設計自動化 72－6
(1994. 10. 27)

タイミング制約を考慮したスタンダードセル概略配線手法

鈴木 武志　小出 哲士　若林 真一　吉田 典可

広島大学 工学部

〒724 広島県東広島市鏡山一丁目4番1号

あらまし 本稿では，タイミング制約を満たし，チャネル密度の最小化を目的とする概略配線手法を提案する．提案手法では，タイミング制約が与えられた2つのレジスタ間のすべてのパスに対してタイミング解析を行い，制約を満たし，かつチャネル密度を最小化する概略配線を求める．本手法は大きく分けて3つのフェーズからなる．まず，フェーズ1では初期概略配線経路を決定し，フィードスルーを挿入する．フェーズ2ではタイミング制約が与えられている2つのレジスタ間のパスに対してタイミング制約を満たすように概略配線経路を改良する．最後にフェーズ3ではタイミング制約を満たした上でチャネル密度が最小となるように概略配線経路を改良する．

和文キーワード　スタンダードセル，概略配線，タイミング制約，配線改良，スラックの分散

A Timing Driven Standard Cell Global Routing Method

Takeshi SUZUKI, Tetsushi KOIDE,
Shin'ichi WAKABAYASHI and Noriyoshi YOSHIDA

Faculty of Engineering, Hiroshima University

4-1, Kagamiyama 1 chome, Higashi-Hiroshima 724 JAPAN

Abstract In this paper, we propose a new global routing method, whose objective is to minimize the channel density under the given timing constraint. In the proposed method, timing analysis is done for all the paths between two registers. Moreover, the channel density is minimized, while keeping the given timing constraint. The proposed method is composed of three phases. In phase 1, initial global routes are determined and the feedthroughs are inserted. In phase 2, the global routes are improved to satisfy the timing constraint between two registers. Finally, in phase 3, the global routes are improved to minimize the channel density under the given timing constraint.

英文 key words standard cell, global routing, timing constraint, routing improvement, slack distribution

1 まえがき

近年の VLSI 設計技術の進歩により，チップのサイズはかなり小さくなってきており，単位長当たりの配線抵抗，及び配線容量による信号の伝搬遅延 (Interconnection Delay) のクロック周期に対する割合がかなり大きくなっている．このため回路の性能を向上させるには，タイミング制約を満たし，かつチップ面積を小さくするようなレイアウト設計手法 (Timing Driven Layout) が必要である．配置設計においては，非線形計画法を階層的に適用し，タイミング制約を満たすセルの配置を求める手法 [16]，非線形計画法を用いてタイミング制約を満たすように配置改良を行なう手法 [10] などさまざまな配置手法が提案されており，面積が小さいセルの配置を求めることができる．しかし，概略配線フェーズで実際の配線経路が決定されるため，配線フェーズでは配置フェーズよりもさらに正確にタイミング制約を考慮した手法により概略配線を行なう必要がある．

現在までに，スタンダードセルレイアウト設計における概略配線手法は数多く提案されている [3][5][12][17]．しかし，これらはすべてチップ面積，及び配線長の最小化のみを考慮しており，タイミング制約については考慮していない．文献 [8] において，セルベースのレイアウト設計における配線遅延を考慮した概略配線手法が提案されているが，2 つのレジスタ間のパスに対してタイミング解析を陽には行なっていないという問題点がある．また，ある 1 つのネットに対して，ソースから複数のロードまでの配線遅延の最小化を目的関数とするスタイナ木構成手法が数多く提案されている [2][4][7][11][13]．しかし，これらをスタンダードセルの概略配線フェーズで使用する場合，(1) 配線混雑度の考慮，(2) パスのタイミング制約の考慮，が必要となり，そのままでは使用できないという問題点がある．そこで本稿では，タイミング制約を満たし，チャネル密度の最小化を目的とする概略配線アルゴリズムを提案する．提案手法では，タイミング制約を受ける 2 つのレジスタ間のすべてのパスのタイミング解析を行ない，配線遅延の制約とチャネル密度の最小化の両方を考慮する．

