PAPER Special Section on VLSI Design and CAD Algorithms

A Practical Clock Tree Synthesis for Semi-Synchronous Circuits

Keiichi KUROKAWA[†a)], **Takuya YASUI**[†], *Nonmembers*, **Masahiko TOYONAGA**[†], *and* **Atsushi TAKAHASHI**[††], *Regular Members*

SUMMARY In this paper, we propose a new clock tree synthesis method for semi-synchronous circuits. A clock tree obtained by the proposed method is a multi-level multi-way clock tree such that a clock-input timing of each register is a multiple of a predefined unit delay and the wire length from a clock buffer to an element driven by it is bounded. The clock trees are constructed for several practical circuits. The size of constructed clock tree is comparable to a zero skew clock tree. In order to assure the practical quality of the clock trees, they are examined under the five delay conditions, which cover various environmental and manufacturing conditions. As a result, they are proved stable under each condition and improve the clock speed up to 17.3% against the zero skew clock trees.

key words: *semi-synchronous circuit, clock scheduling, environmental and manufacturing conditions, zero skew clock tree, various timing clock tree*

1. Introduction

The semiconductor manufacturing process technology has improved the scale, performance and power consumption of LSI circuits. To design a large-scale circuit simply, most of designers adopt a synchronous circuit strategy of separating a circuit design into a clock part and a logic part. They assume a simultaneously distributed clock over the chip in the logic design. In DSM (deep sub-micron) process such as 0.25-micrometer rule, a large part of the signal propagation delay is caused by the wire, which makes a clock design difficult to satisfy the assumption of the strategy. To overcome this difficulty, a new design methodology is required.

A semi-synchronous circuit design that enables the different clock-input timings for registers would be one of solutions.

Fishburn [1] gave the necessary and sufficient conditions for a synchronous circuit to work correctly with a clock period in terms of the maximum and minimum signal path delays between registers and their clock-input timings. As a result, he showed a new approach to improve the circuit by carefully tuning clock-input timings.

Takahashi et al. [2], [3] interpreted the condition by using a constraint graph, and introduced a fast algorithm that determines clock-input timings for all registers. Albrecht et al. [4] proposed independently a different algorithm for a latch-based circuit. Neves [5], Kourtev [15] and Friedman had adopt it to a robust circuit design under the process variation. Liu et al. [6] recently proposed a combined method of the retiming technique and the clock-input timing technique. They showed methods for semi-synchronous circuits, however they did not specify the realization of layout design.

It is essential to control the clock-input timings physically in the semi-synchronous framework as well as the conventional framework.

In a usual type of clock circuit called zero-skew clock tree (ZSCT) as shown in Fig. 1(a), the clock skew would be relatively small since the numbers of buffers from the clock source to registers are the same, and the routing delays are almost same [7]–[9]. On the other hand, in a clock tree as shown in Fig. 1(b) for the semi-synchronous framework, called a various timing clock tree (VTCT), the delays from the clock source to registers differ.

There are two difficulties to realize VTCT. The first is the size of clock tree that tends to be larger than the ZSCT. The second is the stability of the clock-input timings under the physical conditions such as environmental and manufacturing conditions. Huang et al. [10] and, Xi and Dai [11] provided a smart approach that is obtained by modifying a ZSCT. Inoue et al. [12] showed that a smaller clock tree is obtained by setting

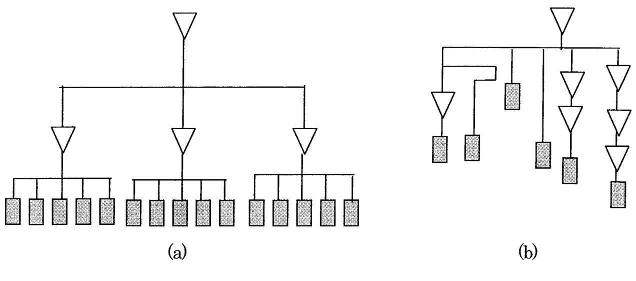


Fig. 1 Zero-skew clock tree and semi-synchronous clock tree. (a) Zero-skew clock tree and (b) Semi-synchronous clock tree.

Manuscript received February 27, 2001.
Manuscript revised May 31, 2001.

[†]The authors are with Semiconductor Company, Matsushita Electric Industrial Co., Ltd., Nagaokakyo-shi, 617-8520 Japan.

[††]The author is with the Department of Communications and Integrated Systems, Tokyo Institute of Technology, Tokyo, 152-8552 Japan.

a) E-mail: kurokawa@mrg.csdd.mei.co.jp

the clock-input timings between the neighboring registers as small as possible. Chen et al. [13] introduced an associated skew problem in which registers of the same clock-input timings are clustered. They evaluated approaches that provide smaller VTCT, though they did not examine the actual circuits.

In this paper, we propose a new clock tree synthesis method. In the clock tree obtained by the proposed method, the ratio of routing delay is limited in order to restrain the variation of the routing delay.

