論文 / 著書情報
Article / Book Information

Title	A Semi-Synchronous Circuit Design Method by Clock Tree Modification
Authors	Seiichiro Ishijima, Tetsuaki Utsumi, Tomohiro Oto, Atsushi Takahashi
Citation	IEICE Trans. Fundamentals, Vol. E85-A, No. 12, pp. 2596-2602
Pub. date	2002, 12
URL	http://search.ieice.org/
Copyright	(c) 2002 Institute of Electronics, Information and Communication Engineers

PAPER Special Section on VLSI Design and CAD Algorithms

A Semi-Synchronous Circuit Design Method by Clock Tree Modification****

Seiichiro ISHIJIMA[†*], **Tetsuaki UTSUMI**[†**], **Tomohiro OTO**[†***], ***Nonmembers***, ***and*** **Atsushi TAKAHASHI**[†a)], ***Regular Member***

SUMMARY A circuit in which the clock is assumed to be distributed periodically to each individual register though not necessarily to all registers simultaneously, called a semi-synchronous circuit, is expected to achieve higher frequency or a smaller clock tree compared with an ordinary synchronous circuit, called a complete-synchronous circuit. In this paper, we propose a circuit design method that realizes a semi-synchronous circuit with higher frequency by modifying the clock tree of a complete-synchronous circuit. We confirm that the proposed method is easy to incorporate with current practical design environment by designing a four stage pipelined processor compatible with MIPS operation code. The obtained processor circuit is the first semi-synchronous circuit designed systematically with theoretical background.

key words: *semi-synchronous circuit, clock tree, MIPS processor*

1. Introduction

A complete-synchronous circuit, in which the clock is distributed periodically to all registers simultaneously, is a target for circuit synthesis in current practical design environment. In deep-sub-micron (DSM) era, the degradation in complete-synchronousness of circuit is inevitable, and the design margin is wasted to keep the differences from the complete-synchronousness small. In recent years, 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 was proposed and have been studied for practical use. By using the semi-synchronous framework, the improvements of many performance features, which are, for example, clock frequency, clock tree size and peak power consumption, are expected to be achieved. In fact, clock tree algorithms in the semi-synchronous framework proposed as bounded-skew [2]–[4], useful-skew [10], [11], associative-skew [1], and semi-synchronous [6], [7], [9] can reduce the clock tree size.

Manuscript received March 18, 2002.
Manuscript revised June 20, 2002.
Final manuscript received August 1, 2002.

[†]The authors are with the Graduate School of Science and Technology, Tokyo Institute of Technology, Tokyo, 152-8552 Japan.
[*]Presently, with NEC Electron Devices.
[**]Presently, with Toshiba Corporation.
[***]Presently, with Sony Computer Entertainment Inc.
a) E-mail: atsushi@lab.ss.titech.ac.jp

[****]The preliminary version was presented at [5].

One way to design a semi-synchronous circuit is as follows: the clock timings of registers are determined to improve the circuit performance by using the minimum and maximum delays between registers, and the clock timings of registers are realized by constructing a clock tree. However, since the detailed routing between circuit elements are usually completed after clock tree routing, the minimum and maximum delays between registers might be changed after constructing the clock tree and the timing violations might be caused by these changes. In order to prevent timing violations, the larger delay margin and/or the improvement of routing delay estimation before routing would be required. The larger delay margin is, the smaller the quantity of optimization is. The improvement of routing delay estimation is essential to achieve better circuit, though there is a limit to the improvement. Thus, it is hard to adopt the above way in designing a high performance semi-synchronous circuit.

In this paper, we propose a semi-synchronous circuit design method that matches current practical design environment. In the proposed method, first, a complete-synchronous circuit is designed by current design environment and the delay information is extracted by the obtained circuit layout. The clock tree in the obtained circuit is called a zero skew clock tree. Next, a clock schedule to improve the circuit performance is determined by using the information, and the zero skew clock tree is modified in order to achieve the clock schedule. Finally, the layout of a semi-synchronous circuit is obtained after few change orders caused by the modification of the zero skew clock tree are fixed.