以降では，2 節で提案手法で用いる配線遅延モデル，及びグラフモデルを定義し，概略配線問題の定式化を行なう．3 節では，提案アルゴリズムの詳細について説明する．最後に 4 節で提案アルゴリズムを計算機上で実現しシミュレーション実験を行なった結果について述べ，考察を行なう．

2 準備

本稿で取り扱うスタンダードセルモデルでは，セルはすべて同じ高さとし，各端子はセルの上下辺に等電位端子を持つものとする．提案アルゴリズムで用いるグリッドグラフ $G=(V,E)$ を図 1のように定義する．ここで，各節点 $v \in V$はチップ上の各チャネルにおける定数個の端子に対応する．枝集合 Eは，垂直方向枝 E_V と水平方向枝 E_Hに分けられ，枝 $(u,u') \in E_V$は，セルの両端の定数個の等電位端子対に対応し，枝 $(u,v) \in E_H$はチップ上の各チャネルの節点間に存在するものとする．

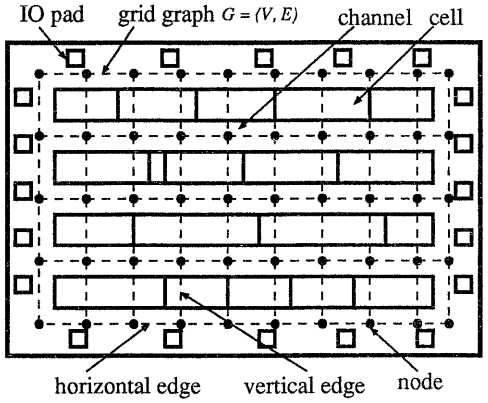

図 1 グリッドグラフ $G=(V,E)$

配線遅延モデルは，次式を用いる [9]．

$$del(n,s,i) = (c_{M1} \cdot W_n^{M1} + \sum_j C_j^n)(R_o^n + r_{M1} \cdot L_i^{M1}) + (c_{M2} \cdot W_n^{M2} + \sum_j C_j^n)(R_o^n + r_{M2} \cdot L_i^{M2})$$

ここで，

- $W_n^{M1(2)}$: 第 1(2) 層のネット n のスタイナ木の長さ
- $L_i^{M1(2)}$: 第 1(2) 層のソースからロード i のパスの長さ
- $c_{M1(2)}$: 第 1(2) 層の単位長当たりの配線容量
- $r_{M1(2)}$: 第 1(2) 層の単位長当たりの配線抵抗
- $\sum_j C_j^n$: ネット n のロード容量の総和
- R_o^n : ネット n に入力するセルの出力等価抵抗

以下に本稿で取り扱う概略配線問題を示す．

【概略配線問題】

入力： (1) セルの配置，(2) 各ネットの端子の集合，(3) 配線遅延の上限値

出力： 各ネットの概略配線経路

制約： (1) タイミング制約を満たす
(2)100%配線を行なう

目的関数： チップ面積→最小化

---

3 提案アルゴリズム

提案アルゴリズムは大きく分けて以下の 3 つのフェーズからなる．

フェーズ 1: 初期概略配線経路の決定，及びフィードスルーの挿入

フェーズ 2: 配線遅延の制約を満たすように概略配線経路を改良

フェーズ 3: 配線遅延の制約を満たした上で，チャネル密度の最小化を目的とする概略配線経路の改良

フェーズ 1 では，すべてのネットに対して，配線遅延に影響する 2 つの要素であるソース，ロード間の配線距離とネットの総配線長のトレードオフを考慮することにより初期概略配線経路を求め，必要ならばフィードスルーを挿入する．フェーズ 2 では，2 つのレジスタ間のパスを探索することによりタイミング解析を行ない，タイミング制約を満たしていないパス (クリティカルパス) 上の各ネットに対して概略配線経路の改良を行なう．この段階で，タイミング制約に違反しやすいパス上の各ネットの概略配線経路がある程度固定される．フェーズ 3 では，フェーズ 2 によりある程度固定されたパス上の経路以外の経路に対して，タイミング制約を満たし，チャネル密度が最小となるように概略配線経路の改良を行なう．このとき，ゼロスラックアルゴリズム [6] に基づきスラックを求め，求めたスラックの値より各経路の許容配線長を計算し，許容配線長を超えず，配線混雑度が最小となるように概略配線経路を改良する．