The synthesis procedure, first, determines the clock-input timing of each register by using the method of simulated-annealing. The clock-input timing of each register is set to a multiple of a predefined unit delay which is roughly equal to the delay caused by a clock buffer with typical load capacitance. Next, the proposed method determines the topology of the clock tree and inserts the clock buffers into the clock tree. The registers with the same clock-input timing are partitioned into clusters. The registers in each cluster are driven by a clock buffer. The wire length from a parent clock buffer to its child register is bounded in order to restrain the routing delay. The clock-input timing of a parent clock buffer is set to that of the registers in the cluster minus the unit delay. The clock buffers are regarded as a register in the next clustering. The unit delay from a parent to children is achieved by controlling the gate delay of the parent clock buffer. To control the gate delay, the procedure selects an appropriate topology, parent clock buffer location, and branching points of the interconnection for each cluster. Finally, the detailed routing of the clock tree is done by a vender tool.

The clock trees are constructed for several practical circuits. The size of each clock tree is comparable to a zero skew clock tree. In order to assure the practical quality, they are examined under the five delay conditions, which cover various environmental and manufacturing conditions. As a result, they are proved stable under each condition and our clock tree improves the clock speed up to 17.3% against the zero skew clock trees.

2. Preliminaries

Figure 2 illustrates a part of a synchronous circuit. Let R_i and R_j be the registers to which a clock with period T is inputted. Let S_i and S_j be the clock-input timings for R_i and R_j, respectively. Let CL be the combinatorial circuit between R_i and R_j with the signal propagation path $P_{\max_{ij}}$ of the maximum signal path delay $W_{\max_{ij}}$ and $P_{\min_{ij}}$ of the minimum signal path delay $W_{\min_{ij}}$.

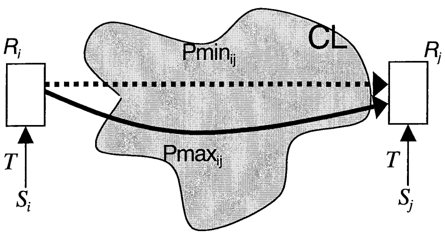


Fig. 2 A part of a synchronous circuit.

A synchronous circuit works with period T if and only if the following two inequalities are satisfied for every register pair with signal propagation paths as in [1].

Setup constraint (no zero clocking constraint)

$$S_i - S_j \leq T - W_{\max_{ij}} \tag{1}$$

Hold constraint (no double clocking constraint)

$$S_j - S_i \leq W_{\min_{ij}} \tag{2}$$

In case of design based on the conventional zero-skew clock tree, since S_i and S_j are assumed to be equal, the inequalities can be rewritten as follows.

$$S_i - S_j = 0$$

$$0 \leq W_{\min_{ij}} \leq W_{\max_{ij}} \leq T \tag{3}$$

The clock period T is bounded by $W_{\max_{ij}}$. While, if S_i and S_j are not necessarily to be equal and clock-input timings satisfying

$$S_i - S_j + W_{\max_{ij}} \leq T \leq \max(W_{\max_{ij}}). \tag{4}$$

are selected then the circuit works with clock period T less than the maximum signal path delay of the circuit.

We call the procedure of finding appropriate clock-input timings as *clock schedule optimization*.

3. The Verification under the Environmental and Manufacturing Variations

3.1 The Delay Model

The delay model used in constructing a clock tree is as follows. The signal propagation delay d_{path} from a gate to other gates consists of the gate delay d_{gate} and the routing delay d_{wire}.

The gate delay d_{gate} is defined as

$$d_{gate} = D_b + R_g \times C_{total} \tag{5}$$

where D_b, R_g, and C_{total} are the intrinsic delay, the output resistance, and the load capacitance of gate, respectively. The load capacitance C_{total} is

$$C_{total} = C_{in} + C_w \times L \tag{6}$$

where C_{in}, C_w, and L are the sum of input capacitances of gates driven by the gate, the unit wire capacitance, and the total wire length of the connections, respectively. In DSM, the capacitance per unit wire length varies with every interconnection, however, it is estimated by a statistical analysis. That is because a shorter wire would not be important to the signal propagation delay and a longer wire would be well estimated statistically.

The routing delay d_{wire} is defined as

$$d_{wire} = R_w \times L \times C_{total} \tag{7}$$

where R_w is the unit wire resistance. The delay model of d_{wire} is relatively rough for DSM era. Although more accurate formulation is possible, it is enough for us in constructing a clock tree since the wire resistance is controlled small in the clock tree.

The circuit with clock tree constructed is verified by using major vendor's timing analysis tools in which more accurate delay model is adopted.

3.2 The Environmental and Manufacturing Variations

To assure the quality of the design, the functionality of the LSI under the environmental and manufacturing variations should be verified. In the process narrower than 0.5-micrometer, the routing delay of a long interconnection can be comparable to the gate delay. Accordingly, the variation of routing delay as well as that of gate delay becomes large. Moreover, the direction of the variation of these delays might not be same.

The gate and wire are developed independently in manufacturing. Although the changes of most parameters such as temperature, voltage, widths of wire, and etc. influence both gate delay and routing delay, the amount of variation of gate delay caused by the change of a parameter is different from that of variation of routing delay caused by it. For example, in a certain process, the amount of variation of routing delay caused by the change of temperature is larger than that of gate delay, while the routing delay is almost invariant of the voltage. Therefore, the variation of gate delay and routing delay should be considered separately in the following.