In order to keep the validity of the delay information after the modification of the zero skew clock tree, the modification should not much affect the rest of the circuit. Since the effect of routing delay caused by the wire, especially for long wire, is large in DSM era, the changes of global routes of wires as well as the change of global placement will invalidate the delay information, that is, it makes the differences between the resultant actual delays and the used delay values larger. However, the effect of local changes is still small enough. Therefore, the modified clock tree is obtained by inserting clock buffers into the zero skew clock tree without changing the clock tree topology. Then, the change orders caused by the modification of the zero skew clock

tree are fixed by local modifications of placement and routing. Moreover, the clock schedule is determined so that the difference from the clock schedule defined by the zero skew clock tree is as small as possible to keep the modification small. The topology of the zero skew clock tree is also determined so that the estimated required modification is as small as possible. Of course, there is a tradeoff between the improvement of circuit performance and the amount of modification. The improvement of circuit performance is maximized until the required modifications are allowable.

The proposed method is applied to design a four stage pipelined processor compatible with MIPS operation code. The obtained processor circuit is the first semi-synchronous circuit designed systematically with theoretical background. Although the complete-synchronous circuit was changed into the semi-synchronous circuit in short time, the processor speed is improved more than 10%. We make sure that our proposed method matches to current design environment, and the circuit speed is improved.

2. Semi-Synchronous Circuit

In this paper, we consider a circuit with a single clock that consists of registers and combinatorial circuits among them. A circuit in which the clock is assumed to be distributed periodically to each individual register, though not necessarily to all registers simultaneously, is called a *semi-synchronous circuit*. The *clock timing* $s(v)$ of register v is the difference in clock arrival time between the register and an arbitrary chosen (perhaps hypothetical) reference register. A set of clock timing of all registers is called a *clock schedule*.

We assume the framework, called the *semi-synchronous framework*, that a circuit works correctly if the following two types of constraints are satisfied for every register pair with signal propagation.

Hold Constraints:

$$s(v) - s(u) \leq d_{\min}(u, v) - hold(v)$$

Setup Constraints:

$$s(u) - s(v) \leq T - (d_{\max}(u, v) + setup(v))$$

Here T is the clock period, $d_{\min}(u, v)$ ($d_{\max}(u, v)$) is the minimum (maximum) propagation delay from register u to register v through combinatorial circuits and $setup(v)$ ($hold(v)$) is the setup (hold) time of register v. The difference between the minimum and maximum propagation delay from a register to a register is caused by various sources, for example, the variety of paths between the registers, the difference between the rise and fall times of a gate, and the delay variation caused by manufacturing and environment. A clock schedule is said to be *feasible* if the above two types of constraints are satisfied for every register pair with signal propagation.

An example of semi-synchronous circuit is shown in Fig. 1. For simplicity, here we assume that $hold(v) = setup(v) = 0$ for each register v. In Fig. 1, the maximum propagation delay from register a to register b is 12 and it is the maximum propagation delay in this circuit. Therefore, if the clock timing of every register is the same, then the minimum clock period is 12. However, the clock is distributed to each register at different timing, that is, $s(a) = 3$, $s(b) = 6$, $s(c) = 7$, and $s(d) = 0$. This circuit works at clock period 9 since every constraint is satisfied when $T \geq 9$.

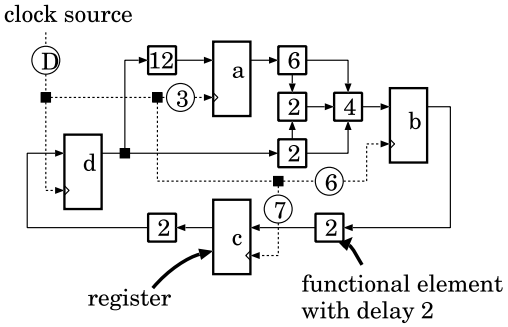


Fig. 1 Semi-synchronous circuit.