以降では，アルゴリズムの各フェーズについて述べる．

3.1 フェーズ 1: 初期概略配線

フェーズ 1 では，各ネットの初期概略配線経路の決定，及びフィードスルーの挿入を行なう．まず，すべてのネットに対して，等電位端子対を節点とする完全グラフを構成し，枝の重みとして節点間のマンハッタン距離を与える．次に各ネットの配線遅延の下界値を配置より計算し，遅延の小さなネットから順に端子を節点とする完全グラフ上で木を構成していく．ソース，ロード間の配線遅延は，ネットの総配線長，及びソース，ロード間の距離に影響されるため，ネットの総配線長が最小となる最小木と，すべてのソース，ロード間の距離が最小となる最短経路木のトレードオフを考慮した木を構成する必要がある．そこで，木を構成するアルゴリズムは，文献 [1] の Dijkstra の最短経路木を求めるアルゴリズムと Prim の最小木を求めるアルゴリズムを組み合わせたものを用い，以下の重みの小さな枝を順に取り込み各ネットの木を構成する (図 2)．

$$(\alpha \cdot L_i) + l_{ij}$$

ここで，L_iはソースからロード i までのパス長，l_{ij}は枝 (i,j) のマンハッタン距離，αは定数 $(0 \leq \alpha \leq 1)$ とする．その後，この木を配線混雑度に基づくコストが小さくなるようにグリッドグラフ G 上へ埋め込み，グリッドグラフ上でスタイナ木を構成し，フィードスルーを挿入

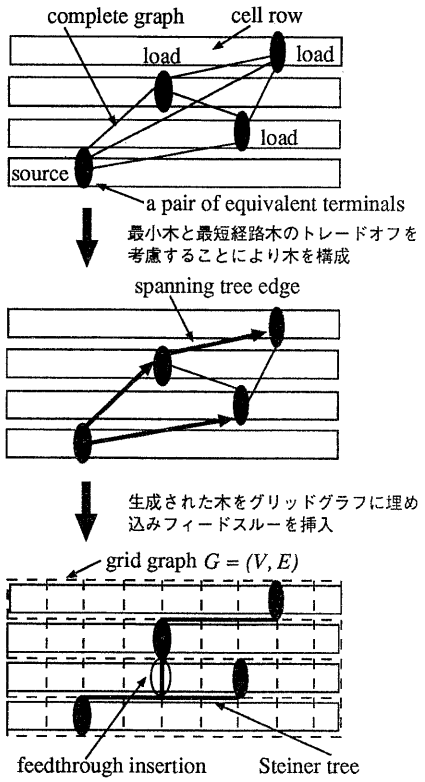

図 2 各ネットのスタイナ木の構成

する (図 2)．グリッドグラフ上へ埋め込む際の水平方向枝 $e \in E_H$のコスト $c(e)$ は次式を用いる．

$$c(e) = d(e)/D(e)$$

ここで，$d(e)$ はこの枝を通過するスタイナ木の数，$D(e)$ は e が属するチャネル内の水平方向枝の中で，通過するスタイナ木の数が最大の枝 e'における $d(e')$ の値とする．以下にアルゴリズムの各ステップを示す．

ステップ 1.1 すべてのネット $n \in N$に対して配置の結果より求めた遅延の下界値を計算する．

ステップ 1.2 遅延の下界値に基づき，ネットを昇順にソートする．

ステップ 1.3 N内のネットの中で遅延の下界値の最も小さいネット n に対して完全グラフ $G_n = (V_n, E_n)$ を構成し，各枝に対応する各端子間のマンハッタン距離を重みとして与える．