The typical gate delay value is defined by Eq. (5). The smallest and largest gate delay are defined from typical gate delay value by multiplying coefficients that take the variations of processes, voltages, and temperatures into account. The typical, smallest, and largest of routing delay are defined similarly. Then there are nine combinations of gate and routing delays. However, it is sufficient to examine only five conditions that cover the nine conditions. We denote these five conditions as shown in Fig. 3 as ***BB***, ***TT***, ***WW***, ***BW***, and ***WB*** where

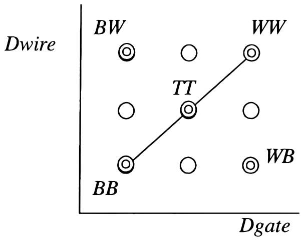


Fig. 3 Environmental and manufacturing five conditions for DSM.

$$D_{path}(BB) = D_{gate}(Best) + D_{wire}(Best),$$

$$D_{path}(TT) = D_{gate}(Type) + D_{wire}(Type),$$

$$D_{path}(WW) = D_{gate}(Worst) + D_{wire}(Worst),$$

$$D_{path}(BW) = D_{gate}(Best) + D_{wire}(Worst),$$

$$D_{path}(WB) = D_{gate}(Worst) + D_{wire}(Best). \tag{8}$$

where ***Type***, ***Best*** and ***Worst*** denote typical, smallest, and largest, respectively.

Let us consider a part of a synchronous circuit shown in Fig. 2. Let S_i be the typical clock delay from clock source to register R_i, and r_i be the percentage of the routing delay in S_i. Let A be the coefficient for maximum gate delay and routing delay and a be the coefficient for minimum gate delay and routing delay. Then the clock delays to R_i under ***TT***, ***BB***, ***WW***, ***BW***, and ***WB*** are

$$S_i, aS_i, AS_i, aS_i + (A-a)r_iS_i, \text{ and } AS_i - (A-a)r_iS_i,$$

respectively. Similarly, clock delays to R_j under five conditions are defined. Then the differences of clock-input timings between R_i and R_j under five conditions are

$$S_i - S_j,$$

$$a(S_i - S_j),$$

$$A(S_i - S_j),$$

$$a(S_i - S_j) + (A-a)(r_iS_i - r_jS_j), \text{ and}$$

$$A(S_i - S_j) - (A-a)(r_iS_i - r_jS_j),$$

respectively.

The deviation from the designed difference of clock-input timings as well as the variations of signal path delays may cause setup and hold violations. Even if S_i and S_j are designed to be equal, the difference of clock-input timings exists under ***BW*** and ***WB*** conditions if r_i is not equal to r_j. The differences in conditions ***BB*** and ***WW*** are larger than the zero skew clock tree, since the clock-input timings differ in general in VTCT. Thus, the circuits need to be verified in five conditions.

Although the main concern in this paper is the improvement of the reliability of VTCT with respect to the global variations over the chip, the local variations can be taken into account. For example, if the range of gate delay between $d_{gate_{\min}}$ and $d_{gate_{\max}}$ due to the local variation is inevitable, the $d_{gate_{\min}}$ or $d_{gate_{\max}}$ is used instead of d_{gate} in the delay calculation and the expressions of setup and hold constraints. However the local variations are smaller than global variations in general. For example, [18] reported that the local temperature variation is less than 10% at 100 C operation environment. In such cases, the delay variation due to the local variations is only 5% of total delay. Thus the local variations are neglected in expressions and experiments in this paper for simplicity.

4. A New Clock Tree

In conventional design, designers usually consider only one condition, usually the worst one, since it is difficult to consider five conditions simultaneously. The timing violations under other four conditions will be absorbed by some appropriate margin. This design strategy is premising when the zero skew tree is used since the clock skew is not varied drastically under five conditions. However, in semi-synchronous framework, the clock skew is large in general. If a large margin is required to absorb timing violations, then the performance of a circuit is degraded. We should generate a VTCT with lower clock skew to validate the design strategy that considers the worst condition with a smaller margin.

We propose a new type of clock tree as shown in Fig. 4 in order to adopt the conventional design strategy that considers only one condition with a smaller margin in the semi-synchronous circuit design.

The basic idea is to limit the routing delay in a clock tree. Of course, there exists a clock skew caused by the variations of gate delay. However, from Eq. (8), severe conditions such as $\boldsymbol{BW}$ and $\boldsymbol{WB}$ are nearly equal to conditions $\boldsymbol{BB}$ and $\boldsymbol{WW}$. As a result, the clock skew in our clock tree is relatively small, and that enables us to adopt the conventional design strategy in semi-synchronous circuit design. In the circuit with the proposed clock tree, the timing violations are seldom occurred under another condition. However, if the timing violations are occurred under another condition, combinational logics of the circuit are locally modified to satisfy the timing constraints under all the conditions.

