設 計 自 動 化 76－5
（1995．7．20）

バッファ挿入を伴う等遅延クロックツリー生成アルゴリズム

福田 浩之　枝廣 正人*　油井 信康
山崎 博義　石塚 昭夫

日本電気(株) 〒211 神奈川県川崎市中原区下沼部 1753
* 〒216 神奈川県川崎市宮前区宮崎 4-1-1

あらまし LSI レイアウト設計におけるクロックスキューを最小化するためのバッファ挿入を伴う等遅延クロックツリー生成のアルゴリズムに関して報告する。クロックツリーの途中へバッファを挿入することによってクロック遅延の減少、通常配線幅で配線可能なことでのレイアウトの自由度の増加等の効果が期待できる。ここで報告するアルゴリズムは等遅延配線の配線長を見積もって、それをバッファ挿入のためのクラスタリングの際に利用することによってバッファ間の負荷容量をバランスさせることを特徴としている。さらに LSI レイアウト設計で実現するための 2 点間ルータに関しても報告する。

和文キーワード LSI CAD、クロック配線、等遅延配線、バッファ挿入

Zero-skew Clock Tree Synthesis with Buffer Insertion

Hiroyuki Fukuda　Masato Edahiro*　Nobuyasu Yui
Hiroyosi Yamazaki　Akio Ishizuka

NEC Corporation
1753 Shimonumabe, Nakahara-ku, Kawasaki, Kanagawa 211, Japan
* 4-1-1 Miyazaki, Miyamae-ku, Kawasaki, Kanagawa 216, Japan

Abstract This paper presents a zero-skew clock tree routing method which includes buffer insertion and point to point router. Buffer insertion is important technique for clock tree synthesis to minimize clock delay and skew. And it also increases layout flexibility because it allows regular width wirings for clock nets.

The proposed buffer insertion method which balances loading of buffers consists of wire capacitance estimation based on zero-skew routing and optimization of clusters.

Key words LSI CAD, Clock Routing, Zero-Skew Routing, Buffer Insertion

1 はじめに

近年のLSIの動作速度の高速化によって、フリップフロップ(FF)の同期動作が重要な課題となっている。そのためには各FFに入力するクロック信号の遅延のばらつき(クロックスキュー)を最小に押えることが重要である。

また同時に、配線幅の微細化に伴い素子間の信号の遅延のうち配線遅延が占める割合がゲートの遅延に比較して大きくなって来ている。このためクロックスキューを最小にするためにはクロックドライバの出力端子から各FFまでのクロックネットの配線遅延をコントロールすることが必要である。

こうした状況で、従来より等遅延クロックツリー生成(等遅延CTS)アルゴリズムが報告されている[2], [6]。ところが、クロックネットを1つのネットで配線すると、エレクトロマイグレーションを防ぐために太幅配線が必要になり、クロック波形の鈍りやクロック遅延の増大、一般信号配線の配線性の悪化等の問題が生じる。

そこで、クロックの供給元からFFとの間に複数段のバッファをツリー状に挿入しそれらの間を等遅延に配線する事によって全体としてのクロックスキューを最小にしようとする等遅延CTSに関してアルゴリズムが報告されている[7], [8]。

しかし、バッファ挿入を等遅延配線と独立に行なう方法では挿入されたバッファ間で配線容量も含めた付加容量を均等化することは困難で、それが原因となってスキューが生じたり、スキュー合わせのための冗長配線を行なったとしても配線長やクロック遅延が増加する問題があった。

そこで本報告では等遅延配線による配線容量の見積りをバッファ挿入のためのクラスタ作成の際に行なうことによってより負荷容量がバッファ間で均等になるようにすることを特徴をした等遅延CTSのアルゴリズムとその実際のLSIのレイアウト設計への実用化の結果について述べる。

2 クロックネットのレイアウトモデル

本報告で述べるクロックネットのレイアウトモデルでは、クロック配線は図1で示す通り、複数段のバッファツリーを作成し、各バッファからの配線については入力端子までの等遅延配線を行なうことによってルートのバッファからの末端のFFまでの等遅延を保証するものである。

バッファの挿入に関しては以下のことを考慮して行なっている。

1. バッファツリーの各段ごとに同一の駆動能力を持つバッファを使って各バッファの負荷容量を合わせ、各段ごと等遅延を可能にしている。
2. 遅延の最小化を図るため、FFの配置位置によってツリーの構成、バッファの配置位置を最適化する。
3. FFはチップ上に分散して配置されているが一様に配置されているとは限らない。このためFFが密に配置されている場所と疎に配置されている場所とではバッファの挿入数を変える。
4. 各バッファに関して駆動できる負荷容量の上限があらかじめ決められている。バッファ挿入はこの制約の元で行われる。

