A Global Router Optimizing Timing and Area for High-Speed Bipolar LSI's

Ikuo Harada† and Hitoshi Kitazawa‡

† NTT LSI Laboratories
3-1, Morinosato Wakamiya, Atsugi-shi
Kanagawa , 243-01, JAPAN

‡ NTT R&D Headquarters
1-1-7 Uchisaiwai-chou, Chiyoda-ku
Tokyo, 100, JAPAN

Abstract — A timing-driven global routing algorithm applicable to high-speed bipolar LSI's is proposed. Path-based timing constraints are directly modeled and routing paths are selected using novel heuristic criteria to minimize area as well as to satisfy the timing constraints by keeping track of the critical path delay and channel densities. Using bipolar-specific features, this router can be applied to Gbit/s LSI's. Experimental results shows that the average delay improvement was 17.6% and the algorithm is quite promising.

1 Introduction

The clock speeds of digital LSI's are rapidly increasing and some bipolar LSI's are reportedly able to process over-Gbit/s signals[1]. Therefore, the timing constraints for those LSI's are becoming much more crucial. Moreover, with the recent advances in VLSI fabrication technology, the interconnection delay is becoming a dominant factor. Therefore, timing-driven layout design has received increased attention in recent years. Much effort has been devoted to placement for minimizing critical delay[2, 3] and clock skew minimization[4, 5].

There has been only limited progress, however, in timing-driven routing[6, 7, 8]. Those routers use the single-net routing method under net-delay constraints, whereas conventional global routers are constructed to minimize routing area. J. Huang et al.[9] reported an area-minimizing method under net-delay constraint. However, the timing constraints are indeed given as the critical path constraints.

This paper proposes a heuristic method for minimizing the routing area and satisfying critical path constraints for standard cell LSI design. Novel heuristic criteria for the critical path delays and channel densities are used in the routing path selection mechanism based on the edge deletion routing scheme[10, 11].

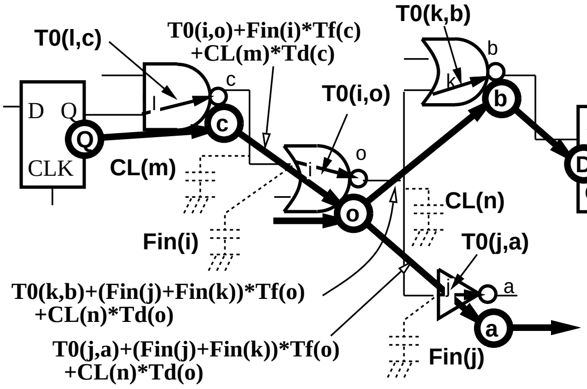

Figure 1: Delay model.

In bipolar LSI's that process over-Gbit/s signals, some special features are required such as differential drive net routings and multi-pitch width routings. These features were also integrated in the proposed algorithms.

2 Problem Formulation

2.1 Delay Model

In general, delays dependent on both capacitance and resistance in wire and cell terminals. However, in bipolar circuits, because their wires are made wider than those in CMOS circuits to reduce current density, the wire resistance is rather small. Thus, the authors adopted the capacitance delay model to simplify the problem. However, the extension to the RC delay model does not have any detrimental influence on the proposed algorithm.

Let $T_0(t_i, t_o)$ be the intrinsic delay of a cell and the fan-in factors $F_{in}(t)$ represent the capacitance at t. When a signal propagates from an input terminal t_i of a cell to fan-out terminals through an output terminal t_o, the delay time T_{pd} is

$$\begin{aligned} T_{pd} &= T_0(t_i, t_o) + (\sum_{t \in \mathcal{F}} F_{in}(t)) \cdot T_f(t_o) + \\ &\quad CL(n) \cdot T_d(t_o), \end{aligned} \qquad (1)$$

```
Algorithm Global_Router()
begin
    /* Feedthrough & ext. terminal assignment */
  xpin&feed_assign() ;
    /* $G_r(n) = (V_r, E_r)$ initialization */
  for all net $n$ do make_Gr($n$) ;
    /* $G_d(P)$ initialization */
  for all $P \in \mathcal{P}$ do make_Gd($P$) ;
  $N_b = \{e \in E_r | non\text{-}bridge\}$ ;
    /* Initial Routing */
  while $N_b \neq \emptyset$ do begin
    $e$ = select_edge($N_b$) ;
      /* $G_r(n)$, $N_b$, etc. modifications */
    delete_and_modify($e$) ;
  end ;
    /* Improvement loops by rerouting nets */
  recover_violate() ; /* Violation recover loop */
  improve_delay() ; /* Delay improvement loop */
  improve_area() ; /* Area improvement loop */
end
```

Figure 2: Outline of the global routing algorithm.

where $T_f(t_o)$ is the fan-in delay factor, $T_d(t_o)$ is the unit capacitance delay, $\mathcal{F}$ is a set of fan-out terminals from t_o, and $CL(n)$ is the wiring capacitance obtained from the wire length for net n.