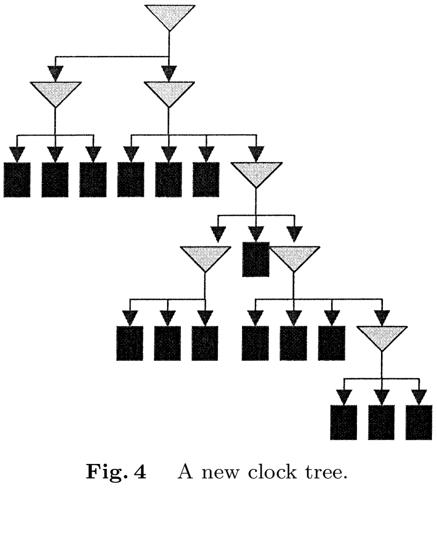


Fig. 4 A new clock tree.

The proposed clock tree is a multi-level multi-way clock tree. The root and internal nodes of the tree correspond to the drive buffers, and the leaves correspond to the registers. A drive buffer drives other drive buffers and/or registers. The clock-input timing of each node represents the clock delay from the clock source to the node minus the target clock delay.

4.1 The Discrete Clock Schedule Optimization

The method of simulated annealing [14] is used to obtain a discrete clock scheduling solution. A multiple of the unit delay value U_t is assigned to each register R_i as a clock-input timing S_i by simulated annealing. U_t is previously defined such that U_t is roughly equal to the delay caused by a buffer with typical load capacitance.

Let V and E be the set of registers and be the set of register pairs with signal propagation path, respectively.

In simulated annealing, the clock schedule is evaluated by the following cost function to obtain a clock schedule with shorter clock period that consists of discrete clock timings satisfying Eqs. (1) and (2).

$$\begin{aligned} COST &= A\times \max_{\{i,j\}\in E}(S_i - S_j + W_{\max_{ij}} - T) \\ &\quad + B\times \sum_{\{i,j\}\in E} \left\{ \max\left(0, S_i - S_j + W_{\max_{ij}} - T\right) \right\} \\ &\quad + C\times \sum_{\{i,j\}\in E} \left\{ \max\left(0, S_j - S_i - W_{\min_{ij}}\right) \right\} \\ &\quad + D\times \sum_{j\in V} (S_j - S_o)^2 \qquad (9) \end{aligned}$$

where T denotes the minimum clock period derived by the conventional scheduling method [2] and $S_o = 0$ is the target clock-input timing of registers. The first term denotes the difference of the target clock period T and an achieved clock. The second and third terms are the sum of the setup violations and the sum of the hold violations, respectively. The fourth term objective is to obtain an akin tree to ZSCT that takes practical size. The parameters of A, B, C and D are constant values.

In the experiments, we define $U_t = 0.6$ [ns] by evaluating worst buffer delay, and limit the maximum interconnection length in a cluster such that the delay caused by the resistance of the interconnection is at most 5% of U_t. We set the target clock delay 2.4 [ns]. That is, the clock-input timing of a register is defined 0 [ns] when the clock delay to the register is 2.4 [ns]. The clock-input timings for registers are restricted from -1.2 [ns] to 1.8 [ns] by 0.6 [ns] step, which follow the range of clock-input timings in the continuous clock schedule obtained by [2] that achieves the minimum clock period.

4.2 The Discrete Clock Tree Synthesis

In the clock synthesis procedure, clock buffers are inserted into the clock tree. Each clock buffer drives registers and/or other clock buffers with the same clock-input timing. The clock-input timing of a clock buffer that drives registers/buffers with clock-input timing $i \times U_t$ is set to $(i-1) \times U_t$.

The delay from a parent clock buffer to the children (registers/buffers) should be U_t in the clock tree. The maximum wire length from a parent to a child is limited so that the routing delay of a wire is negligible. The delay U_t from a parent to children is achieved by controlling the gate delay (D_{gate}) of the parent buffer. Note that the gate delay of the parent buffer depends on the driving ability of the buffer, the sum of the input-pin capacitance of the children and the interconnection capacitance.

Let R_{s_i} be the set of registers to which $i \times U_t$ is assigned as the clock-input timing ($1 \leq i \leq m$) in the discrete clock schedule.

First, the clock synthesis procedure partitions R_{s_m} into clusters such that the registers in a cluster are driven by a buffer. Let B_{m-1} be the set of clock buffers that drive R_{s_m}. Then, the procedure repeats the following from $i = m-1$ to 1: partition $R_{s_i} \cup B_i$ into clusters such that the registers/buffers in a cluster is driven by a buffer, and define B_{i-1} as the set of clock buffers that drive $R_{s_i} \cup B_i$.

Let $G(C)$ be the center-of-gravity of the cluster C, and $MST(C)$ be the length of the Minimum Spanning Tree of cluster C. The distance $d(C_s, C_t)$ between two clusters C_s and C_t is defined as $\min\{d(v_s, v_t) | v_s \in C_s, v_t \in C_t\}$ where $d(v_s, v_t)$ is the Manhattan distance between v_s and v_t.

The detailed clustering procedure in each repetition is as follows.