In a *complete-synchronous circuit*, the clock timing of registers is equal, that is, $s(v) = s(u)$ for any registers u and v. The clock tree in a complete-synchronous circuit is called a *zero skew clock tree*. Although it is impossible to obtain an ideal zero skew clock tree due to design, manufacturing, and/or environment, we call a clock tree designed under the complete-synchronous framework a zero skew clock tree, too, in order to distinguish it from a clock tree designed under the semi-synchronous framework. Similarly, a circuit designed under the complete-synchronous framework is called a complete-synchronous circuit, too.

Under the assumption that the clock timing $s(v)$, $s(u)$ can be controlled, for given the delay $d_{\max}(u, v)$, $d_{\min}(u, v)$, setup time $setup(v)$ and hold time $hold(v)$, the minimum clock period in the semi-synchronous framework $T_{S\min}$ can be determined in polynomial time [8]. The minimum clock period in the semi-synchronous framework $T_{S\min}$ is smaller than or equals to that in the complete-synchronous framework $T_{C\min}$. When a clock period is larger than or equals to $T_{S\min}$, a feasible clock schedule exists. When a clock period T ($\geq T_{S\min}$) is given, the range of clock timing of each register is determined so that a feasible clock schedule is obtained whenever the clock timing of every register is chosen from the range of clock timing of the register. The range of clock timing of each register should be kept large enough to be realized by constructing a clock tree and to be tolerant to delay variations. The larger clock period T is, the larger the range of clock timing is.

In a zero skew tree, the clock timings of registers are the same, say 0. When T is larger than or equals to $T_{C\min}$, the range of clock timing of every register

can be determined so that it contains 0. When T is smaller than $T_{C\min}$, the ranges of clock timings of some registers can not be determined so that it contains 0. Since we construct a clock tree in the semi-synchronous framework by modifying a zero skew tree, the modification of clock timing of a register is not required if the range of clock timing of the register contains 0.

3. Semi-Synchronous Circuit Design Flow

Figure 2 shows the proposed design flow. The phases in shaded rectangles are the features in the proposed design method. The other phases are the same as the current design flow for complete-synchronous circuits.

In the proposed method, first, a layout of a complete-synchronous circuit is designed, which corresponds to the phases from RTL Description through Routing. We do not add any restriction in a layout method except a construction of a clock tree. In Clock Tree Topology Generation, the topology of a clock tree is determined so that the modification of the clock tree in the following phases is as small as possible using the information obtained in Clock Scheduling (Preliminary). In Clock Scheduling (Final), the clock schedule is calculated from the delay extracted from the layout of a complete-synchronous circuit. When the clock schedule is realized by the modification of the clock tree in Clock Tree Modification, a semi-synchronous circuit with higher clock frequency is obtained. The obtained semi-synchronous circuit layout is verified in Verification.

In the following, we explain a way to the clock tree modification, and feature phases in the proposed design flow.

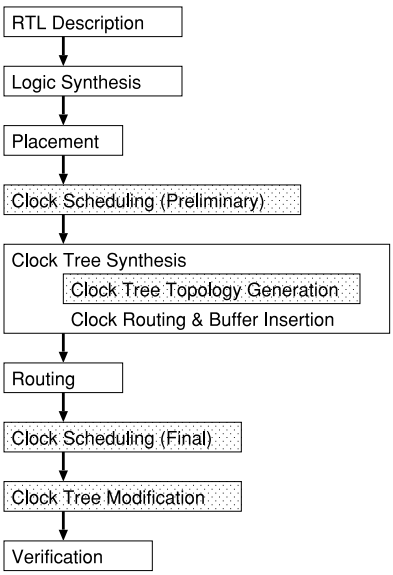


Fig. 2 Semi-synchronous circuit design flow.

3.1 Basic Techniques in Clock Tree Modification

A clock schedule is realized by modifying a zero skew clock tree. In order to realize the clock schedule, if the clock tree topology is modified, the difference between the delay used for clock scheduling and the actual delay will be increased. If the buffers are removed from the clock tree, the clock tree will be lacking in driving ability. In such cases, the circuit may not work correctly. Thus, the clock schedule is realized only by inserting buffers into the clock tree.

