The 2004 IEEE Asia-Pacific Conference on
Circuits and Systems, December 6-9, 2004

# CLOCK PERIOD MINIMIZATION METHOD OF SEMI-SYNCHRONOUS CIRCUITS BY DELAY INSERTION

*Yukihide KOHIRA*†, *Atsushi TAKAHASHI*†

†Department of Communications and Integrated Systems, Tokyo Institute of Technology
2-12-1-S3-58 Ookayama, Meguro-ku, Tokyo 152-8552 Japan
TEL:+81-3-5734-2665 FAX:+81-3-5734-2902
Email:†{kohira, atsushi}@lab.ss.titech.ac.jp

## ABSTRACT

The minimum clock period in semi-synchronous framework might be reduced if delays are increased by delay insertion. In this paper, we propose a delay insertion algorithm to reduce the minimum clock period in semi-synchronous framework. We show that the proposed algorithm achieves the minimum clock period in semi-synchronous framework by delay insertion if the delay of each element is unique. Experiments show that the amount of inserting delay and computational time are smaller than the conventional algorithm.

## 1. INTRODUCTION

The semiconductor manufacturing process technology has improved the scale, speed and power consumption of LSI circuits. However, increasing ratio of routing delay in signal propagation delay makes the simultaneous clock distribution to every register difficult. The increases of size and power consumption of a clock distribution circuit have become serious issues in conventional synchronous circuits. While, a semi-synchronous circuit, in which the clock is assumed to be distributed periodically to each individual register though not necessarily to all registers simultaneously, is expected to give an essential solution. By using semi-synchronous framework, the improvements of clock frequency, clock distribution circuit size, peak power consumption, and etc. are expected to be achieved.

In recent studies of semi-synchronous circuit, clock scheduling algorithms [3] and clock distribution circuit synthesis algorithms [2] for given logic circuits were proposed. However given logic circuits are synthesized for complete-synchronous framework. In order to improve the clock period in complete-synchronous framework, the circuits are synthesized so that the maximum delay between registers is as small as possible. While, in semi-synchronous framework, the clock period might not be reduced even if the maximum delay is reduced. So the optimization of circuit synthesis that takes semi-synchronous framework into account must be investigated.

In this paper, we propose a delay insertion algorithm to reduce the minimum clock period in semi-synchronous framework with less amount of inserting delay. Delay insertion is realized by gate-sizing, wire sizing, buffer insertion, and etc. Since the inserting delay is small, the improved circuits are obtained by small modifications. First, the proposed algorithm determines a clock schedule that achieves the minimum clock period by ignoring some constraints. Second, the algorithm inserts delays to recover ignored constraints according to the delay-slack and delay-demand of the determined clock schedule. We show that the proposed algorithm achieves the minimum clock period by delay insertion if the delay of each element in the circuit is unique (That is, maximum delay = minimum delay for each element). Experiments show that the amount of inserting delay and computational time are smaller than the conventional algorithm [6].

## 2. PRELIMINARIES

In this paper, we consider a circuit consisting of registers and gates, and wires connecting them. We refer to them as elements. A circuit is represented by the *circuit graph* $G = (V_g, E_g)$, where $V_g$ is the vertex set corresponding to elements in the circuit and $E_g$ is the directed edge set corresponding to signal propagations in the circuit. In this paper, we assume each element has an unique delay. Let $d(v)$ be the weight of $v \in V_g$ which corresponds to the delay of corresponding element. Let $d(e) = d(u)$, where $e = (u, v) \in E_g$ $(u, v \in V_g)$. This means that an edge weight is equal to the weight of the head vertex of the edge. Let $V_r$ be a register set. Necessarily, the register set is a subset of $V_g$. An example of the circuit graph is shown in Fig. 1. In Fig. 1, $\{a, b, c, d\}$ is the register set, and the figure in each vertex represents its weight.

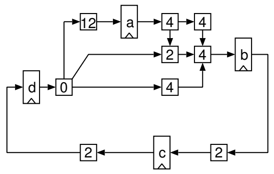

**Fig. 1**. Circuit graph $G$.

In semi-synchronous circuits, the clock input timing of a register may be different from other registers. The *clock timing* $S(r)$ of register $r$ is defined as the difference in clock arrival time between $r$ and an arbitrary chosen reference register.

A circuit works correctly with clock period $T$ if the following two types of constraints are satisfied for every register pair with signal propagations [1].

