (ns)
S1269	38	1764	1094	1.55	11.16
S1423	75	2117	1243	5.77	0.06
S3271	117	4733	2827	1.18	5.50
S3384	185	5172	2964	2.16	33.03
S4863	105	6850	4212	2.63	84.86
S6669	239	9379	3403	2.23	283.29
S15850	428	25554	14329	3.25	6.60
S35932	1728	51246	17829	2.03	121.60
S38417	1164	60751	33770	2.66	0.27

We perform two experiments. In the first experiment, we assume that the number of buffer types is unlimited (*i.e.*, any delay value can be implemented by a single buffer). In the second experiment, we assume that the number of buffer types is limited in the cell library. Our experimental results are discussed below.

In the first experiment, we compare the result of our approach with the result of previous approach [14]. Table 2 tabulates our results on the first experiment. The column *Inserted Buffers* denotes the number of inserted buffers. Note that our approach guarantees minimizing the number of inserted buffers. Experimental data show that, in most benchmark circuits, our approach can greatly reduce the number of inserted buffers. The column *CPU Time* reports two kinds of CPU times: the column *Original Solution Space* gives the CPU time for solving the MILP formulation without solution space reduction; and the column *Solution Space Reduction* gives the CPU time for solving the MILP formulation with solution space reduction.[8] No matter solution space reduction is performed or not, the CPU time of previous approach is less than one minute. Without solution space reduction, the CPU time of our approach is often in many hours; on the other hand, with solution space reduction, the CPU time of our approach is less than one hour.

[8] We use the heuristic approach proposed in [14] to reduce the solution space. Note that the heuristic approach proposed in [14] does not guarantee obtaining the optimal solution. Fortunately, as shown in [14], in each benchmark circuit, the optimal solution (for minimizing the required inserted delay) can be obtained after solution space reduction. In our experiments, we also find that, in each benchmark circuit, the optimal solution (for minimizing the number of inserted buffers) can be obtained after solution space reduction.

Table 2. First experiment results.

Circuit	Inserted Buffers		CPU Time (s)			
	[14]	Ours	Original Solution Space		Solution Space Reduction	
			[14]	Ours	[14]	Ours
S1269	46	34	4	1981	3	212
S1423	1	1	5	3	4	3
S3271	46	30	7	1185	5	154
S3384	44	40	11	23045	6	1294
S4863	241	189	12	16252	8	1191
S6669	636	511	26	37243	9	1688
S15850	21	15	35	28980	8	1468
S35932	320	320	40	31419	11	1357
S38417	2	2	31	196	8	26

Table 3. Second experiment results.

Circuit	Inserted Buffers	CPU Time (s)	
		Original Solution Space	Solution Space Reduction
S1269	115	2327	642
S1423	2	6	3
S3271	94	1421	524
S3384	167	25102	2265
S4863	421	23250	2252
S6669	1278	40845	4908
S15850	54	34086	2401
S35932	904	38497	3687
S38417	5	528	40

In the second experiment, we assume that the cell library has only 6 buffer types and their delays are 0.01 ns, 0.05 ns, 0.10 ns, 0.25 ns, 0.50 ns, and 1.00 ns, respectively (note that the minimum resolution of delay value is 0.01 ns). Table 3 tabulates our results on the second experiment. Since the previous approach [14] is not applicable to a limited number of buffer types, we only report the results of our approach. It should be mentioned that our approach guarantees minimizing the number of inserted buffers. Besides, we have the following two observations on the CPU time.

(1) Without solution space reduction, the CPU time of our approach is often in many hours; on the other hand, with solution space reduction, the CPU time of our approach is less than two hours.
(2) Compared with the CPU time of our approach on the first experiment, in which the number of buffer types is unlimited, the CPU time of our approach on the second experiment does not increase too much.

7. CONCLUSIONS

In this paper, we propose an MILP approach to minimize the number of inserted

buffers for the simultaneous application of clock skew scheduling and delay insertion (under the lower bound of sequential timing optimization and the lower bound of required inserted delay). Note that our approach guarantees obtaining the optimal solution. Experimental data show that, in most benchmark circuits, our approach can greatly reduce the number of inserted buffers.

Our approach is the first work to consider the minimization of the number of inserted buffers. Compared with the previous work [14], our work has two main contributions. First, our approach guarantees minimizing the number of inserted buffers. Second, our approach is applicable to a limited number of buffer types.

Our approach can be used as the post-processing approach for the previous work [14]. However, since solving the MILP formulation is an NP-hard problem, our approach may be very time-consuming for large circuits. Fortunately, after solution space reduction is performed, the optimal solution of each benchmark circuit can be obtained within an acceptable CPU time. Our future work will be to develop a good polynomial time complexity heuristic algorithm for large circuits.