Assume that the delay from clock source to buffer B in the clock tree in Fig. 3 is requested to be increased to realize a clock schedule. Then the clock tree is modified by inserting a buffer into the clock tree just before B as shown in Fig. 4. The input capacitance of D is set equal to that of B as shown in Fig. 4. In the modified clock tree, the change of length of wire W is small and the length of wire between D and B is negligible. Thus the changes of gate delay of A and routing delay of W are small since the change of load capacitance is small. Therefore, the increase in delay to B is caused by the gate delay of D. D can consist of more than one buffer. The number of buffers and the size of each buffer to be inserted are chosen so that the increased delay is fit within the requested range.

The increase in delay is realized by inserting buffers but the reduction in delay is not always realized by inserting buffers. Therefore, setting the clock timing of

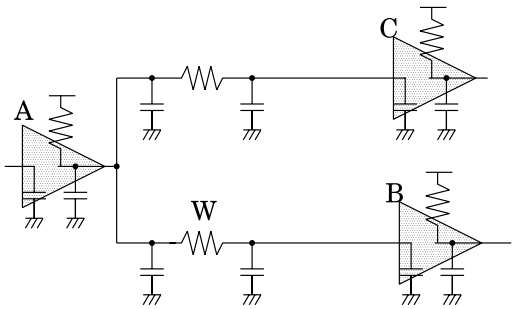


Fig. 3 A part of clock tree.

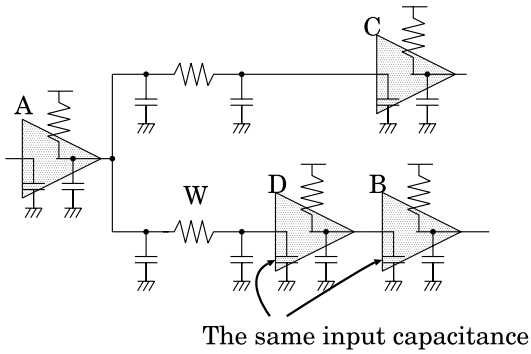


Fig. 4 After clock buffer insertion.

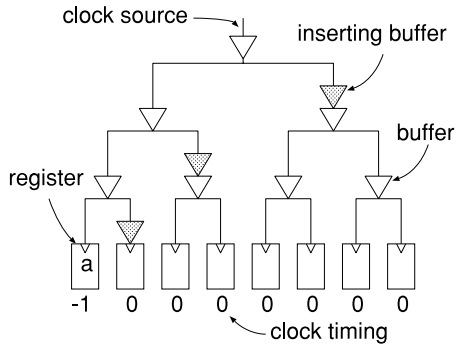


Fig. 5 Setting clock timing smaller by buffer insertion.

a register smaller is realized by putting off the timing of the other registers. In such cases, the buffers have to be inserted into every opposite branch on the path from the clock source to the register. In Fig. 5, the clock timing of register a is advanced by one where the delay to a register is the number of buffers on the path from the clock source to the register.

3.2 Clock Tree Topology Generation Phase

In designing a complete-synchronous circuit, a zero skew clock tree is constructed. In the clock tree modification phase, the obtained zero skew clock tree is modified to realize a clock schedule. In the clock tree modification, setting the clock timing of a register smaller requires usually much cost as compared with setting it larger. However, the required cost is relatively small if the register is near the clock source in the clock tree topology. Thus, we predict the registers which will be changed clock timing smaller in the clock tree modification phase, and construct the clock tree topology so that these registers are near to the clock source and so that they get together in the clock tree topology.

The prediction is done by using the clock schedule obtained using the gate delay information extracted from the netlist in the preliminary clock scheduling phase. For a register with negative clock timing in the preliminary clock schedule, it is expected that the clock timing of it is also negative in the final clock schedule although the final clock schedule is determined by using the delay information extracted from the layout.