Our delay model is illustrated in Fig. 1, where each value attached to each arrow in cells are the intrinsic delay. Because most cells have only one output terminal, the simplified graph, thick lines in Fig. 1, is adequate for analyzing critical paths. In this paper, this simplified graph is called the global delay graph G_D.

2.2 Critical Path Constraints

VLSI speed is limited by the largest delay path in a circuit, i.e., the "critical path." The critical path constraint P is thus defined as a trio (S_P, T_P, τ_P), where S_P and T_P are signal source and sink terminals, and τ_P is the delay limit. The delay constraint set $\mathcal{P}$ is the collection of constraints P required by the VLSI designer.

To estimate the critical path delay in global routing, the delay constraint graph $G_d(P)$ is defined for each constraint P as a subgraph of the global delay graph G_D. $G_d(p)$ includes vertices corresponding to S_P and T_P and all paths from the S_P vertices to the T_P vertices.

2.3 Global Routing Problem

When assuming bipolar standard cell VLSI design, global routing involves selecting cell terminals, assigning feedthrough positions, and determining terminals interconnected in each channel.

When given the cell placements, interconnection requirements, blockages on the routing layers, external terminal positions, the delay parameters for each cell type, and the critical path delay constraint set $\mathcal{P}$ are given, the global routing problem is defined to minimize the chip area, subject to the delay constraint set $\mathcal{P}$.

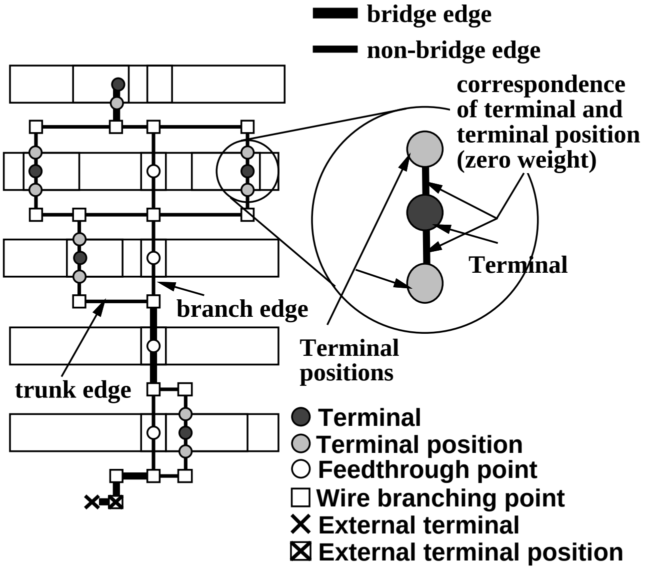

Figure 3: Definition of the routing graph $G_r(n)$.

3 Algorithm

3.1 Outline

The routing graph $G_r(n) = (V_r(n), E_r(n))$ is a weighted graph that serves as a candidate for the interconnection wiring of net n. Though a large-grid global routing graph was used in previous work[9], it is difficult to ensure the complete feedthrough assignment for multi-pitch wirings (see Section 4.2) by maintaining only the feedthrough capacity. Besides, there are not so many free feedthrough positions in bipolar LSI's.

Thus, our router assigns one feedthrough position for each net in a cell row in the first stage. It finds the feedthrough position by searching from the center of the x coordinates of the terminals. For nets that need to go through two or more cell rows, feedthrough positions are assigned in the same x coordinates if possible.

These assignments depend on the net ordering, and the order is defined according to a static delay analysis. By the forward and backward search of $G_d(P)$ with zero interconnection capacitance, slack values are obtained for each vertex of $G_d(P)$. Net ordering involves arranging the slack values in ascending order.