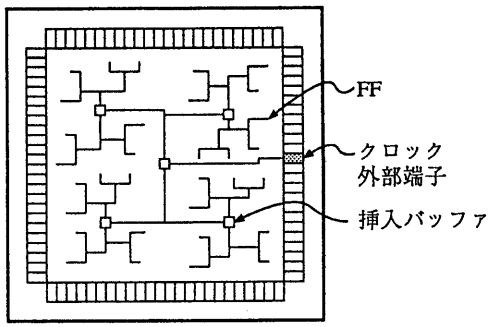


図1: クロックネットのレイアウトモデル

各バッファからの配線では以下のことを考慮して行なっている。

1. 図2で示すように各バッファの出力端子から下位のバッファまたはFFまでは2分木状の配線が行われる。この中で各バッファごとの等遅延配線を行っている。
2. 配線幅はブロックの配置に影響を及ぼさず、また一般信号配線の障害にもなりにくいように通常信号線と同じにする。
3. 配線層は通常信号線と同じものを使って配線される。ただし、遅延を最小化するために通常信号に先だって配線され、また配線抵抗の小さな配線層を優先的に使用する。

3 レイアウト手法

3.1 等遅延CTSを含むLSIレイアウト設計のフロー

以下に本等遅延CTSをGA/CBIC自動レイアウトシステムの中で実現した際の位置づけを示す。

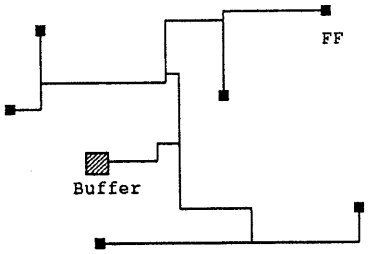


図 2: 等遅延配線

1. 初期ネットリスト
   自動レイアウトシステムの入力ネットリストではクロックネットは、そのクロックを入力する全てのFFに直接接続する形で表現されている。すなわち、クロックツリーを構成するバッファは入力ネットリストには含まれていない。このため、レイアウトの処理を実行する以前にチップへの収容性や遅延を見積もる際には、等遅延CTSシステムで追加されるバッファ数や配線量を見積もった上で行なっている。

2. 素子の配置
   クロックに接続するFFも含めて全素子の自動配置を行う。この時クロックネットの接続は考慮しない。なぜなら、CTSが配置位置に基づいてバッファ挿入を行い、ツリーを作成するので、この時点でクロックネットを考慮した自動配置は一般ネットの配線性も考慮すると意味が無いためである。また、一般にピン数が大きなネットを扱う事は自動配置システムにとって負担が大きく処理時間が増大してしまうという問題があるためでもある。

3. 等遅延CTS
   一般ネットの配線が行なわれる前に、自動配置の結果のFFの配置位置を基にしてバッファの挿入、等遅延配線を行う。一般ネットの配線が無いことで配線に自由度が大きくスキューや遅延の最小化にとって有利となる。

4. 配線
   CTSで既に配線された配線は固定した上で一般ネットの配線を行う。それは、クロックネットの配線は等遅延の目的で最適化されているため、迂回配線等が生じている。一般ネットの配線の際にクロック配線が配線性を基にして最適化され、再配線されてしまうのを防ぐことが必要になるためである。
   ただし、等遅延CTSでの配線では一般ネットの配線との相互関係を完全に考慮する事は困難であるため、この時点になってクロックネットの配線の小さな経路変更が必要になる場合もある。

5. バック アノテーション
   CTS結果のクロックネットの情報も含む配線遅延の情報を遅延解析ツール、論理合成ツールにインターフェースする。この時、初期ネットリストと対応関係がとれるように、CTSによって追加されたバッファの遅延情報とツリーの階層によって分割された配線遅延の情報を足し合わせて初期ネットリストのクロック信号に対する情報としてインターフェースしている。

3.2 等遅延CTSの処理概要

今回報告する等遅延CTSの処理は大きくバッファの挿入の処理と1つのバッファが駆動するクラスタ内での配線の処理とに分かれる。バッファの挿入処理ではFFに対して階層的なクラスタを作成することが主な内容である。バッファ内での配線は2分木状の配線で等遅延配線を行なうものである。実際の配線処理では特殊な2点間ルータを使っている。

さらに、バッファの挿入の処理と配線の処理とは機能的に独立しているのではなく、密接な関係を持っている。すなわち、バッファの挿入の際のクラスタ作成においては等遅延配線での配線長を見積もり、それを基に負荷容量をバランスさせるようにしている。また等遅延配線の際にはバランスされなかった負荷容量を迂回配線を行なうことによって解決している。

次節から各処理に関して詳しく述べる。

3.3 クラスタ分割

既に配置済のFFに対して、バッファ挿入の基準となるクラスタを作成する。多段のバッファを挿入する場合にはこの処理を下位レベルから順に行う。

クラスタ分割の目的としては以下のものがある。

1. 配置位置の近いFFどうしが同一のクラスタに含まれるようにする。
   クロックネットの配線長を短くすること、遅延時間を小さくすることに効果がある。配置位置の近さはマンハッタン距離を基に評価する。