ステップ 1.4 G_n上で最小木と最短経路木のトレードオフを考えることにより，ネット n の木 T_nを求める．

ステップ 1.5 T_nの枝を配線混雑度を考慮することによりグリッドグラフ G 上の枝に埋め込む．

ステップ 1.6 $N = N - \{n\}$．$N = \phi$なら終了，そうでな

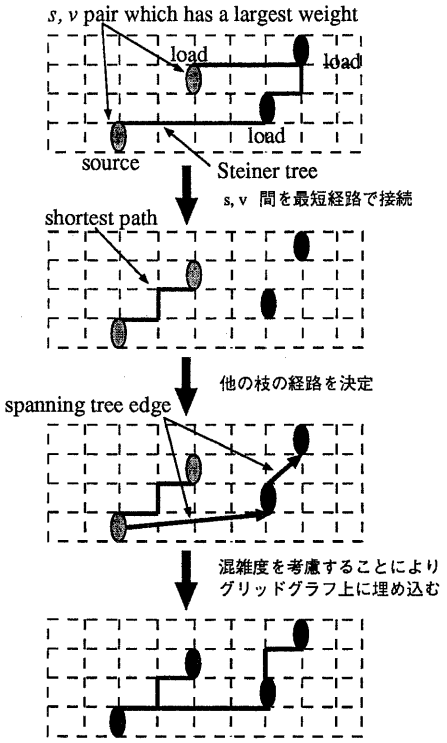

図 3 配線遅延に基づく概略配線経路の改良

ければステップ 1.3 へ.

3.2 フェーズ 2: 配線遅延に基づく配線改良

フェーズ 2 では，タイミング制約が与えられている 2 つのレジスタ間のすべてのパスに対して，タイミング制約を満たすように概略配線経路の改良を行なう．まず，タイミング制約に違反しているパス上の各ネットの経路に対して，(1) ネット n の経路 s，v 間の現在の遅延 $d_c(n,s,v)$，(2) その経路の遅延の下界値 $d_l(n,s,v)$，(3) その経路を通っているパス数 $Inf(n,s,v)$[18]，に基づく重みを以下のように定義する．

$$w(n,s,v) = Inf(n,s,v) \times (d_c(n,s,v) - d_l(n,s,v))$$

$d_l(n,s,v)$ は，配線遅延モデルの W を最小木の長さ，L をソースからロードまでの最短経路長とすることにより求める．このように重みを設定することにより，遅延の下界値との差が大きく，多くのパスの経路となっているネットが大きな重みをもつことになる．

この重みの最も大きいネットに対して，まず経路 (n,s,v) は s，v 間の最短経路で結び，配線混雑度を考慮してグリッドグラフ上に埋め込む．その他の経路に対しては，フェーズ 1 と同様のアルゴリズムを用いて最小木と最短経路木のトレードオフを考慮することにより木を構成し，それをグリッドグラフ上へ埋め込んでスタイナ木を構成する (図 3)．この枝 (s，v 間の経路) は以後定数回の改良の間はロックし，改良される枝として選ばれないようにする．この操作をすべてのパスがタイミング制約を満たすまで繰り返し行なう．こうすることで，制約違反を引き起こしやすいネットの経路をある程度先に決定することができる．以下にフェーズ 2 の各ステップを示す．

