Physical Design CAD in Deep Sub-micron Era

Takashi Mitsuhashi, Takahiro Aoki, Masami Murakata,
and
Kenji Yoshida
Semiconductor DA & Test Engineering Center, TOSHIBA Corporation
580-1, Horikawa-cho, Saiwai-ku, Kawasaki 210, Japan

Abstract

In this paper, we will investigate the impacts of miniaturization of device dimensions that causes a paradigm shift in LSI design methodology. Major design issues in deep sub-micron LSIs, namely, wire delay, circuit complexity and power consumption will be discussed based on scaling theory. To resolve these issues, a concept called Layout Driven Synthesis and Optimization is introduced. Based on this concept, EDA programs including circuit optimizer, clock tree synthesis, technology mappers and so on, have been developed. Timing optimization and power minimization methods using the concept will be discussed in detail. Evaluation results obtained by proposed approach show superior performance and dramatic reduction of design period, and indicate validity of layout driven synthesis and optimization concept.

1 Introduction

As described by Dennard[1, 2], by adoption of ideal scaling with scaling parameter κ, geometrical dimensions and all voltages are reduced by $1/\kappa$. Impurity density of substrate is increased by κ. As a result, device density increases by κ^2 and power-dissipation density remains constant, and gate delay is reduced by $1/\kappa$. Following this scaling scenario, we can improve gate delay, increase device density, keep power-dissipation constant forever. This was a golden prophecy that promised the prosperity of today's LSI industries.

The reality is different from what is expected by the theory. Increasing power-dissipation is one of the most serious problem. One of the reason why we cannot achieve the constant power-dissipation density is that we cannot reduce the supply voltages. If the supply voltage remains constants, the power-dissipation increases by κ^3. The other reasons for increasing power-dissipation are increasing circuit complexity and operating frequency.

Wire delay is another serious problem. By ideal scaling, gate delay decreases by $1/\kappa$, but in reality, wire delay dose not decrease in such manner. One reason is that, because of the 3-D effect, the wire capacitance does not decrease as expected. The other reason is that, different from local interconnection, wire delay for global interconnection increases with increasing chip size. In conventional EDA paradigm, it was assumed that gate delay is dominant and wire delay can be negligible in higher level of design stages. As collapse of this paradigm, a serious "the chicken or the egg" type problem occurs. We need accurate wire delay for precise logic synthesis and even for high level synthesis. However, we cannot get accurate wire delay without layout that require netlist generated by logic synthesis, as input.

This problem was recognized in early '90s in logic optimization field, and several research efforts have been devoted. K. J. Singh et al.[3] and Yoshikawa et al.[4] proposed delay reduction methods that utilized the mapped netlist and improve it by local remapping, but layout is not considered explicitly. Pedram and Bhat [5, 6] proposed a technology mapping method that consider the wire delay and congestion by simulated placement of the circuit. Their approach is a pioneering work, but correlation between real placement and simulated one is more serious than expected. Lin et. al [7] proposed a gate sizing method that minimizes area under constraints of timing and delay, but actual placement is not considered. Kannan[8], Aoki et al.[9], and Sato et al.[10] proposed a logic optimization method based on layout. They claim 10 to 30% delay reduction. In other fields of EDA, this new paradigm becomes clear. Examples will be clock tree synthesis, RTL floorplanning, and so on. The important thing is that, for design optimization , we cannot do any thing without information based on layout. In this paper, post-layout logic optimization for timing and low power will be described, as an example.

2 Scaling Theory and Design Issues

Performance of Synchronous circuits, that is a major methodology in VLSI design, is determined by path delay between storage elements and skew in the clock distribution system. Maximum operating frequency of the circuit is determined by the worst path delay, not by average path delay. Timing optimization problem is formulated as to minimize the maximum path delay in the LSI. On the other hand, power dissipation is determined by sum of power dissipation of each gate in the LSI. Analysis of the interconnection length distribution in a logic chip[2] shows that interconnections consist of two groups, one is local interconnect and the other is global interconnect. These two groups show different statistical behavior with the scaling of dimensions.

Local interconnection: Applying ideal scaling by factor $\kappa(\kappa > 1)$to wire in LSI chip, wire width, thickness of metal, insulator thickness and even wire length become $1/\kappa$. As a result, resistance and capacitance of wire become κ and $1/\kappa$, respectively. This means RC delay cannot be improved by ideal scaling. To reduce the delay other scaling methods are proposed[2]. One example is to reduce wire width, metal thickness, and insulator thickness by $1/\sqrt{\kappa}$. By this scaling method, we can achieve RC delay scaling by $1/\kappa$.