1. Let C_j be the initial cluster that consists of $v_j \in (R_{s_i} \cup B_i)$.
2. Select a pair of clusters (C_s, C_t) such that $d(C_s, C_t)$ is minimum among unselected pairs. If there is no unselected pair or $d(C_s, C_t) > L_{\max}$, then finish the clustering procedure.
3. Let $EIL = MST(C_s \cup C_t \cup G(C_s \cup C_t))$ be the Estimated Interconnection Length of $(C_s \cup C_t)$. Estimate the gate delay (D_{gate}) of a buffer that drives $(C_s \cup C_t)$ by using EIL.
4. If the estimated gate delay is less than U_t, then merge cluster C_s and C_t. (add $(C_s \cup C_t)$ and delete C_s and C_t)
5. Return to 2.

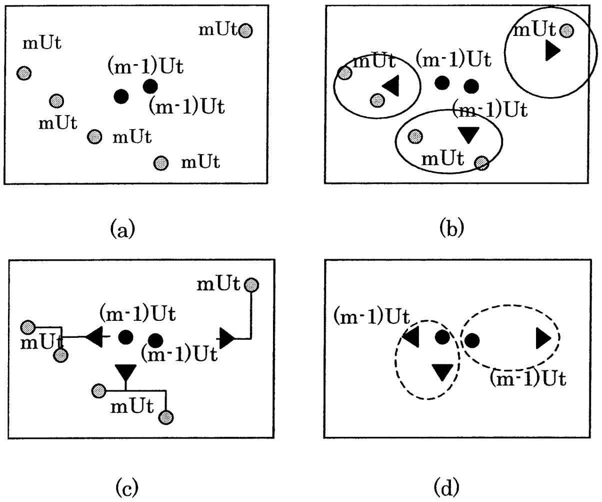


Fig. 5 Process to construct the clock tree.

In the estimation of the gate delay, we use the parameter in ***Worst*** condition.

In Fig. 5, a light circle denotes the register with maximum clock-input timing $m \times U_t$. The black circles and triangles denote parent nodes assigned $(m-1) \times U_t$. Figure 5(b) shows clusters of clock-input timing of $m \times U_t$. The clustering is repeated as shown in Fig. 5(c) and Fig. 5(d).

4.3 The Clock Tree Global Routing

The interconnection for a cluster is constructed by using cost-radius-balanced Steiner tree (CRBST) [17]. For a single source v_s and multiple sinks V, the CRBST constructs a Steiner tree that connects them. A Steiner tree constructed by the CRBST is varied by parameter c of the CRBST.

The CRBST starts with a single source and iteratively adds a sink with minimum cost to a partial Steiner tree S_t until every sink is contained in the Steiner tree. Roughly speaking, the cost of sink v is defined as

$$COST(v) = \min_{u \in S_t} \left\{ c \times \frac{d(v_s, v)}{Ra} \times d_T(v_s, u) + d(u, v) \right\} \quad (10)$$

where Ra is $\max_{u \in V}(d(v_s, u))$, and $d_T(v_s, u)$ is the wire length from v_s to u in S_t. Thus, the CRBST performs like Prim's minimum spanning tree algorithm when parameter c is nearly equal to "0" or a sink is located near the source. While the CRBST performs like Dijkstra's shortest path algorithm when parameter c is nearly equal to "1" and a sink is apart form the source.

Table 1 The experiment circuits.

Circuits	# of Cells	# of Regs.
C1	6536	1226
C2	21657	5363
C3	24646	7672

See [17] for detail.

A source, the buffer for a cluster, is located at the center-of-gravity of the cluster. Various Steiner trees for each cluster are constructed by changing parameter c of the CRBST, and select an appropriate one among them. That is, a Steiner tree such that the gate delay estimated by using the length of the Steiner tree is near U_t but does not exceed U_t and that the estimated routing delay is negligible is selected.

If it is impossible to select such a Steiner tree, buffer location would be moved or the cluster would be divided.

In the experiment, it was confirmed that the difference of gate delays in the clock tree with respect to pre/post layout parasitics is less than 3.3%.

5. Experiments

5.1 The Experimental Conditions

5.1.1 Overview of the Benchmark Circuits

Table 1 shows the overview of the benchmark circuits. These circuits are sub blocks of an image processing LSI that has been manufactured in our company by 0.25-micrometer process on 1.8 Volt. The clock speed specification is 12.3 nanosecond (ns).

5.1.2 The Parameters of Simulated-Annealing

The parameters of simulated-annealing are as follows.

Cooling parameter: 0.95
Starting temperature: 10.0

The end temperature is defined as the temperature when the cost improvement converges within +/−0.1%.

5.1.3 The Five Conditions

As a matter of fact, the five conditions are different in each silicon foundry. According to semiconductor data sheets published on the Web, $D_{gate}(Best)/D_{gate}(Type)$ is ranged from 0.5 to 0.7, $D_{gate}(Worst)/D_{gate}(Type)$ is ranged from 1.6 to 1.8 in 0.25 to 0.35 micrometer process [16]. According to those values, we select the environmental conditions as shown in Table 2 for our experiments. $D_{wire}(Worst)/D_{wire}(Type)$ is set smaller than $D_{gate}(Worst)/D_{gate}(Type)$ since D_{wire} is not influenced by voltage variations.