**Setup (No-Zero-Clocking) Constraints**

$$S(r_i) - S(r_j) \leq T - d_{\max}(r_i, r_j)$$

0-7803-8660-4/04/$20.00 ©2004 IEEE

**Hold (No-Double-Clocking) Constraints**

$$S(r_j) - S(r_i) \leq d_{\min}(r_i, r_j)$$

where $d_{\max}(r_i, r_j)$ is the maximum delay and $d_{\min}(r_i, r_j)$ is the minimum delay from register $r_i$ to $r_j$ (Fig. 2).

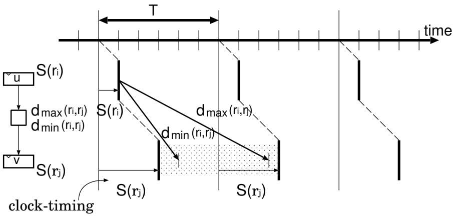

**Fig. 2**. Timing chart.

Since complete-synchronous circuits have the premise that a clock ticks all the register simultaneously, the clock period must be larger than the maximum delay between registers. On the other hand, semi-synchronous circuits can work correctly with the clock period which is smaller than the maximum delay between registers, if all the register pair with signal path satisfies two types of constraints.

Let $T_S(G)$ be the minimum clock period in semi-synchronous framework under the assumption that clock can be inputted to each register at an arbitrary timing. Hereafter, we simply call $T_S(G)$ the *minimum clock period.* $T_S(G)$ is determined by the *constraint graph* $G_T(V_r, A_r, w)$ defined as follows. $V_r$ is the vertex set of the constraint graph corresponding to registers in the circuit. $A_r$ is the directed edge set of the constraint graph corresponding to two types of constraints. $w$ is the weight function of edge. An edge from $r_i$ to $r_j$ with $w(r_i, r_j) = d_{\min}(r_i, r_j)$, called the D-edge, corresponds to the Hold constraint, and an edge from $r_j$ to $r_i$ with $w(r_j, r_i) = T - d_{\max}(r_i, r_j)$, called the Z-edge, corresponds to the Setup constraint. Let $G_{T=t}(V_r, A_r, w)$ be the constraint graph when the clock period $T$ is $t$.

Let the weight of a directed cycle be the sum of edge weights on the directed cycle in the constraint graph. We refer to the cycle whose weight is 0 and negative as *zero-cycle* and *negative-cycle*, respectively.

**Theorem 1 ([1],[4])** $T_S(G)$ is the minimum $t$ such that there is no negative-cycle in the constraint graph $G_{T=t}(V_r, A_r, w)$.

For example, the delay from register $d$ to $a$ in $G$ in Fig. 1 is 12 which is the maximum delay between registers. So the minimum clock period in complete-synchronous framework is 12. The constraint graph of the circuit when the clock period is 9 is shown in the left of Fig. 3. The cycle $(a, d, b)$ is a zero-cycle, so the minimum period $T_S(G)$ in semi-synchronous framework is 9. The semi-synchronous circuit that achieves the clock period 9 is shown in the right of Fig. 3.

Delay insertion to a circuit is represented by replacement of an edge by a series of two edges with a positive weight vertex. The circuit $G'$ obtained by inserting delays to $G$ in Fig. 1 is shown in the left of Fig. 4. In the constraint graph $G_T(V_r, A_r, w')$ of $G'$, edge wights are changed from $G_T(V_r, A_r, w)$ of $G$ according to the delay insertion. In $G_{T=7}(V_r, A_r, w')$ of $G'$ shown in the right of Fig. 4, the cycle $(a, d, c, b)$ and $(a, d, b)$ are zero-cycles when the clock period is 7. In this case, the minimum clock period is improved by delay insertion.