ステップ 2.1 制約に違反しているパスを求める．

ステップ 2.2 パス上の各ネットの経路に対して，$w(n,s,v)$ を計算する．

ステップ 2.3 $w(n,s,v)$ の最も大きいネットの s，v 間を最短経路で結び，グリッドグラフ上に埋め込む．

ステップ 2.4 ネット n の中で，s，v 間以外の経路に対しては，フェーズ 1 と同様のアルゴリズムにより再配線する．

ステップ 2.5 すべてのパスがタイミング制約を満たしていれば終了．そうでなければステップ 2.1 へ．

3.3 フェーズ 3: チャネル密度に基づく配線改良

フェーズ 2 において，タイミング制約を満たすように概略配線経路の改良を行なったので，フェーズ 3 では，タイミング制約を満たす範囲でチャネル密度が減少するように概略配線経路の改良を行なう．まず，文献 [6] のゼロスラックアルゴリズムによりパス上の概略配線経路の各枝に対して要求遅延時間と実際の遅延時間の差 (スラック) を計算する (図 4)．ここで，制約に違反しやすいパス上の経路ほど，要求遅延時間と実際の遅延時間の差が小さくなる．計算されたスラックは各枝の許容配線遅延と考えることができ，それに基づき各枝の許容配線長を計算する．そして，許容配線長を超えずチャネル密度に基づくコストが最も小さくなるような概略配線経路を選択し，この枝の経路を変更する (図 5)．グリッドグラフ上の水平方向枝 $e \in E_H$ のコスト $c(e)$ は次式を用いる．

$$c(e) = d(e)/D(e)$$

この操作を，すべてのスラックが 0 になるか，定数回改良が行なわれなくなるまで繰り返し行なう．

以下にフェーズ 3 の各ステップを示す．

ステップ 3.1 パス上の各経路に対して，要求遅延時間と実際の遅延時間の差 ($Slack(n,s,v)$) を計算する．

ステップ 3.2 経路上で最もスラックの値が小さい経路 (n,s,v) を選択する．

ステップ 3.3 経路 (n,s,v) を通り，パス上のスラックの値がすべて $Slack(n,s,v)$ と等しいパス $(n',s,u) \rightarrow \cdots \rightarrow (n,s,v) \rightarrow \cdots \rightarrow (n'',s,w)$ を求める．

ステップ 3.4 パス $(n',s,u) \rightarrow \cdots \rightarrow (n,s,v) \rightarrow \cdots \rightarrow (n'',s,w)$ 上の経路に対して，チャネル密度

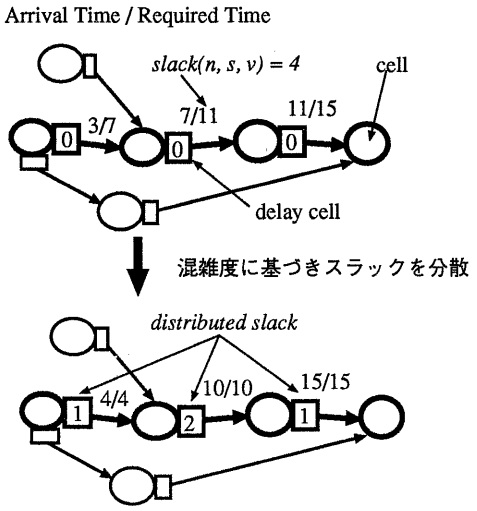

図 4 ゼロスラックアルゴリズム

に基づくコストを与え，そのコストに基づきスラック $Slack(n, s, v)$ の値を分散させる．

ステップ 3.5 分散されたスラックに基づき，各枝の許容配線長を計算する．

ステップ 3.6 チャネル密度に基づくコストを最小とし，許容配線長を超えないような最短経路を求め，この枝の概略配線経路を改良する．

ステップ 3.7 影響された他の経路の値を更新．

ステップ 3.8 すべての経路の処理が終っていない場合にはステップ 3.2 へ．

ステップ 3.9 すべての経路のスラックが 0 になるか，定数回改良が行なわれない場合には終了．そうでなければ，ステップ 3.1 へ．

4 シミュレーション実験

提案手法を SPARC Server 1000 上で C 言語を用いて実現し，シミュレーション実験を行なった．実験は，表 1 に示す 4 つのベンチマークデータを用いて行なった．ここで，fract, struct は MCNC ベンチマークデータ，C2, C7 は ISCAS ベンチマークデータである．配置は，C2, C7 に対しては RITUAL[16] を用いて行ない，fract, struct に対しては TimberWolfSC4.2c[15] を用いた．また詳細配線は，YACR2[14] を用いて行なった．