Table 2 Five conditions.

Condition	*BB*	*TT*	*WW*	*BW*	*WB*
Dgate/Dgate(Type)	0.5	1.0	2.0	0.5	2.0
Dwire/Dwire(Type)	0.5	1.0	1.5	1.5	0.5

Table 3 Comparison of clock period (unit: ns).

Circuits	Sync.	Prev. Semi-Sync	Our	Improved(%)	
				Semi	Our
C1	11.956	9.747	9.89	18.5	17.3
C2	11.911	11.66	11.67	2.1	1.2
C3	11.808	9.545	9.91	19.2	16.1

Table 4 Comparison of clock tree for the benchmarks.

Circuit	#of Buffers		Wire length(mm)	
	0-Skew	Ours	0-Skew	Ours
C1	46	47	45.47	46.92
C2	204	211	195.88	208.37
C3	280	281	260.70	263.11

5.2 The Experimental Result

5.2.1 The Benchmark System

Our clock synthesis generates the clock net lists, clock buffer placements and their routing topologies. They are linked to a major vendor's layout tool to complete designs. Standard Delay Format (SDF) files are prepared by the parasitic extraction and the static delay analysis using vendor tools. The verification diagrams are obtained by modifying SDF files according to the five conditions.

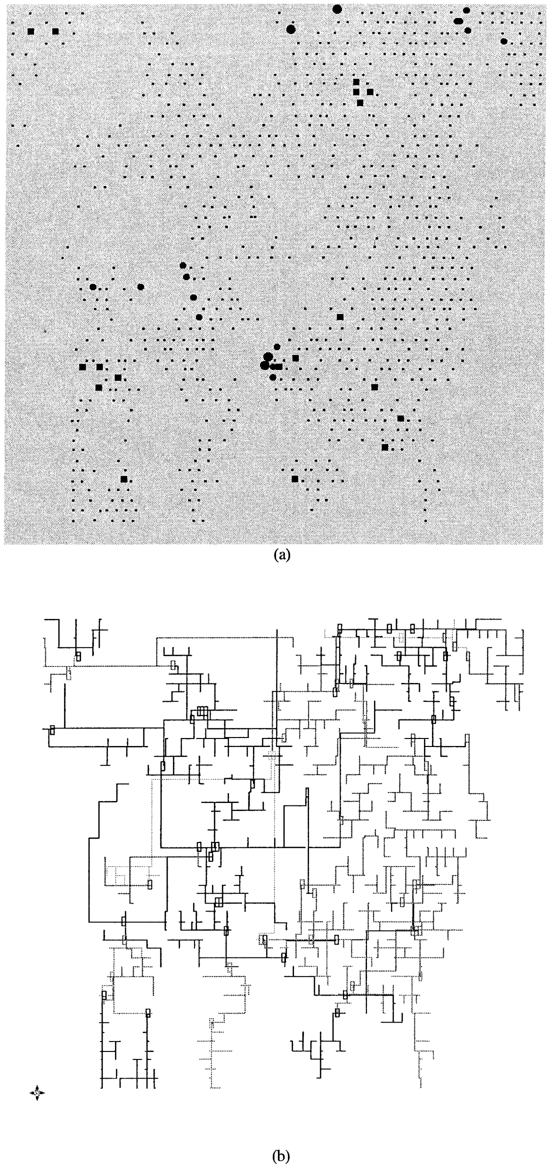
5.2.2 Comparisons

We compared our clock tree and the previous VTCT obtained by [2]. The discrete clock schedule is obtained by our algorithm in UA1 (200 MHz Work Station by the Sun-Microsystems Inc.). The CPU time was about 1.5 hours.

We worried that our discrete clock-input timing strategy is worse in clock speed than the continuous clock schedule since its solution space is limited. However, it achieved almost the same improvement level, i.e., the clock speed is improved 1.2 to 17.3% in our strategy while 2.1 to 19.2% in the continuous clock schedule (Table 3).

Note that the improvement of the circuit C2 is small since C2 contains a signal propagation path from the register to the same register (a loop) with large signal path delay which invalidates the semi-synchronous technique.