REFERENCES

1. J. P. Fishburn, "A depth-decreasing heuristic for combinational logic," in *Proceedings of IEEE/ACM Design Automation Conference*, 1990, pp. 361-364.
2. J. P. Fishburn, "LATTIS: An iterative speedup heuristic for mapped logic," in *Proceedings of IEEE/ACM Design Automation Conference*, 1992, pp. 488-491.
3. N. Maheshwari and S. S. Sapatnekar, *Timing Analysis and Optimization of Sequential Circuits*, Kluwer Academic Publishers, Boston, 1999.
4. I. S. Kourtev and E. G. Friedman, *Timing Optimization through Clock Skew Scheduling*, Kluwer Academic Publishers, Boston, 2000.
5. M. C. Papaefthymiou, "Understanding retiming through maximum average-delay cycles," *Mathematical Systems Theory*, Vol. 27, 1994, pp. 65-84.
6. J. P. Fishburn, "Clock skew optimization," *IEEE Transactions on Computers*, Vol. 39, 1990, pp. 945-951.
7. R. B. Deokar and S. S. Sapatnekar, "A graph-theoretic approach to clock skew optimization," in *Proceedings of IEEE International Symposium on Circuits and Systems*, Vol. 1, 1994, pp. 407-410.
8. C. Albrecht, B. Korte, J. Schietke, and J. Vygen, "Cycle time and slack optimization for VLSI chips," in *Proceedings of IEEE International Conference on Computer Aided Design*, 1999, pp. 232-238.
9. N. V. Shenoy, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "Minimum padding to satisfy short path constraints," in *Proceedings of IEEE International Conference on Computer Aided Design*, 1993, pp. 156-161.
10. S. H. Huang and Y. T. Nieh, "Clock period minimization of non-zero clock skew circuits," in *Proceedings of IEEE International Conference on Computer Aided Design*, 2003, pp. 809-812.
11. B. Taskin and I. S. Kourtev, "Delay insertion method in clock skew scheduling," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, Vol. 25, 2006, pp. 651-663.

12. S. H. Huang and Y. T. Nieh, "Synthesis of nonzero clock skew circuits," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, Vol. 25, 2006, pp. 961-976.
13. S. H. Huang and Y. T. Nieh, "Clock skew scheduling with race conditions considered," *ACM Transactions on Design Automation of Electronic Systems*, Vol. 12, Article 45, 2007.
14. S. H. Huang, C. H. Cheng, C. M. Chang, and Y. T. Nieh, "Clock period minimization with minimum delay insertion," in *Proceedings of IEEE/ACM Design Automation Conference*, 2007, pp. 970-975.
15. S. H. Huang, G. Y. Jhuo, and W. L. Huang, "Minimum buffer insertions for clock period minimization," in *Proceedings of IEEE International Symposium on Computer, Communication, Control and Automation*, 2010, pp. 426-429.

Shih-Hsu Huang (黃世旭) received the B.S. degree in Computer Science and Information Engineering from National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 1989, the M.S. degree in Computer Science from National Tsing Hua University, Hsinchu, in 1991, and the Ph.D. degree in Computer Science and Information Engineering from National Taiwan University, Taipei, Taiwan, in 1995. From 1995 to 2000, he was with Computer and Communications Research Laboratories, Industrial Technology Research Institute, Hsinchu, rising to the position of deputy manager of IC design department, responsible for the design of high performance IC's. In 2000, he joined the Department of Electronic Engineering, Chung Yuan Christian University, Chungli, Taiwan, as a faculty member, where he is currently a Full Professor. He has served on technical program committee for several conferences, such as Asia and South Pacific Design Automation Conference (ASPDAC) and VLSI Design/CAD Symposium of Taiwan. His research interests include high-level synthesis, sequential optimization, clock tree synthesis, and physical design.

Guan-Yu Jhuo (卓冠宇) received the B.S. degree in Electronic Engineering from Chun Yuan Christian University, Chungli, Taiwan, R.O.C., in 2008, and the M.S. degree in Electronic Engineering from Chung Yuan Christian University, Chungli, in 2010. His research interests include timing optimization and sequential optimization.

Wei-Lun Huang (黃偉倫) received the B.S. degree in Electrical Engineering from Chun Yuan Christian University, Chungli, Taiwan, R.O.C., in 2007, and the M.S. degree in Electronic Engineering from Chung Yuan Christian University, Chungli, in 2010. His research interests include clock tree synthesis and low power design.