Global interconnection: Different from local interconnection, wire length increases with $\sqrt{\kappa_A}$, where κ_A is the scaling factor for chip area. Scaling factor for resistance and capacitance are $\kappa^2\sqrt{\kappa_A}$, $\sqrt{\kappa_A}$, respectively. RC delay is described by $\kappa^2\kappa_A$. This means a drastic increase of global interconnection delay with scaling. Although several efforts to reduce RC delay of global interconnection are reported, the delay still increase as shown in Figure 1. Management of global interconnection is one of the most important design issue in deep sub-micron technology.

Power dissipation is another design problem in deep submicron VLSIs. Kuroda, et al[11]. reported a statistical data that power dissipations of MPUs and DSPs presented at ISSCC increase 4 times every 3 years.

The power dissipation of CMOS LSI is dominated by charge and discharge of load capacitance, although there is short circuit current component. Macroscopic power dissipation of a whole CMOS LSI chip is described by,

$$P_C = N \cdot \bar{C} \cdot f \cdot V_{DD}^2, \tag{1}$$

where, N is the number of gates in the LSI, f is op-

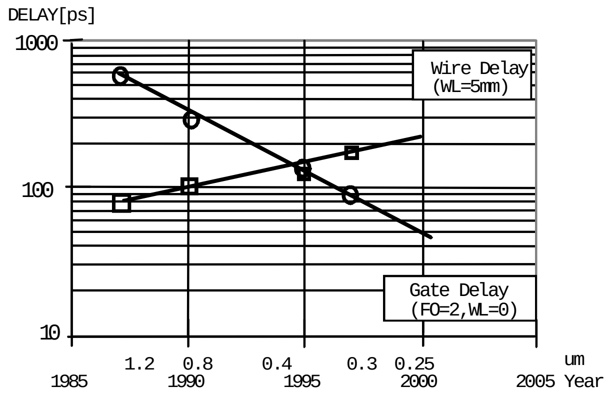

Figure 1: Wire Delay and Gate Delay

erating frequency, V_{DD}^2 is power supply voltage. $\bar{C}$ is average effective load capacitance, and described by,

$$\bar{C} = \frac{1}{N}\sum_{i=1}^{N} C_L^i \cdot p_i, \tag{2}$$

where C_L^i is load capacitance of gate i, p_i is switching probability of gate i in a clock cycle. Assuming changing rate per year of clock frequency α_f, scaling factor α_κ, power supply voltage α_{κ_V}, chip area κ_A, and power dissipation of current average LSIs P_0, power dissipation of average LSI after n year is described by,

$$P(n) = (\alpha_{\kappa_A}\alpha_\kappa\alpha_f\alpha_{\kappa_V}^2)^n P_0 \tag{3}$$

This equation(3) describes the trend of power dissipation, if trend parameters are given.

In this section, two critical issues in deep submicron LSI design have been discussed. One is wire delay, the other is power dissipation. In the following section, solutions based on layout driven synthesis and optimization will be described.

3 Layout Driven Optimization

In this section, a placement based net optimizer (PNO)[9, 12] is presented as an example of Layout Driven Optimization and Synthesis concept. PNO is a general purpose netlist optimizer based on layout information, that perform gate sizing, buffer insertion(Figure 2) and fanout decomposition(Figure 3). The program is applicable for timing optimization and minimization of power dissipation. The feature of the program is, that not only wire length for delay estimation but wire congestion and possibility of cell insertion on the target chip are considered.