The interconnection wiring should be a tree in $G_r(n)$ and should include all terminal vertices. Vertices $V_r(n)$ in $G_r(n)$ have a one-to-one relation with circuit terminals or with physical points in the channel or feedthrough positions, as shown in Fig. 3. The edges represent the correspondence between the circuit terminal and its physical positions, or the interconnections between physical points. That is, they represent trunks or branch lines.

Edges whose deletion cause the routing graph to be disconnected are called *bridges*. Others are called *nonbridges*. Edges for deletion are selected only from *nonbridges*.

First, after routing graphs for all nets are made, one edge e is selected from all the edges of all $G_r(n)$ and then deleted. Next, criteria for heuristics are modified according to the deletion. The modified criteria are used in the next edge selection. Thus, the interconnection wiring of all nets is determined concurrently. The initial routing process is independent of net ordering.

3.2 Delay Estimation and Tentative Tree

To estimate the interconnection delay, the router estimates all wire lengths. The shortest paths from the driving terminal vertex to all other terminals are first obtained with Dijkstra's shortest-path algorithm. The union of all paths is the tentative tree.

When considering the RC delay model, the tentative tree generation could be modified, for example, by adopting the method of S. Prasitjutrakul[7]. Note that the routing flow and the heuristic criteria adopted in our algorithm are not influenced by this delay model change and/or tentative tree generation.

The interconnection delays are calculated according to the tentative tree The critical paths are, thus, obtained as the longest paths for each $G_d(P)$. For each constraint, a margin or slack $M(P)$ is obtained by subtracting the delay for the critical path from the specified time limit τ_P.

The deletion of e ($\in E_r(n)$) influences the $G_d(P)$ edges corresponding to n. We define a criterion: the local margin $LM(e,P)$ for $e \in E_r(n)$ and P ($\in P(e)$) to evaluate the influence, where $P(e)$ is a set of constraints whose $G_d(P)$ include edges corresponding to n.

By generating a tentative tree, assuming the deletion of e, a new delay value d' for edges (v,w) in $G_d(P)$ can be obtained. Let the original longest path delay to v be $lp(v)$ and that to w be $lp(w)$. The local margin $LM(e,P)$ is then defined as

$$\begin{aligned} &LM(e,P) = \\ &\qquad M(P) - \max_{(v,w)} \max(0, (lp(v) + d' - lp(w))). \quad (2) \end{aligned}$$

If w is on the original critical path, the $LM(e,P)$ is exactly the new $M(P)$ value after deleting e. Otherwise, it is a rather pessimistic estimation of the new $M(P)$ value.

According to the $LM(e,P)$, three heuristic parameters are defined for evaluating the disadvantages of deleting e. One is the critical count,

$$C_d(e) = |\{P | P \in P(e), LM(e,P) \leq 0\}|. \quad (3)$$

The second is the global delay $Gl(e)$, which is the difference between the sums of penalties:

$$\begin{aligned} Gl(e) &= \sum_{P \in P(e)} pen(LM(e,P),P) - \\ &\quad \sum_{P \in P(e)} pen(M(P),P), \quad (4) \end{aligned}$$

where the penalty function $pen(x,P)$ is

$$pen(x,P) = \begin{cases} 1 - x/\tau_P & \text{if } x \geq 0, \\ exp(-x/\tau_P) & \text{if } x < 0. \end{cases}$$

The third parameter is the local delay increase $LD(e)$, which is the sum of the delay increase in $G_d(P)$ edges resulting from the deletion of e. $LD(e)$ is a weak criterion predicting the future possibility of increasing critical path delay.

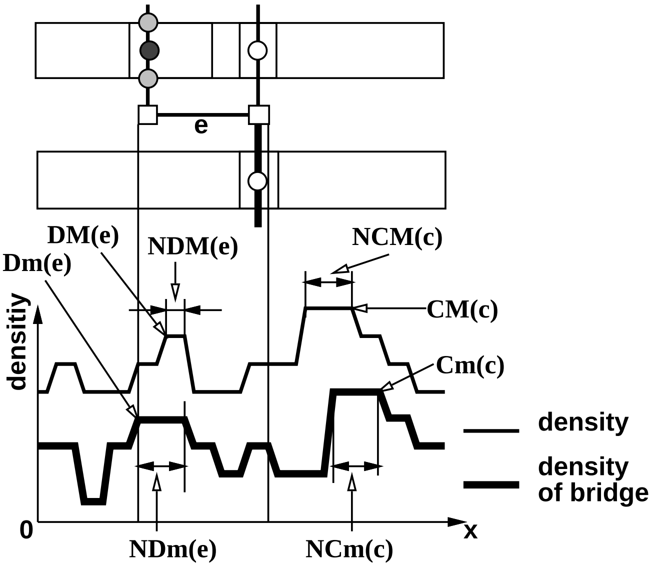

Figure 4: Density parameters.