3.3 Clock Scheduling Phase

The clock schedule such that the sum of differences between the due clock timing and the target clock timing of registers is minimum can be obtained by the algorithm in [12] that solves the target scheduling problem. We use this algorithm to obtain clock schedules in two phases.

Target scheduling problem
Input:

- the minimum (maximum) propagation delay between registers $d_{\min}(u,v)$ ($d_{\max}(u,v)$)
- setup (hold) time $setup(v)$ ($hold(v)$)
- target of clock period T
- target of clock timing $o(v)$

Output: clock timing $s(v)$
Constraint:

- Setup constraint
- Hold constraint

Objective: minimize $\sum(s(v) - o(v))$

In the preliminary clock scheduling phase, the minimum (maximum) propagation delays between registers which are estimated by the gate delays extracted from the netlist are used. The target of clock timing $o(v)$ is set to 0 for each register v. The registers of which clock timing are negative in a clock schedule with a shorter clock period are focused in the clock tree topology generation phase.

In the final clock scheduling phase, the minimum (maximum) propagation delays between registers which are extracted from the layout of a complete-synchronous circuit are used. The number of buffers inserted into clock tree in order to realize a clock schedule is expected to be small if the clock schedule to be realized is similar to the clock schedule defined by the zero skew clock tree. Thus, the target of clock timing $o(v)$ is set to the clock timing estimated from the zero skew clock tree.

The amount of required modification depends on the range of clock timing of each register. The wider the range of clock timing is, the easier realizing the clock timing is. The range depends on the target clock period. The target clock period is determined so that required modifications are allowable.

3.4 Clock Tree Modification Phase

The clock tree in the layout is modified to change the complete-synchronous circuit into a semi-synchronous circuit. As explained in Sect. 3.1, the clock tree is modified only by inserting the buffers into the clock tree.

The netlist of clock tree is modified according to the clock schedule obtained in the final clock scheduling phase. The clock timing of each register has a range. The delay to be realized has an inevitable range, too. The set of buffers is inserted so that the range of delay is fit in the range of the clock timing.

If there is no area to insert the buffers into layout, the obstacles are shifted to make the room for the buffers. After buffers are inserted into layout, the modification of routing is performed. These layout modification can be done by the modification function of a layout tool. Notice that the changes of signal propagation delays between registers caused by this modification are small since there is no global change in placement and routing.

3.5 Verification Phase

If the clock is distributed to each register within the range determined in the final clock scheduling phase, the semi-synchronous circuit works correctly. But the error of delay estimation or the change of signal delay by modification of layout may cause timing violations. The circuit is verified using the delay information extracted from the modified layout. Whether a semi-synchronous circuit works correctly is proved by examining whether hold constraint of all register pairs are satisfied. The hold constraint is independent of the clock period, so if the constraint is not satisfied, the circuit does not work regardless of the clock period. The hold constraint is checked by comparing the maximum delay from the clock source to the clock input pin of register v with the minimum delay from the clock source to the data input pin of register v via other register. When the former is smaller than the latter, the hold constraint is satisfied.

The minimum clock period is calculated with the setup constraint. The setup constraint is satisfied if the clock period is large. The minimum clock period is the maximum of the differences between the minimum delay from the clock source to the clock input pin of register v and the maximum delay from the clock source to the data input pin of register v.

4. Experiments

We designed a four stage pipelined processor compatible with MIPS operation code, from RTL source, by the proposed method. The layout of the complete-synchronous circuit was designed by three of us in three months. While, it takes about one hour on converting the layout of a complete-synchronous circuit into that of a semi-synchronous circuit. The semi-synchronous processor core and the complete-synchronous processor core are integrated into a chip to eliminate the influence on delay caused by the process variations. Figure 6 shows the outline of the chip we designed.

The process used is ROHM 0.35 μm CMOS, 3 metal layers. The standard cell library of VDEC is used. The chip size is 4.9 mm square. The RTL sources and netlists are written in Verilog. Design Compiler of Synopsys for logic synthesis, Apollo of Avanti! for layout and Verilog-XL of Cadence for simulation are used.