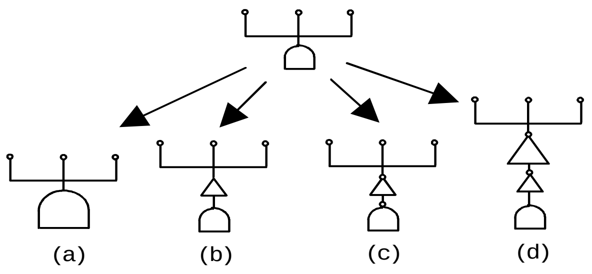

Figure 2: Repower:Gate Sizing and Buffer Insertion

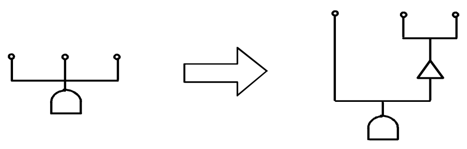

Figure 3: Fanout Decomposition and Buffer Insertion

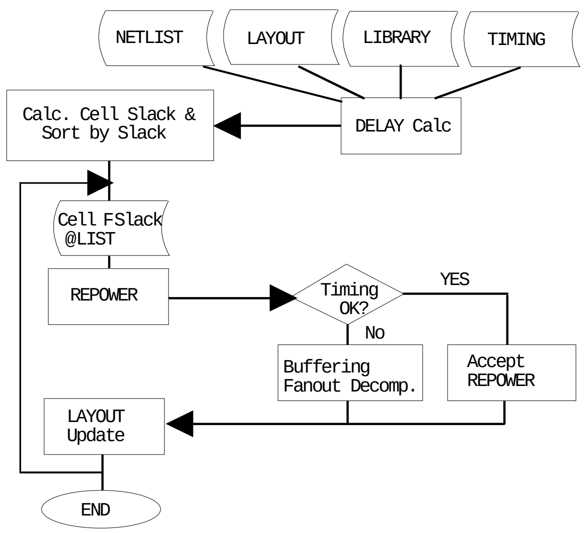

Figure 4: PNO: Circuit Optimization after Layout

3.1 Timing Optimization

Placement based net optimizer(PNO) is an effective logic optimization program based on accurate delay value extracted from layout. PNO searches for a feasible solution, or a solution that satisfies timing constraints, started from an arbitrary initial solution. Figure 4 depicts the system configuration of PNO. Delay related to each cell is calculated using input data, namely Netlist, Layout, Library, and Timing Spec. Slack of each cell is calculated, and cells are sorted by its slack value. For each cell that does not satisfy the timing constraints, following operation is performed: (1) reduce delay by gate sizing and buffer insertion that is described in Figure 2, if timing spec. is satisfied go to next cell. (2) if timing is not satisfied, try fanout decomposition and buffer insertion as depicted in Figure 3. New cells that are generated by buffer insertion and gate sizing are placed on the original layout given as input. In this layout updating process, impact on given layout is minimized to avoid bad influence on the chip performance.

Problems we must take care in fanout decomposition and buffer insertion, are as follows: (1) Some kinds of fanout decomposition increase wire congestion as shown in Figure 5. Decomposition program is required not to increase wire congestion. (2) Optimal placement of the inserted buffer is not guaranteed, because of less considerration of layout status of the chip. (3) Accurate delay estimation based on wiring resistance and capacitance is essential.

To cope with above mentioned problems, a network model that represent cell positon and wiring route as exact as possible is adopted. The network model consists of a set of vertices representing cells and Steiner points, and a set of edges representing possible wire routes. Steiner point is a vertex on a graph, that represents a possible branching point without cells. The network model is embedded in a plane that represents chip area. Each cell and Steiner point is mapped to a position where those geometrical objects are placed. Edges are also embedded in a plane where possible routes are available. Possible buffer insertion points where vacancy is available are also added to the model as edge vertex sets.

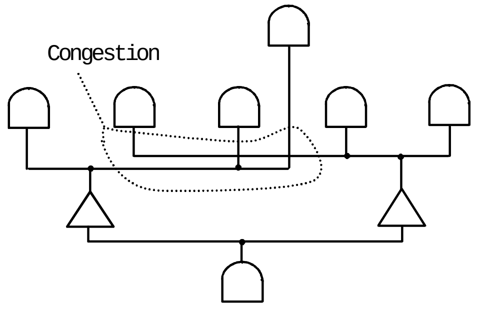

Figure 5: Wire Congestion by Fanout Decomposition

Figure 6 is an example of network model. In this graph, a set of vertices $V_C = \{V_0, V_1, V_2, V_3\}$ represents cells, $V_S = \{V_4, V_5, V_6\}$ represents Steiner points, and $V_B = \{g, g_1, g_2\}$ represents possible buffer insertion point, respectively. The edge set means possible wiring route. Direction is given to each edge, edges connected to signal source are directed from signal source to sink. Other edges are bi-directional.