3.3 Channel Density Estimation

The channel densities for each x coordinate on a wiring grid can be obtained by counting the number of $G_r(n)$ trunk edges. Let $d_M(c,x)$ be the number of edges running on x in channel c and let $d_m(c,x)$ be the number of bridge edges running in the same position. The maximum density in channel c and the length of the maximum points on $d_M(c,x)$ are defined as follows:

$$\begin{aligned} C_M(c) &= \max_x d_M(c,x), & (5) \\ NC_M(c) &= |\{x | d_M(c,x) = C_M(c)\}|. & (6) \end{aligned}$$

$C_m(c)$ and $NC_m(c)$ are similarly defined on $d_m(c,x)$. As seen from the definition, $C_m(c)$ and $C_M(c)$ are the lower and upper bounds of the density in channel c. To keep $C_m(c)$ low and to reduce $C_M(c)$ as fast as possible should be an effective strategy. Because the increase in $C_m(c)$ cannot be recovered, maintaining $C_m(c)$ is a stronger criterion. $NC_m(c)$ expresses how easily $C_m(c)$ increases. Thus, it is dangerous to delete edges in channels other than one with a large $NC_m(c)$. On the other hand, $NC_M(c)$ exhibits difficulty in reducing $C_M(c)$. Thus, deleting edges in a channel with a small $NC_M(c)$ is a greedy strategy.

The density parameters $D_M(e)$, $D_m(e)$, $ND_M(e)$, and $ND_m(e)$ for the edges e ($\in E_r$) are, using the interval of e, also similarly defined. Fig. 4 illustrates these parameters. The $d_M(c,x)$ and $d_m(c,x)$ chart example is shown and the density parameters are also illustrated.

3.4 Edge Selection Heuristics

The edge $e(\in N_b)$ is selected by comparing heuristics. To begin with, the delay criteria are compared. First $C_d(e)$'s, then $Gl(e)$'s, and next $LD(e)$'s are used. Because preceding criteria mean a more fatal disadvantage, the edge with a larger penalty is selected when ties are first broken.

The density conditions are examined when delay criteria are all the same. Assume that we are comparing e_1 and e_2, and let c_1 and c_2 be the channels where e_1 and e_2 exist. The followings are the conditions under which e_1 is selected.

The first condition prefers a trunk edge to a branch edge because the trunk edges directly reduces channel densities, whereas deleting the branch edge eliminate the possibility of reducing densities. The second condition concerns $C_m(c) - D_m(e)$ $(= F_m(c, e))$. If $F_m(c_1, e_1) < F_m(c_2, e_2)$, e_1 is selected so as not to increase $C_m(c_1)$.

The third condition concerns $NC_m(c) - ND_m(e)$ $(= N_m(c, e))$. When $F_m(c_1, e_1) = F_m(c_2, e_2) = 0$ in the second comparison, $N_m(c_1, e_1)$ shows the length of the most congested points uncovered by e_1. Otherwise, i.e. $F_m(c_1, e_1) = F_m(c_2, e_2) > 0$, $NC_m(c_1)$ shows how edges in c_1 are preferable and $ND_m(e_1)$ shows how e_1 is preferable. Thus, e_1 is selected when $N_m(c_1, e_1) < N_m(c_2, e_2)$ to maintain $C_m(c_1)$.

Similarly, $C_M(c_1) - D_M(e_1) < C_M(c_2) - D_M(e_2)$ is the fourth criterion and $NC_M(c_1) - ND_M(e_1) < NC_M(c_2) - ND_M(e_2)$ is the fifth criterion used to select edge e_1.

These conditions are examined in the order listed above. At each stage, symmetric conditions are examined for e_2. The comparisons go down until ties are broken. If they are still even after all comparisons, the longer edge is selected.

3.5 Improvements

The initial routing result is improved in three rerouting phases. The rip-up and rerouting processes are repeated one by one in each phase. The first is the constraint violation recovery phase: all nets on the critical paths that violate the constraints are improved. Then, in the delay improvement phase, all critical path nets under all constraints are considered. Nets in the critical paths of smaller $M(P)$ are rerouted first. The net rerouting order in the same critical path is arbitrary. The edge selection method is the same as that used in the initial routing.