In the logic synthesis phase, we did not optimize the complete-synchronous circuit with respect to the clock period. The clock period is assumed to be reduced in the semi-synchronous framework even after the circuit is optimized with respect to the clock period in the complete-synchronous framework, since the optimization in the complete-synchronous framework does not consider the semi-synchronous framework.

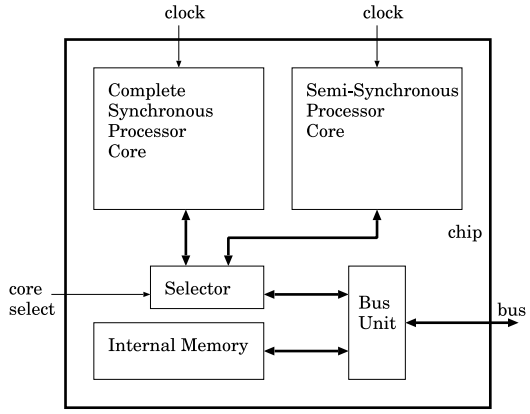


Fig. 6 Outline of chip.

Table 1 Maximum delay and minimum clock period.

maximum delay [ns]	minimum clock period [ns] (%)
21.21	18.89 (89.1)
18.34	15.32 (83.5)
14.21	11.71 (82.4)
12.70	10.83 (85.3)

Table 2 Clock period and required modification.

clock period [ns]	#register	change [ns]
13.58	12	4.87
13.81	8	3.84
14.31	4	2.51
16.66	0	0.00

To confirm this assumption valid, the circuit used in this experiment with a single processor core is synthesized for various target clock period in the complete-synchronous framework. The minimum clock period in the complete-synchronous framework "maximum delay" and that in the semi-synchronous framework "minimum clock period" using gate level delay information are shown in Table 1.

As a result of logic synthesis for a fabricated chip, the processor core consists of 7456 cells and 1478 registers. The peripheral circuit including internal memory contains 2249 registers. Since the clock timing of the registers in the peripheral circuit is constrained to be equal, these registers are regarded as one register in clock scheduling phases.

The minimum clock period in the complete-synchronous framework and that in the semi-synchronous framework using gate level delay information are 14.33 [ns] and 12.73 [ns], respectively. The minimum clock period in the complete-synchronous framework and that in the semi-synchronous framework using delay information extracted from the layout are 16.66 [ns] and 13.58 [ns], respectively.

In the final clock scheduling phase, the trade-off between the clock period in the semi-synchronous framework and the amount of required timing modification is evaluated. In Table 2, "#register" and "change"

Table 3 Result of clock scheduling.

register name	range [ns]	timing [ns]
MAU	−1.90 ∼−1.61	−1.62
DCHU/MA_DATA_REG[0]	0.75 ∼ 1.13	0.77
DCHU/MA_DATA_REG[31]	0.07 ∼ 1.01	0.15
CU/OV_Q_REG	0.08 ∼ 0.74	0.15

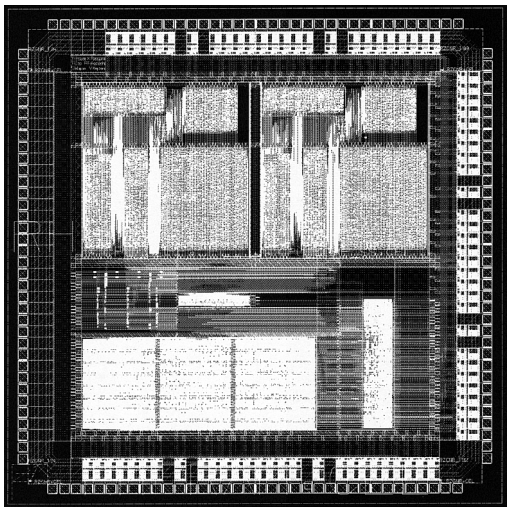


Fig. 7 Layout of chip.

represent the number of registers requested to change the clock timing and the sum of required changes in clock timing, respectively. The number of registers requested to change the clock timing differs in the clock period. The clock periods at which the number of requested registers is changed are listed in Table 2. Among these clock periods, we selected 14.31 [ns] as the target clock period of semi-synchronous circuit since the required modifications in the clock period shorter than it is not allowable, that is, layout tools can not complete the layout modification.