表 2は，提案手法と従来手法との比較結果である．ここで，従来手法の結果については，文献 [8] において記述されている結果を引用した．fract, struct に対する従来手法の実験結果については文献 [8] において報告されていないため，この部分は空白とした．実験の結果，提案手法は従来手法に対して，配線遅延に関しては平均 27.5%良

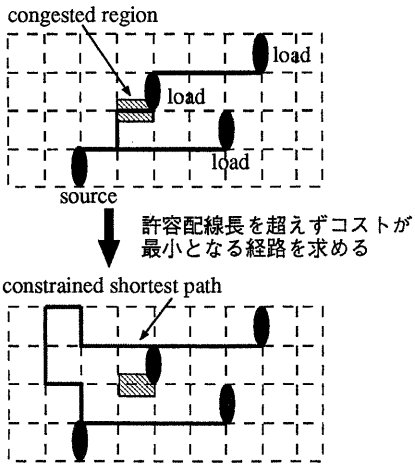

図 5 チャネル密度に基づく概略配線経路の改良

い解を求めることができた．しかしトラック総数に対する比較結果では，C2 に対しては，従来手法とほぼ同等の解を得ることができたが，C7 に対しては，従来手法に対して 7.5%多いトラック数が配線に必要となっている．これはフェーズ 3 において，(1) チャネル密度の改良が 2 端子間の改良である，(2) スラックを分散させることによってネットベースの改良に置き換えている，ことから概略配線経路の大幅な改良が行なわれないためであると考えられる．また，フェーズ 1，2 において，配線遅延に対する改良を重点的に行なっているためであるとも考えられる．よって，今後の課題としては，(1) フェーズ 3 の改良操作の多端子ネットへの拡張，(2) 各ネットの概略配線経路を同時に改良する手法の開発，(3) フェーズ 2 とフェーズ 3 の間にチャネル密度に対する改良フェーズを付け加える，などがあげられる．

表 3は，各ネットの初期概略配線経路を (1) 最小木 (MST) を用いて構成した場合，(2) 最短経路木 (SPT) を用いて構成した場合，及び (3) 最小木と最短経路木のトレードオフを考慮した木 (Tree) を用いて構成した場合，の 3 通りに対して比較実験を行なった結果である．実験の結果，最小木と最短経路木のトレードオフを考慮することにより木を構成した場合の方が，ほぼ同じチャネル密度で配線遅延の小さい結果を出すことがわかった．

5 あとがき

本稿では，レジスタ間のパスのタイミング制約を考慮し，チャネル密度の最小化を目的とする概略配線アルゴリズムを提案した．また，提案手法を計算機上で実現しシミュレーション実験を行なった結果について報告した．実験の結果，提案手法は従来手法に対して配線遅延に対しては平均 27.5%少ない解を求めることができ，提案手

表 1 実験データ

Data	#Cells	#Nets	#Const.	#Rows
fract	125	147	38	12
struct	1888	1920	127	20
C2	590	963	270	9
C7	2150	2465	405	18

#Cells:セル数, #Nets:ネット数
#Const.:制約数, #Rows:セル列数

表 2 実験結果

Data	提案手法			従来手法	
	ΣT	*Delay*	CPU	ΣT	*Delay*
fract	110	32.8	1.7	—	—
struct	301	110.3	27.1	—	—
C2	119	25.22	7.2	120	31.56
C7	366	58.25	18.3	340	89.46

ΣT:トラック数の総和, *Delay*:最大遅延 (ns)
CPU:計算時間 (秒)

表 3 木の構成方法に関する実験結果

Data	*MST*		*SPT*		*Tree*	
	ΣT	*Delay*	ΣT	*Delay*	ΣT	*Delay*
fract	110	33.0	112	36.2	110	32.8
struct	304	111.8	325	141.2	301	110.3
C2	120	25.2	147	32.7	119	25.2
C7	370	62.3	435	92.8	366	58.3

ΣT:トラック数の総和, *Delay*:最大遅延 (ns)

法の有効性を確認することができた．しかし，チャネル密度に対しては従来手法より良い解を求めることができず，より一層の改良を行なうアルゴリズムを開発する必要があることが分かった．今後の課題としては，(1) チャネル密度の改良フェーズの追加，(2) ネットの処理順序についての考察，及び (3)3 層配線モデルへの拡張，がある．

謝辞

本研究を行なうにあたり，プログラムの作成にご協力頂いた本学学部生岡本誠君に感謝致します．

参考文献

[1] C. J. Alpert, T. C. Hu, J. H. Huang and A. B. Kahng: "A direct combination of the Prim and Dijkstra constructions for improved performance-driven global routing," Proc. of International Symposium on Circuits and Systems, pp. 1869–1872 (1993).