The final phase is area improvement. In this phase, nets running in the most congested part are rerouted first. The edge selection heuristics are slightly modified to improve area: the density conditions are examined after $C_d(e)$ comparisons, and the $Gl(e)$ and $LD(e)$ are compared last.

4 Bipolar-specific Features

4.1 Differential drive

Differential drive is often used in ECL circuits to preserve noise margins especially for large fan-out nets. When two nets are specified as a differential drive pair, those nets must be routed physically parallel to each other.

The differential pairs assumed to be 2-pitch nets (described in Section 4.2) in the feedthrough assignment phase. Next, the one-to-one correspondence of edges in each routing graph is recognized by searching both graphs from driving terminal vertices. The correspondence is established if, and only if, routing graphs $G_r(n_1)$ and $G_r(n_2)$ are homogeneous and the relative positions of all adjacent vertices in $G_r(n_1)$ are the same as the corresponding ones in $G_r(n_2)$, where n_1 and n_2 make up the differential pair.

Table 1: Test bipolar circuits.

Data	Cir-	Place-	The number of		
Name	cuits	ments	cells	nets	consts.
C1P1	C1	P1	451	597	64
C1P2		P2			
C2P1	C2	P1	932	1176	68
C2P2		P2			
C3P1	C3	P1	936	1184	346

The edge deletion process is also modified so that when an edge included in the differential drive net is deleted, the corresponding edge in the paired one is also deleted.

4.2 Multi-pitch wires

Multi-pitch wires are required to reduce wire resistance and skews for very large fan-out nets like a clock. The adjacent feedthrough positions are occupied by the multi-pitch net. The delay and density criteria can be obtained using the wire widths. Note that if the net is specified to be interconnected in the w pitch width, the net is called a w-pitch net.

4.3 Feed-Cell Insertion

The standard cells of bipolar circuits normally have no space for feedthrough nets. Thus, the global router must utilize the space on the feed cell for the bipolar design. However, the global router often runs out of available feedthrough positions. By inserting feed cells, the global router makes additional space to ensure complete routing.

After the first assignment, the number of required feed cells $F(w, r)$ can be determined for the w-pitch nets in the rth cell row. $F(r) = \sum_w w \cdot F(w, r)$ is the required number for cell row r, and $F = \max_r F(r)$ is the number added for every row.

First, flags only for w-pitch nets are set on the feedthrough positions where w-pitch nets are assigned. Then, all previous assignments are canceled. $F(w, r)$ groups of w feed cells are then inserted into the cell row for each w $(w \neq 1)$. Each group is inserted almost evenly spaced between existing cells. Those groups are also flagged only for w-pitch nets. $F(1, r) + F - F(r)$ feed cells are then similarly inserted for single-pitch nets. Thus, the chip is widened by F pitches.

Finally, the feedthroughs are again assigned in the same way as described in Section 3.1, except that the width flags are taken into account. Therefore, the second assignment is always successful. These insertion and re-assignment processes assure the completeness of global routing.

5 Experimental Results

The proposed global router was implemented in C language on UNIX workstations. We tested three bipolar circuits. The first circuit, C1, was the regenerator-section overhead processing circuit for a 10-Gbit/s transmission system[1]. The others were also for the transmission system. Table 1 lists the circuit data.

Table 2: Experimental results.

Routing Results With Constraints				
Data Name	Delay (ps)	Area (mm^2)	Length (mm)	CPU (sec)
C1P1	375.6	10.85	618.4	33.0
C1P2	379.0	11.56	678.1	28.0
C2P1	527.5	14.77	1279.8	87.4
C2P2	615.4	16.20	1596.0	77.3
C3P1	324.0	19.20	1047.5	209.6

Routing Results Without Constraints				
Data Name	Delay (ps)	Area (mm^2)	Length (mm)	CPU (sec)
C1P1	377.7	10.84	624.0	17.5
C1P2	425.5	11.60	685.9	16.9
C2P1	689.5	14.74	1289.8	46.3
C2P2	795.3	16.18	1739.5	51.8
C3P1	327.9	19.31	1074.4	44.9

Table 3: Difference from the lower bound.