At the clock period 14.31 [ns], the clock timing of 4 registers have to be modified. The ranges of clock timings of these registers "range" and the adopted clock timings of them "timing" are shown in Table 3. The ranges of clock timing of the other registers contain 0, so the clock timings of these registers are not modified. The increase of delay by buffer insertion is estimated by the information of the cell library. The clock schedule is realized by inserting 17 buffers into the clock tree of the complete-synchronous circuit.

In Table 3, "MAU" corresponds to the registers in the peripheral circuit. In the clock tree topology generation phase, registers in "MAU" are collected under a subtree since the clock timing of "MAU" is also negative in the clock schedule obtained in the preliminary clock scheduling phase. Therefore, the negative clock timing of "MAU" is realized by inserting buffers into one branch in the clock tree.

Figure 7 shows the resultant layout. The sizes of processor cores are the same. We confirmed that the complete-synchronous circuit and the semi-synchronous circuit work correctly. In verification using delay information extracted from the modified layout, the minimum clock periods of the complete-synchronous circuit and that of semi-synchronous circuit are 16.46 [ns] and 14.21 [ns], respectively, The maximum delay in the complete-synchronous circuit is smaller than the maximum delay before layout modification since the distance to the peripheral circuit from the complete-synchronous processor core is smaller than that from the semi-synchronous processor core. The maximum delay before layout modification is caused by a path between the one processor core and the peripheral circuit. In verification using fabricated chips in which a critical part of the circit is activated, they are 16.0 [ns] and 13.4 [ns], respectively.

5. Conclusion

In this paper, we proposed semi-synchronous circuit design method that matches current practical design environment. In experiment, it takes about one hour on converting the layout of a complete-synchronous circuit into that of a semi-synchronous circuit. The designed semi-synchronous circuit works about 16% faster than the complete-synchronous circuit in simulation and about 19% faster in fabricated chip.

In designing the circuit, the target clock period is usually given in advance. The achieved clock period of a circuit obtained by our proposed method depends on the result of complete-synchronous circuit synthesis. It is not clear how to define the target clock period in complete-synchronous circuit synthesis in order to obtain a best circuit that archives the given target clock period in our proposed method. It might be required to design several times by changing the target clock period in complete-synchronous circuit synthesis. However, the clock scheduling using gate level delay information will reduce the time for designing the circuit that does not achieve the target clock period.

As a future work, we are going to improve our design method to make better use of the semi-synchronous framework.

Acknowledgments

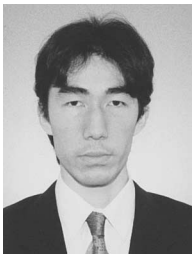
The authors are grateful to Professor Yoji Kajitani, the University of Kitakyushu, Dr. Kengo R. Azegami, Fujitsu Laboratories Ltd., for their helpful comments. This work is supported in part by Grant-in-Aid for Scientific Research, Grant-in Aid for Encouragement of Young Scientists. This work is part of a project of CAD21 at Tokyo Institute of Technology. The VLSI chip in this study is designed with Avant!, Synopsys and Cadence CAD tools through the chip fabrication program of VLSI Design and Education Center (VDEC),

the University of Tokyo, and has been fabricated in this program with the collaboration by Rohm Corporation and Toppan Printing Corporation.