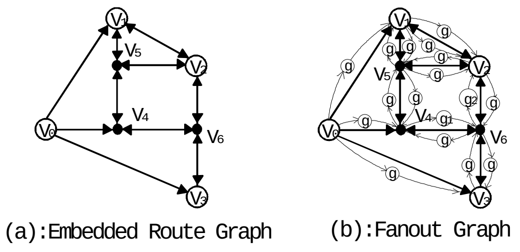

Figure 6: Network Model Generation

The network model is generated by two steps. In first step, route graph $G_R(V_R, E_R)$ is generated, where cell vertex set V_C represents cells given as the problem. XY-coordinate of $v_i \in V_C$ is the position of $Cell_i$. Steiner points V_S and edge is generated as follows: (1) Let signal source vertex be origin vertex. (2) Find the nearest vertex $v_i \in V_C$ in each quadrant from the origin vertex. (3) Generate edges that connect the origin and the nearest vertex in each quadrant. (4) Let the nearest vertex in each quadrant be next origin vertices. (5) Repeate (2) - (4), until all vertices are connected. (6) Make a bounding box for each generated edge. Newly generated vertices of the bounding box are treated as Steiner points and added to V_S. The new edges are added to E_R.

In the second step, fanout graph $G_F(V_F, E_F)$ is generated. Fanout graph is a model used for determining oiptimal tree structure and buffer insertion points. The procedure check each edge and corresponding region on a chip. If there is vacant space in the placement near the region, the program generate a buffer vertex and edges parallel to the edge under checking.

Delay calculation for this problem is interesting, the delay value is structure dependent and strongly influenced by buffer insertion. This nature of the delay calculation, we adopted a dynamic delay calculation method, that calculate the delay $d(v_0, v_t)$ from root v_0 to vertex v_t based on the value $d(v_0, v_{t-1})$,

$$\begin{aligned} d_a(v_0, v_t) &= d_a(v_0, v_{t-1}) + \beta R(v_{t-1}, v_t) C(v_{t-1}, v_t) \\ &\quad + \alpha R(v_{t-1}, v_t) C_i(v_t), \qquad (4) \\ d_a(;v_t) &= \alpha R(v_0) C_l(v_0), \qquad (5) \end{aligned}$$

where α and β are constants, and determined based on delay type to be calculated. $C_i(v_t)$ is load capacitance connected to terminal v_t, $C(v_{t-1}, v_t)$ and $R(v_{t-1}, v_t)$ are capacitance and resistance value for wire segment between v_{t-1} and v_t. These values can be calculated by estimated wire route or Manhattan wire length. $R(v_i)$ is the on-resistance at vertex v_i, $C_l(v_i)$ is total capacitance from vertex v_i to leaves.

If new vertex v_{t+1} is added to the tree at v_t, the delay from v_0 to v_{t+1} is represented as,

$$d_a(v_0, v_{t+1}) = d_a(v_0, v_t) + e(v_t, v_{t+1}), \qquad (6)$$

where,

$$\begin{aligned} & e(v_t, v_{t+1}) = C_\delta(\beta R(v_{t-1}, v_t) + \alpha R_m(v_t)) \\ & +\beta R(v_t, v_{t+1}) C(v_t, v_{t+1}) + \alpha R(v_t, v_{t+1}) C_i(v_{t+1}), \\ & R_m(v_t) = R(v_0) + ... + R(v_{t-1}) + R(v_t), \\ & C_\delta = C(v_t, v_{t+1}) + C_i(v_1). \end{aligned}$$

$e(v_t, v_{t+1})$ can be interpreted as the increase of delay by adding a new vertex v_{t+1} to the tree. When buffer is inserted, a special treatment is required for delay calculation. If there is a buffer in a path, the path delay is simple sum of delay from signal source to buffer input and buffer to net pin.

A^*-search algorithm[14] is used to find a optimal buffer insertion points and generate optimal fanout decomposition. The algorithm is efficient for finding optimal path between two vertices s and v. Usually, following objective function is used for A^*-search,

$$\hat{f}(v) = \hat{a}(v) + \hat{e}(v),$$

where $\hat{a}(v)$ gives estimated cost for optimal path between start vertex s and v, $\hat{e}(v)$ gives estimated cost for optimal path between v and terget vertex t. Compared this equation with equation(6), $d_a(v_0, v_t)$ corresponds $\hat{a}(v)$ and $e(v_t, v_{t+1})$ corresponds $\hat{e}(v)$.

Fanout decomposition and buffer insertion problem is a tree generation problem with timing constraints. The problem is stated as follows: Given a set of timing constraints from signal source to pins, find a optimal tree structure and buffer insertion points. The search is performed as follows: Let N_t be a set of subtree vertex, that is already determined and H be a heap that keeps current vertices for expanding the tree. Procedure is as follows: (1) Get vertex $u \in H$. (2) Evaluate cost for vertex v which is adjacent to u. (3) If v has not trace-back points, return v to H with evaluated value. (4) If v has trace-back points, and new cost is better, replace by new trace-back points and cost. (5) Repeat until the target vertex v^* is obtained from H.