Data Name	lower bound (ps)	Difference (%)	
		Con-strained	Un-constrained
C1P1	356.2	5.44	6.03
C1P2	356.5	6.32	19.4
C2P1	461.5	14.3	49.4
C2P2	467.6	31.6	70.1
C3P1	294.7	9.97	11.3

The constraints for C1 and C2 were obtained through interviews with the logic designers. Constraints for C3 were improved according to the layout data analysis from the initial constraints provided from logic information. Realistic delay parameters were used for C1. Those for C2 and C3 were assumed to have similar average delays.

The placement P1s were given by designers by automatic feed-cell insertion. P2 placements were given by moving the feed cells aside in the cell rows in order to test the even spacing effect of feed-cell insertion.

The critical-path delays were obtained from routing lengths after channel routing in the same delay model. The resulting layouts with and without constraints were compared. Table 2 shows the results. CPU times were measured on SS2.

The improvement in constrained data varied from 0.56% to 23.5% in delays, whereas the area was almost unchanged.

Table 3 shows the differences in the critical-path-delay lower bounds. The lower bounds could be obtained by assuming the wire length for each net to be half the perimeter of the rectangle containing the net terminals.

The difference in the lower bound was largely improved such that it was less than half of the unconstrained results or less than 10%. The average reduction in the critical-path delay was 17.6% of the lower bound.

6 Conclusion

A global routing algorithm that minimizes both the critical path and interconnection area has been described. The global router was devised for standard cell LSI's, especially high-speed bipolar ones. During the routing process, the router keeps estimating wire lengths by generating tentative trees in the routing graph. Novel heuristic criteria evaluating critical-path delays and channel congestion were proposed to delete unsuitable paths. Differential-drive wiring, multi-pitch-width wires, and feed-cell insertion are introduced to preserve noise margins, minimize skews, and compensate for the lack of feedthrough positions in bipolar cells.

The experimental results show large critical-delay reductions in the resulting layouts of the constraint-assigned data. Otherwise, the timing issue would be serious. Other data have shown that the delays have been less than 10% above the lower bounds. The average delay reduction was 17.6%.

Acknowledgment

We thank Mr. Haruhiko Ichino, Mr Keiichi Koike and Mr. Kenji Kawai for valuable discussions about bipolar circuits. We also thank Mr. Yuuichro Takei for his assistance in the experiments.

References

[1] K. Koike, K. Kawai and H. Ichino: "A Design Methodology of Bipolar Standard Cell LSIs for Gbit/s Signal Processing", Proc. of the Biplolar/BiCMOS Circuits and Technology Meeting, pp. 236–239(1993).

[2] M. A. B. Jackson and E. S. Kuh: "Performance-Driven Placement of Cell Based IC's", Proc. of 26th IEEE DA Conference, pp. 370–375(1989).

[3] W. E. Donath, R. J. Norman, B. K. Agrawal, S. E. Bello, S. Y. Han, J. M. Kurtzberg, P. Lowy and R. I. McMillan: "Timing Driven Placement Using Complete Path Delays", Proc. of 27th IEEE DA Conference, pp. 84–89(1989).

[4] M. A. B. Jackson, A. Srinivasan and E. S. Kuh: "Clock Routing for High-Performance ICs", Proc. of 27th IEEE DA Conference, pp. 573–579(1990).

[5] R. S. Tsay: "Exact Zero Skew", Proc. IEEE ICCAD-91, pp. 336–339(1991).

[6] M. A. B. Jackson, E. S. Kuh and M. Marek-Sadowska: "Timing-Driven routing for Building Block Layout", Proc. ISCAS'87, pp. 518–519(1987).

[7] S. Prasitjutrakul and W. J. Kubitz: "A Timing-Driven Global Router for Custom Chip Design", Proc. IEEE ICCAD-90, pp. 48–51(1990).

[8] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh and C. K. Wong: "Provably Good Performance-Driven Global Routing", IEEE Trans. on CAD, **CAD-11**, 6, pp. 739–752(1992).

[9] J. Huang, X. L. Hong, C. K. Cheng and E. S. Kuh: "An Efficient Timing-Driven Global Routing Algorithm", Proc. of 30th IEEE DA Conference, pp. 596–600(1993).

[10] S. Tsukiyama, I. Harada, M. Fukui and I. Shirakawa: "A New Global Router for Gate Array LSI", IEEE Trans. on CAD, **CAD-2**, 4, pp. 313–321(1983).

[11] I. Harada, H. Kitazawa and T. Kaneko: "A Routing System for Mixed A/D Standard Cell LSI's", Proc. IEEE ICCAD-90, pp. 378–381(1990).