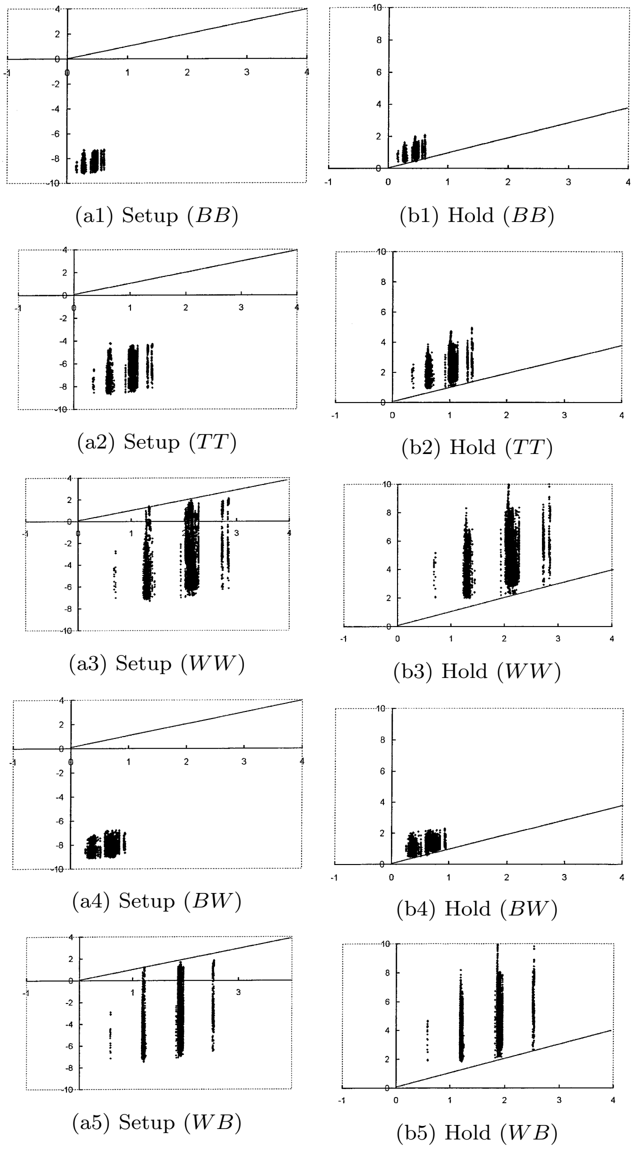
Table 4 shows the comparison of the number of



(a)

(b)

Fig. 6 Clock delay and interconnections of the clock tree. (a) Clock delay distributions and (b) Interconnections.



(a1) Setup (*BB*)

(b1) Hold (*BB*)

(a2) Setup (*TT*)

(b2) Hold (*TT*)

(a3) Setup (*WW*)

(b3) Hold (*WW*)

(a4) Setup (*BW*)

(b4) Hold (*BW*)

(a5) Setup (*WB*)

(b5) Hold (*WB*)

Fig. 7 Verification results over the five conditions.

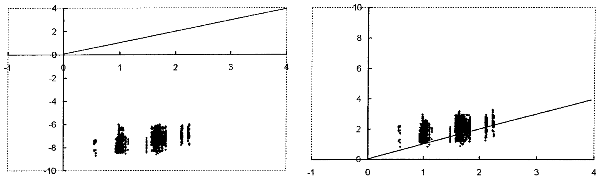
buffers and the wire length of our clock circuit and a zero skew clock circuit. The differences of both indices are less than 7%.

A scheduling result and a clock tree layout based on our procedure are shown in Figs. 6(a) and (b), respectively. The big black circle, the small black circle, the dot, and the black square are corresponding to registers with clock delay 1.2 ns, 1.8 ns, 2.4 ns, and 3.0 ns, respectively.

5.2.3 Verifications

The result of the setup and hold verification for circuit C1 under five conditions are shown in the diagrams in Fig. 7. In the diagram for setup verification, the horizontal and vertical axes are S_j and $S_i + W_{\max} - T$, respectively, and each dot in the diagram corresponds to the signal path in the circuit. If there is no setup violation in the circuit, each dot is located below the diagonal line in the diagram. Similarly, in the diagram for hold verification, the horizontal and vertical axes are S_j and $S_i + W_{\min}$, respectively, and each dot is located above the diagonal line if there is no hold violation. It is confirmed that the circuit C1 works correctly in every condition since the region of dots does not intersect the diagonal line in each diagram in Fig. 7.

To evaluate the validity of our clock tree structure, we generate a clock tree for the circuit C1 with the same clock schedule of the proposed clock tree, but without limitation of routing delay. The verification result in ***BW*** is shown in Fig. 8. Many hold violations generated by the variation of clock input timing are observed in the circuit. A larger routing delay in the clock tree



(a6) Setup (BW) (b6) Hold (BW)

Fig. 8 Verification results of the clock tree controlled by routing delay.

causes a larger variation of the routing delay, and a larger variation of the clock input timing.

6. Conclusion

We proposed a new clock tree with synthesis method that produces appropriate clock timings of registers with an LSI quality assurance of the five environmental and manufacturing conditions. By using a discrete clock-input timing strategy, the routing delay of a clock tree as well as the size of it can be small. This reduces the influence of the environmental and manufacturing variations on timing violations. The experimental results for practical three circuits showed that they satisfy the five conditions of quality assurance, while their performances were improved up to 17.3% compared to the zero-skew based circuits.

Our approach is independent of the process technology, and will be effective to improve the clock speed of the circuit in the DSM era. In fact, we have already confirmed that our approach is effective in 0.18-micrometer process.

As a future work, we are planning to develop an effective cost function to reduce the peak current for relaxing the EMI and IR-Drop problems in our framework.

Acknowledgment

We thank to Prof. Yoji Kajitani and Mr. Tomoyuki Yoda of Tokyo Institute of Technology for their useful suggestions, and Mr. Yoshifumi Okamoto and Mr. Shiro Tsuji of Matsushita Electric Industrial Co., Ltd. for their supports.