3.2 Power minimization

Placement based net optimizer(PNO) is applicable to minimization of power dissipation. Convensionally power minimization procedure is considered as follows: (1) Find a circuit that satisfy the every timing constraints, (2) minimize the power dissipation under

@

Table 1: Comparison of two low power design method
@

	Without L.P.	Conven.	Proposed
$I_{DD}(mA)$	94.2	92.5	22.0
$Delay(nS)$	15.0	15.0	15.0

constraints of timing. We have proposed a new and effective procedure[12] for power minimization, that is described as follows: (1) change every cell to the smallest size without considering timing constraints, (2)then improve critical path by PNO. Comparison of two methods shows that the new procedure yields good results.

Major power dissipation of LSI consists of charge and discharge current of wiring capacitance I_w, cell input pin capacitance I_{pin}, parastic capacitance in cell I_{cell}, and short circuit current I_s. The method describrd here can reduce I_{pin}, I_{cell}, and I_s. I_{pin} and I_{cell} can be reduced by using smaller dimension cells. Short circuit current I_s depends on slope of input waveform and load capacitance of the gate. Excess I_s can be privented by setting limitation on load capacitance.

Above mentioned two methods are compared by bench mark test using an LSI circuit that consists of 32kcell and 34knet. The device technology is $0.5\mu m$. Results shown in table 1 indicates that proposed method gives good results and preferable. Without Layout Driven Optimization and Synthesis concept, It is difficult for us to perform such a drastic power dissipation by EDA programs.

4 Experimental Results

In this section, we will present some experimental results that validate the effectiveness of layout driven synthesis and optimization approach. A placement based net optimizer(PNO), that is an engine for timing optimization and power minimization, is implemented on SUN/Sparc 10. Bench mark results of both timing optimization and power minimization will be described, respectively.

Timing optimization - procedure for delay reduction is as follows: (1) placement by timing driven layout[13], (2) timing optimization by PNO. In step (2), critical paths are recognized and improved by PNO. As shown in Table 2[9], 8 circuits were selected for evaluation. Path delay before timing optimization

@

Table 2: Timing Improvement by PNO
@

CKT	Tech.	#	#	Delay(nSec)		Imprv.
@ @	μm	Net	Cell	Before	After	(%)
chip-1	0.5	60k	50k	23.00	18.98	17.5
				17.49	11.14	36.3
				15.74	11.14	29.1
				7.95	7.39	7.7
chip-2	0.8	23k	20k	50.09	41.99	16.2
				47.28	39.60	16.2
				44.61	35.75	19.9
				43.72	36.29	17.0
chip-3	0.8	54k	43k	52.44	44.77	14.8
				27.78	24.33	12.4
chip-4	0.8	23k	18k	42.87	39.88	7.0
				32.13	18.44	42.6
				22.53	16.52	26.7
				18.14	14.72	18.7
				17.81	12.99	27.1
				5.26	5.14	2.3
chip-5	0.8	14k	11k	10.88	10.81	0.6
				41.88	39.00	6.9
chip-6	0.5	12k	12k	14.87	13.68	8.1
chip-7	0.5	19k	17k	15.65	12.44	20.5
chip-8	0.5	30k	26k	13.73	11.54	15.9

is indicated in column "Before", delay after optimization is displayed under "After". 17% delay reduction in average is observed. CPU time required for PNO is evaluated for chip-6, -7, and -8, and reported 15.3min., 28.5min., and 17.5min., respectively.

Power minimization - procedure for power minimization is: (1) placement by timing driven layout, (2) change all cells into minimal dimension, and (3) improve path delays that violate timing constraints by PNO. As shown in Figure 7, two classes of sizing have been adopted. One is usual gate sizing, the other is F/F sizing. In usual gate sizing, flip-flops are not changed by optimizer. In F/F sizing, flip-flops with smaller size transistors associated with output buffer is replaced as a storage element. The size of output buffers are optimized by PNO. By virtue of small F/F, power dissipation by clock is reduced significantly. Flexible structure allows optimal selection of buffer size depending on the delay.