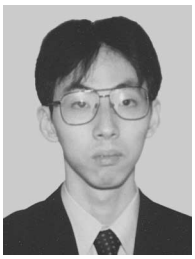
References

[1] Y. Chen, A.B. Kahng, G. Qu, and A. Zelikovsky, "The associative-skew clock routing problem," Proc. International Conference on Computer-Aided-Design (ICCAD), pp.168–172, 1999.
[2] J. Cong, A.B. Kahng, C.K. Koh, and C.W.A. Tsao, "Bounded-skew clock and Steiner routing under Elmore delay," Proc. International Conference on Computer-Aided-Design (ICCAD), pp.66–71, 1995.
[3] J. Cong and C.K. Koh, "Minimum-cost bounded-skew clock routing," Proc. International Symposium on Circuits and Systems (ISCAS), vol.1, pp.215–218, 1995.
[4] D.J. H. Huang, A.B. Kahng, and C.W.A. Tsao, "On the bounded-skew routing tree problem," Proc. 32nd Design Automation Conference (DAC), pp.508–513, 1995.
[5] S. Ishijima, T. Utsumi, T. Oto, and A. Takahashi, "Semi-synchronous circuit design method by clock tree modification," Proc. Workshop on Synthesis and System Integration of Mixed Technologies (SASIMI), pp.382–386, 2001.
[6] K. Kurokawa, T. Yasui, M. Toyonaga, and A. Takahashi, "A practical clock tree synthesis for semi-synchronous circuits," IEICE Trans. Fundamentals, vol.E84-A, no.11, pp.2705–2713, Nov. 2001.
[7] M. Saitoh, M. Azuma, and A. Takahashi, "Clustering based fast clock scheduling for light clock-tree," Proc. Design Automation and Test in Europe Conference and Exhibition (DATE), pp.240–244, 2001.
[8] A. Takahashi and Y. Kajitani, "Performance and reliability driven clock scheduling of sequential logic circuits," Proc. Asia and South Pacific Design Automation Conference (ASP-DAC), pp.37–42, 1997.
[9] M. Toyonaga, K. Kurokawa, T. Yasui, and A. Takahashi, "A practical clock tree synthesis for semi-synchronous circuits," Proc. ACM International Symposium on Physical Design (ISPD), pp.159–164, 2000.
[10] J.G. Xi and W.W.M. Dai, "Jitter-tolerant clock routing in two-phase synchronous systems," Proc. International Conference on Computer-Aided-Design (ICCAD), pp.316–320, 1996.
[11] J.G. Xi and W.W.M. Dai, "Useful-skew clock routing with gate sizing for low power design," Proc. 33rd Design Automation Conference (DAC), pp.383–388, 1996.
[12] T. Yoda and A. Takahashi, "Clock schedule design for minimum realization cost," IEICE Trans. Fundamentals, vol.E83-A, no.12, pp.2552–2557, Dec. 2000.

Seiichiro Ishijima received his B.E. degree in electrical and electronic engineering and M.E. degree in integrated systems from Tokyo Institute of Technology, Tokyo, Japan, in 2000 and 2002, respectively. He is currently working at NEC Electron Devices. He has been interested in the designing of the electronic circuit and the programming for embedded microprocessors. His recent interest is the designing of mixed signal VLSI circuits.



Tetsuaki Utsumi received his B.E. degree in electrical and electronic engineering and M.E. degree in integrated systems from Tokyo Institute of Technology, Tokyo, Japan, in 2000 and 2002, respectively. He joined Toshiba Corporation in 2002. He is currently with the System LSI Design Division, Toshiba Corporation Semiconductor Company. He has been interested in VLSI layout design.



Tomohiro Oto received his B.E. and M.E. degrees in electrical and electronic engineering from the Tokyo Institute of Technology, Tokyo, Japan, in 1999 and 2001, respectively. He currently works at Sony Computer Entertainment. His research interests are in microprocessor and graphics processor architecture.

Atsushi Takahashi received his B.E., M.E., and D.E. degrees in electrical and electronic engineering from Tokyo Institute of Technology, Tokyo, Japan, in 1989, 1991, and 1996, respectively. He had been with the Tokyo Institute of Technology as a research associate from 1991 to 1997 and has been an associate professor since 1997. He is currently with Department of Communications and Integrated Systems, Graduate School of Science and Engineering. His research interests are in VLSI layout design and combinational algorithms. He is a member of the Information Processing Society of Japan and IEEE.