References

[1] J.P. Fishburn, "Clock skew optimization," IEEE Trans. Comput., vol.39, pp.945–951, 1990.

[2] A. Takahashi and Y. Kajitani, "Performance and reliability driven clock scheduling of sequential logic circuits," Proc. Asia and South Pacific Design Automation Conf., pp.37–42, 1997.

[3] A. Takahashi, K. Inoue, and Y. Kajitani, "Clock-tree routing realizing a clock-schedule for semi-synchronous circuits," Proc. ICCAD, pp.260–265, 1997.

[4] C. Albrecht, B. Korte, J. Schietke, and J. Vygen, "Cycle time and slack optimization for VLSI-chips," Proc. ICCAD, pp.232–238, 1999.

[5] J.L. Neves and E.G. Friedman, "Optimal clock skew scheduling Tolerant to Process Variations," Proc. 33rd Design Automation Conf., pp.623–628, 1996.

[6] X. Liu, M.C. Capaefthymiou, and E.G. Friedman, "Maximizing Performance by Retiming and Clock Skew Scheduling," Proc. 36th Design Automation Conf., pp.231–236, 1999.

[7] M.A.B. Jackson, A. Srinivasan, and E.S. Kuh, "Clock routing for high-performance ICs," Proc. 27th Design Automation Conf., pp.573–579, 1990.

[8] T.H. Chao, Y.C. Hsu, J.M. Ho, K.D. Boese, and A.B. Kahgn, "Zero skew clock routing with minimum wire length," IEEE Trans. Circuits & Syst., vol.39, no.11, pp.799–814, 1992.

[9] M. Edahiro, "A clustering-based optimization algorithm in zero-skew routings," Proc. 30th Design Automation Conf., pp.612–616, 1993.

[10] D.J.-H. Huang, A.B. Kahng, and C.-W.A. Tsao, "On the bounded-skew clock and steiner routing problems," Proc. 32nd Design Automation Conf., pp.508–513, 1995.

[11] J.G. Xi and W.M. Dai, "Useful-skew clock routing with gate sizing for low power design," Proc. 33rd Design Automation Conf., pp.383–388, 1996.

[12] K. Inoue, W. Takahashi, A. Takahashi, and Y. Kajitani, "Schedule-clock-tree routing for semi-synchronous circuits," IEICE Trans. Fundamentals, vol.E82-A, no.11, pp.2431–2439, Nov. 1999.

[13] Y. Chen, A.B. Kahng, G. Qu, and A. Zelikovsky, "The associative-skew clock routing problem," Proc. ICCAD, pp.168–172, 1999.

[14] S. Kirkpatrick, C.D. Gelatt, Jr., and M.P. Vecchi, "Optimization by simulated annealing," Science, 220, pp.671–680, 1983.

[15] I.S. Kourtev and E.G. Friedman, "Clock Skew Scheduling for Improved Reliability via Quadratic Programming," Proc. ICCAD, pp.239–243, 1999.

[16] http://www.bgs.nu/sdw/ "Semiconductor datasheets on the Web."

[17] H. Mitsubayashi, A. Takahashi, and Y. Kajitani, "Cost-radius Balanced Spanning/Steiner Tree," IEICE Trans. Fundamentals, vol.E80-A, no.4, pp.689–649, April 1997.

[18] C.H. Tsai and S.M. Kang, "Cell-level placement for improving substrate thermal distribution," IEEE Trans. Comput.-Aided Des., vol.19, pp. 253–266, Feb. 2000.

Keiichi Kurokawa received his B.E., M.E. degrees mechanical engineering from Ritsumeikan University, Kyoto, Japan, in 1990 and 1992, respectively. He joined Matsushita Electric Industrial Co., Ltd., Osaka, Japan, in 1992. His research interests include high-level and logic synthesis, physical design synthesis of VLSI circuit, and combinational optimization algorithms. He is a member of IEEE.

Takuya Yasui received his B.E., M.E. degrees information engineering from Hiroshima University, Hiroshima, Japan, in 1991 and 1993, respectively. He joined Matsushita Electric Industrial Co., Ltd., in 1993. His research interests include high-level synthesis, physical design synthesis of VLSI circuit for design automation and graph algorithm. He is a member of the Information Processing Society of Japan.

Masahiko Toyonaga received B.S. from Yamaguchi University, Japan, in 1979, M.S. from Kobe University, Japan, in 1981, and Ph.D. degree in electronics engineering from Osaka University, Japan, in 1995. He joined Matsushita Electric Corporation, Kyoto, Japan, in 1982. He joined Semiconductor Research Center of Matsushita Electric Industrial Co., Ltd., Osaka, Japan, in 1993. He is now a member of Technology Liaison Office in Semiconductor Company of Matsushita Electric Industrial Co., Ltd., Tokyo, Japan. His research interests include logic/high-level synthesis, physical design synthesis of VLSI circuit, and stochastic optimization methods for design automation. He is a member of IEEE, and IPSJ.

Atsushi Takahashi received his B.E., M.E., and D.E. degrees in electrical and electronic engineering from the 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.