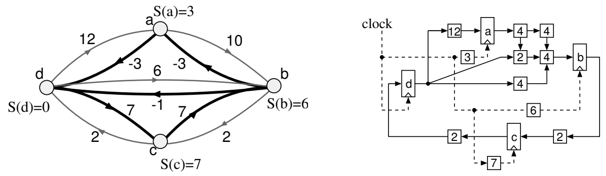

**Fig. 3**. Constraint graph $G_{T=9}(V_r, A_r, w)$ and the corresponding semi-synchronous circuit.

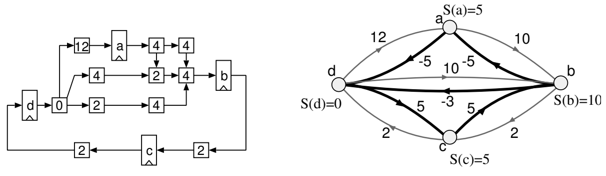

**Fig. 4**. Circuit graph $G'$ and constraint graph $G_{T=7}(V_r, A_r, w')$ after delay insertion to $G$ in Fig. 1.

## 3. PREVIOUS ALGORITHM

### 3.1. A lower bound of the clock period

If a circuit does not work correctly with a clock period $t$, all the negative-cycles in the constraint graph should be eliminated so that the circuit works correctly. Edge weights should be increased in order to eliminate all the negative-cycles. Since D-edge weight is $d_{\min}$ and Z-edge weight is $T - d_{\max}$, we should increase minimum delays or reduce maximum delays so that edge weights are increased.

In semi-synchronous circuits, the lower bound of the clock period that can be achieved by delay insertion is equal to the *maximum delay-to-register ratio* $T_B(G)$.

**Definition 1 ([6])** The maximum delay-to-register ratio is defined

$$T_B(G) = \max_{L \in \text{all cycles in } G} \frac{D(L)}{N(L)},$$

where $N(L)$ is the number of registers in directed cycle $L$ and $D(L)$ is the weight of $L$.

We refer to the maximum delay-to-register ratio $T_B(G)$ as the *minimum clock period by delay insertion.*

In complete-synchronous framework, $T_B(G)$ can be achieved if an arbitrary amount of retiming is allowed. However, the number of register might be increased, and gate decompositions might be required.

**Theorem 2 ([6])** If $T_S(G) > T_B(G)$, every zero-cycle in $G_{T=T_S(G)}(V_r, A_r, w)$ contains at least one D-edge.

Therefore, the minimum clock period is improved by delay insertion to the circuit if $T_S(G) > T_B(G)$.

As logic circuit improvement methods for semi-synchronous circuits, the delay insertion method was proposed in [6] and the gate sizing method was proposed in [5]. The algorithm in [6] achieves the minimum clock period by delay insertion, but it cannot be applied to large circuits since it inserts too much delay and takes too much time. While, the algorithm in [5] does not necessarily achieve the minimum clock period by delay insertion.

### 3.2. Delay-slack

The delay insertion must not increase the maximum delay-to-register ratio. $T_B(G)$ does not change if $D(L)$ of any cycle $L$ in $G$ is at most $T_B(G) \cdot N(L)$. We define the *delay-slack* for each edge that represents the margin within which the delay insertion keeps the minimum clock period by delay insertion.

**Definition 2 (delay-slack)** For a directed cycle $L$ in $G$, the *cycle-slack* is defined as

$$\text{cycle-slack} = T_B(G) \cdot N(L) - D(L).$$

The delay-slack of an edge $(u, v)$ in $G$ is the minimum cycle-slack over all cycles that contain $(u, v)$.

The delay insertion to $(u, v)$ less than or equal to the delay-slack of $(u, v)$ keeps the minimum clock period by delay insertion.

In the algorithm in [6], the delay with the same amount of delay-slack of an edge is inserted to the circuit when the edge belongs to the minimum delay path from $r_i$ to $r_j$ such that $(r_i, r_j)$ is a D-edge that is a part of negative-cycle in $G_{T=T_B(G)}(V_r, A_r, w)$ (Fig. 5). The delay-slack of each edge is recalculated after each delay insertion.

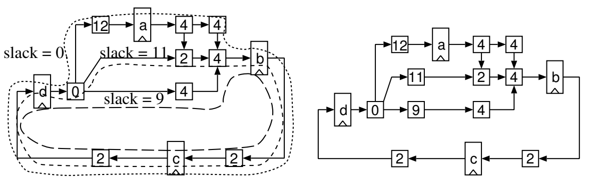

**Fig. 5**. Cycle-slacks of $G$ in Fig. 1 and the circuit $G'$ obtained by the algorithm in [6] ($T_B(G) = 7$).

## 4. PROPOSED ALGORITHM

We propose a delay insertion algorithm that the clock timing of each register is determined by Setup constraints before delay insertion, and delays are inserted into the circuit to satisfy Hold constraints. We show that a circuit which works correctly when the clock period is the minimum clock period by delay insertion is obtained in short time.

### 4.1. Scheduling

We determine the clock timing of each register from the constraint graph consisting of only Z-edges with the clock period $T = T_B(G)$ by clock-scheduling algorithm [4].

### 4.2. Delay insertion

The clock schedule from Setup constraints may not satisfy Hold constraints. So delays are inserted in order to satisfy Hold constraints.