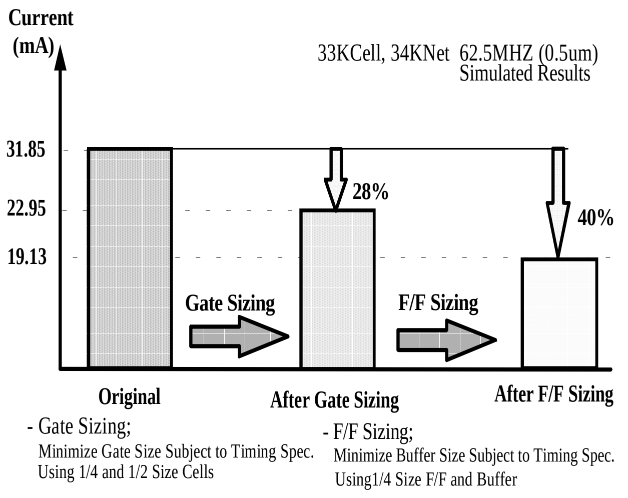

Figure 7: Power Reduction by Gate Sizing and F/F Sizing

5 Conclusions

A concept called layout driven synthesis and optimization that is indispensable for deep sub-micron LSI design is proposed. Power dissipation and wiring delay issues are discussed. These are predicted to be serious problems, if device feature size of LSI is reduced along with ideal scaling theory. We also pointed out that, if the wire delays of global interconnections becomes dominant, EDA paradigm will be required to change. High level or logic level designs based on conventional estimated delay becomes meaningless, and layout driven synthesis and optimization concept will be required. To validate this idea, we have made two experiments; timing optimization, and minimization of power dissipation. Placement based net optimizer was used as optimization engine in both cases. Results obtained by proposed approach show superior performance, and indicate validity of the layout driven synthesis and optimization concept.

References

[1] R.H. Dennard, F.H. Gaensslen, H.N. Yu, V.L. Rideout, E. Bassous, and A.R. LeBlanc, "Design of ion implanted MOSFET's with very small physical dimensions," IEEE J. of Solid-State Circuits, Vol.SC-9,pp.256-268, Oct.1974.

[2] H.B.Bakoglu, "Circuits, Interconnections, and Packaging for VLSI", Addison-Wesley,1990.

[3] K.J.Singh and A. Sangiovanni-Vincentelli, "A Heuristic Algorithm for the Fanout Problem", 27th DAC Proc., pp.357-360, 1990.

[4] K.Yoshikawa, H.Ichiryu, H.Tanishita, S.Suzuki, N.Nomizu, and A.Kondoh, "Timing Optimization on Mapped Circuits", 28th DAC Proc., pp.112-117, 1991.

[5] M.Pedram and N.Bhat, "Layout Driven Technology Mapping", 28th DAC Proc., pp.99-105, 1991.

[6] M.Pedram and N.Bhat, "Layout Driven Logic Restructuring/Decomposition", 1991 ICCAD Proc., pp.134-137, 1991.

[7] S.Lin, M.Marek-Sadoeska, E.S.Kuh, "Dealy and Area Optimization in Standard-Cell Design", 27th DAC Proc., pp.349-352, 1990.

[8] L.N.Kannan, P.R.Suaris, and H-G. Fang, "A Methodology and Algorithms for Post-Placement Delay Optimization", 31st DAC Proc., pp.327-332, 1994.

[9] T.Aoki, M.Murakata, T.Mitsuhashi and N.Goto, "Fanout-tree Restructuring Algorithm for Post-placement Timing Optimization", Proc. of the ASP-DAC '95, pp.417-422, 1995.

[10] K.Sato, M. Kawarabayashi, H. Emura, and N.Maeda, "Post-layout Optimization for Deep Submicron Design", Proc. of the 33rd DAC, pp.740-745, 1996.

[11] T.Kuroda and T.Sakurai, "A new guideline for low-power circuit design", A Technical White Paper on Low Power LSIs(in Japanese), Nikkei Business Publications, Inc.,Tokyo, 1994.

[12] T.Aoki, F.Minami, and M.Murakata, "A gate sizing and buffer insertion for power reduction", (in Japanese), Proc. of IEICE General Conference '96, SA-2, Sept., 1996.

[13] M.Murakata, M.Murofushi, M.Igarashi, T.Aoki, T.Ishioka, T.Mitsuhashi, and N.Goto, "Concurrent Logic and Layout Design System for High Performance LSIs", Proc. of 1995 CICC, pp.465-468, 1995.

[14] N.J.Nilsson, "Problem-Solving Methods in Artificial Intelligence", New York, McGraw-Hill, 1971.