[2] K. D. Boese, A. B. Kahng and G. Robins: "High-performance routing trees with identified critical sinks," Proc. of 30th Design Automation Conference, pp. 182–187 (1993).

[3] R. C. Carden and C.-K. Cheng: "A global router using an efficient approximate multicommodity multiterminal flow algorithm," Proc. of 28th Design Automation Conference, pp. 316–321 (1991).

[4] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh and C. K. Wong: "Provably good performance-driven global routing," IEEE Trans. Comput.-Aided Design of Integrated Circuits & Syst., Vol. CAD-16, No. 6, pp. 739–752 (1992).

[5] J. Cong and B. Preas: "A new algorithm for standard cell global routing," Proc. of International Conference on Computer-Aided Design, pp. 176–179 (1988).

[6] P. S. Hauge, R. Nair and E. J. Yoffa: "Circuit placement for predictable performance," Proc. of International Conference on Computer-Aided Design, pp. 88–91 (1987).

[7] X. Hong, T. Xue, E. S. Kuh, C.-K. Cheng and J. Huang: "Performance-driven Steiner tree algorithms for global routing," Proc. of 30th Design Automation Conference, pp. 177–181 (1993).

[8] J. Huang, C.-K. C. Xian-Long Hong and E. S. Kuh: "An efficient timing-driven global routing algorithm," Proc. of 30th Design Automation Conference, pp. 596–600 (1993).

[9] E. S. Kuh and M. Shih: "Recent advances in timing-driven physical design," Proc. of Asia-Pacific Conference on Circuits and Systems, pp. 23–28 (1992).

[10] H. Kusumoto: "A performance driven layout system using path delays for row based VLSIs," Master thesis, Graduate School of Engr. Hiroshima Univ. (1994).

[11] A. Lim and S.-W. Cheng: "Performance oriented rectilinear Steiner trees," Proc. of 30th Design Automation Conference, pp. 171–176 (1993).

[12] G. Meixner and U. Lauther: "A new global router based on a flow model and linear assignment," Proc. of International Conference on Computer-Aided Design, pp. 44–47 (1990).

[13] S. Prasitjutrakul and W. J. Kubitz: "A performance-driven global router for custom VLSI chip design," IEEE Trans. Comput.-Aided Design of Integrated Circuits & Syst., Vol. CAD-11, No. 8, pp. 1044–1051 (1992).

[14] J. Reed, A. Sangiovanni-Vincentelli and M. Santomauro: "A new symbolic channel router:YACR2," IEEE Trans. Comput.-Aided Design of Integrated Circuits & Syst., Vol. CAD-4, No. 3, pp. 208–219 (1985).

[15] C. Sechen and K. Lee: "An improved simulated annealing algorithm for row-based placement," Proc. of International Conference on Computer-Aided Design, pp. 478–481 (1987).

[16] A. Srinivasan, K. Chaudhary and E. S. Kuh: "RITUAL: A performance driven placement algorithm for small cell ICs," Proc. of International Conference on Computer-Aided Design, pp. 48–51 (1991).

[17] W. Swartz and C. Sechen: "A new generalized row-based global router," Proc. of International Conference on Computer-Aided Design, pp. 491–498 (1993).

[18] M. S. Tanaka, Y. Miyazawa, H. Aizawa and M. Minowa: "Minimum delay placement with influence of nets and hierarchical clustering," Proc. of IEEE Custom Integrated Circuits Conference, pp. 663–666 (1994).