For any edge $e \in E_g$, let $break(e)$ be the time when the earliest signal arrives at $e$, $complete(e)$ be the time when the latest signal arrives at $e$, $hold(e)$ be the earliest possible time when the signal may arrive at $e$ in order to keep Hold constraints, and $setup(e)$ be the latest possible time when the signal may arrive at $e$ in order to keep Setup constraints. Moreover, let *delay-slack* $slack(e)$ be $setup(e) - complete(e)$, that means the amount of delay which can be inserted, and *delay-demand* $demand(e)$ be $hold(e) - break(e)$, that means the amount of delay which must be inserted (Fig. 6).

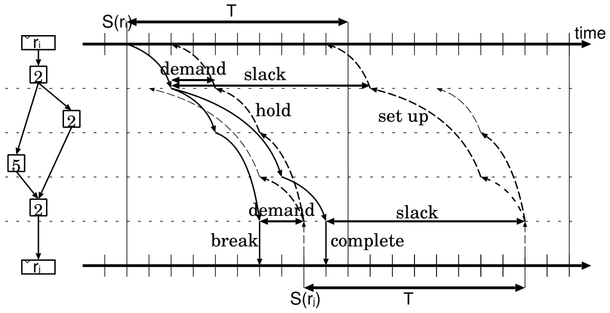

**Fig. 6**. $break$,$complete$,$hold$,$setup$,$slack$,$demand$.

**Definition 3** Let $e = (u, v) \in E_g$. If $u \in V_r$, we define

$$\begin{aligned} break(e) &= S(u) + d(e), \\ complete(e) &= S(u) + d(e), \end{aligned}$$

otherwise,

$$\begin{aligned} break(e) &= \min_{e' \in Fi(e)} \{break(e') + d(e)\}, \\ complete(e) &= \max_{e' \in Fi(e)} \{complete(e') + d(e)\}, \end{aligned}$$

where $Fi(e)$ is the edge set whose tail is $u$.

If $v \in V_r$, we define

$$\begin{aligned} hold(e) &= S(v), \\ setup(e) &= S(v) + T, \end{aligned}$$

otherwise,

$$\begin{aligned} hold(e) &= \max_{e' \in Fo(e)} \{hold(e') - d(e')\}, \\ setup(e) &= \min_{e' \in Fo(e)} \{setup(e') - d(e')\}, \end{aligned}$$

where $Fo(e)$ is the edge set whose head is $v$.

Moreover, we define

$$\begin{aligned} slack(e) &= setup(e) - complete(e), \\ demand(e) &= hold(e) - break(e). \end{aligned}$$

The proofs of the following two theorems are omitted for space.

**Theorem 3** The clock schedule with clock period $T$ satisfies Setup constraints if and only if $slack(e) \geq 0$ for all edges $e$ in $E_g$.

**Theorem 4** The clock schedule with clock period $T$ satisfies Hold constraints if and only if $demand(e) \leq 0$ for all edges $e$ in $E_g$.

If Hold constraints are violated, the delay equal to $\min\{slack(e), demand(e)\}$ is inserted in an edge $e$. Then, Setup constraints are not violated while Hold constraints are relaxed.

### 4.3. Algorithm

We propose a delay insertion algorithm as follows.

Inputs : circuit graph $G$
Outputs : circuit graph $G'$ after delay insertion

**Table 1**. Results.

| Circuit | Num. of gates | $\lvert V_r \rvert$ | $\lvert A_r \rvert$ | clock period CS | SS | Fin | Algorithm in [6] delay | Time [s] | Proposed algorithm delay (%) | Time [s] (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| s298 | 119 | 15 | 84 | 9 | 6.00 | 5.34 | 71 | 0.17 | 14 (19.7) | 0.08 (47.1) |
| s344 | 160 | 16 | 86 | 19 | 17.00 | 10.00 | 451 | 1.30 | 126 (27.9) | 0.23 (17.7) |
| s349 | 161 | 16 | 86 | 19 | 17.00 | 10.00 | 451 | 1.14 | 126 (27.9) | 0.24 (21.1) |
| s444 | 181 | 22 | 173 | 11 | 7.00 | 6.59 | 70 | 0.57 | 19 (27.1) | 0.16 (28.1) |
| s526 | 193 | 22 | 165 | 9 | 6.00 | 5.50 | 66 | 0.38 | 12 (18.1) | 0.18 (47.4) |
| s1423 | 657 | 75 | 1897 | 59 | 54.00 | 53.00 | 6172 | 217.72 | 3779 (61.2) | 5.90 ( 2.7) |
| average | — | — | — | — | — | — | — | — | — (30.6) | — (27.3) |

**Step 1** : Determining the clock timing of each register from the constraint graph consisting of Z-edges with the clock period $T = T_B(G)$.

**Step 2** : Until an edge $e$ that $demand(e) > 0$ exists, repeat the following. Insert delay with $\min\{slack(e), demand(e)\}$ to $e$. Recalculate delay-slack and delay-demand.

**Step 3** : Output circuit graph $G'$ after delay insertion and terminate.

By the proposed algorithm, delay is inserted to an edge whose delay-demand is positive. So we need to show that at least one edge whose delay-slack is positive exists among edges whose delay-demand are positive.

**Theorem 5** If there is a register pair that violates a Hold constraint in a clock schedule satisfying Setup constraints, there is at least one edge whose delay-slack is positive among edges whose delay-demand are positive.

This proof is omitted for space.

By delay insertion, delay-slack and delay-demand of each edge never increase. If the delay is inserted to an edge $e$, then $slack(e) = 0$ or $demand(e) \leq 0$. So $e$ is never inserted delay again. Therefore, the delay insertion to edges is repeated at most the number of edges.

For example, the proposed algorithm is applied to $G$ in Fig. 1. Delay-slacks and delay-demands are determined as shown in Fig. 7. Then we obtain the circuit shown in Fig. 4.

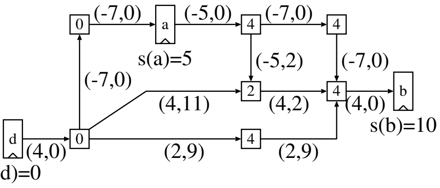

**Fig. 7**. Delay-slacks and delay-demands $(delay\text{-}demand, delay\text{-}slack)$ in $G$ in Fig. 1.

## 5. EXPERIMENTS

We implemented the algorithm in [6] and proposed algorithm in a PC with a 3.06GHz/512K Intel Pentium 4 CPU and 512MB RAM. We performed delay insertion on the LGSynth91 benchmark suite. In experiments, we assume that each gate has unit delay, and routing delays and register delays are zero. The clock period is reduced in 6 out of 24 circuits. The results of improved circuits are shown in Table 1. In Table 1, the minimum clock period in complete-synchronous framework is shown by CS, the minimum clock period in semi-synchronous framework by SS, the minimum clock period by delay insertion by Fin, the amount of inserted delay in previous algorithm and proposed algorithm by delay.

The amount of delay insertion is reduced by about 69.4% and computation time is reduced by about 72.7%.

## 6. CONCLUSION AND FUTURE WORKS

In this paper, we propose a clock period minimization algorithm of semi-synchronous circuits by delay insertion and compare it with the algorithm in [6] for the amount of delay insertion and computational time. We realize a lower bound of the clock period by less delay insertion than the algorithm in [6], because proposed algorithm inserts delay less than or equal to delay-slack.

As future works, we consider where delay is inserted and how the clock timing for each register is determined for the minimization of inserted delays and computation time. Finally, we want to propose a delay insertion algorithm for the real delay model (That is, maximum delay is not equal to minimum delay for each element).

## 7. REFERENCES

[1] J.P. Fishburn. Clock skew optimization. *IEEE Tranc. on Computers*, 39(7):945–951, 1990.

[2] S. Ishijima, T. Utsumi, T. Oto, and A. Takahashi. A semi-synchronous circuit design method by clock tree modification. *IEICE Transactions on Fundamentals*, E85-A(12):2596–2602, November 2002.

[3] M. Saitoh, M. Azuma, and A. Takahashi. A clustering based fast clock schedule algorithm for light clock-trees. *IEICE Transactions on Fundamentals*, E85-A(12):2756–2763, November 2002.

[4] A. Takahashi and Y. Kajitani. Performance and reliability driven clock scheduling of sequential logic circuits. In *ASP-DAC'97*, pages 37–43, 1997.

[5] T. Yasui, K. Kurokawa, M. Toyonaga, and A. Takahashi. A circuit optimization method by the register path modification in consideration of the range of feasible clock timing. In *DA Symposium 2002*, pages 259–264, 2002. In Japanese.

[6] T. Yoda and A. Takahashi. Clock period minimization of semi-synchronous circuits by gate-level delay insertion. *IEICE Transactions on Fundamentals*, E82-A(11):2383–2389, November 1999.