2. クラスタ相互で負荷容量のばらつきを小さくする。
   負荷容量のばらつきがあると、バッファ内での等遅延配線が可能になったとしても同一レベルのバッファ相互での遅延のばらつきが大きくなってしまい、これを冗長配線を作成することによって補正しようとすると配線長、遅延時間ともに悪化の原因となる。
   負荷容量は入力端子容量と見積りの配線容量との和で見積もる。

3. 各バッファ毎に定められた負荷容量制限を満足する。

4. 挿入するバッファの個数を遅延時間を最小にするように最適化する。
   一般に、ある段の挿入バッファ数を増やすことは、ドライブ能力の向上による遅延の減少の効果とは逆に、入力端子容量の増大に伴う前段の遅延の増加につながる。
   このためルートのバッファからFFまでの総遅延を最小化するための複数段の遅延の関係を考慮したバッファの個数の最適化が必要になる。

これらのうち1-3の制約条件制約条件を満たすクラスタ分割を実現するために以下の手順で実行している。

Step1. FF相互のマンハッタン距離を基に、互いに近いFFを併合していくことによって初期クラスタを作成する。
Step2. 配線容量を見積もって、各クラスタの負荷容量を計算する。
Step3. while(負荷容量制限が満たされない or 負荷容量ばらつきが大きい){
  Step3-1. 最大の容量を持つクラスタからFFのflowを流し最小の容量を持つクラスタにFFを付加する
  Step3-2. 配線の見積もりと負荷容量の再計算
  }

図3はStep3-1でのFFのflowの流れの例である。クラスタAの付加容量が大きすぎ、クラスタDの容量が小さすぎる時、位置的にクラスタA, Dの間にあるクラスタB, Cを経由して、クラスタ間でFFの付替を行なうことによって付加容量の平均化を図っている。

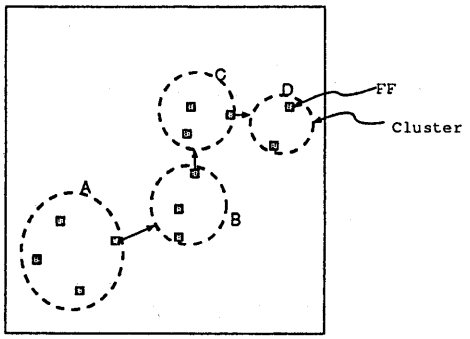


図3: FFのflowの流れ

Step3-2では3.5節で述べる等遅延配線を仮想的に行なうことによって配線長や配線容量を見積もっている。配線禁止の情報を考慮して実際に配線しているのではないため若干の誤差は出るが、この時点での見積もりとしては十分な精度を持つものである。

3.4 バッファの配置

作成されたクラスタに対してバッファをネットリスト上に追加するとともに、既に配置まで行われているレイアウトに対して追加配置する。配置位置は仮想的に等遅延配線を行って、その2分木の根の位置とする。

既にFF及び一般素子に関しては配置が済んでいるためバッファの配置では既配置の素子との重なりを除去する処理が必要である。また、バッファやFFの配置は遅延時間やスキューに直接影響するため大きく位置を移動することは避けなければならない。

既配置の素子の配置位置をできるだけ変えずにバッファ挿入することを以下の手順で実現した。

Step1. 対象バッファをキューに登録する
Step2. while(キューが空でない限り){
    先頭の素子に対して
  Step2-1. コストが最小な配置位置を探し、その位置に配置をする。コストは移動距離が大きいほど、また移動回数の多い素子に重ねようとするほど大きくなるようする。
  Step2-2. 置かれた位置に元からある素子を未配置とし、キューに登録する
  }

Step2-1でのコストは最適配置位置(挿入バッファの場合)または最初におかれていた場所からのマンハッタン距離 d と、重ねようとする素子の大きさ s とそれまでの移動回数 m とから以下の式で計算している。

$\alpha \cdot d + \beta \cdot \sum s + \gamma \cdot \sum (m+1)$

α, β, γ は定数であるが、バッファの場合に限って、$\beta = \gamma = 0$ として等遅延配線から求めた最適配置位置に配置するよう優先度を上げている。

図4はバッファ挿入に伴う重なりの除去の例である。(a)が配置前、(b)が配置後である。

3.5 等遅延配線

1つのバッファからの配線はすでに報告済のアルゴリズム[6]を基本アルゴリズムとして行なっている。

バッファからFF(又は中間バッファ)までの配線は図5のような2分木状の配線を行う。ここで各分岐点間の配線を π 型の分配RCモデルによってモデル化し、バッファからFFまでの配線遅延をElmore遅延モデル[1]で近似することを前提とする。

この場合、各FFまでの等遅延配線は、「ゼロスキュー併合」を行って配線と分岐点の位置を確定することを2分木の下位から順に最上位(バッファの出力端子)まで行うことによって実現される[2], [6]。

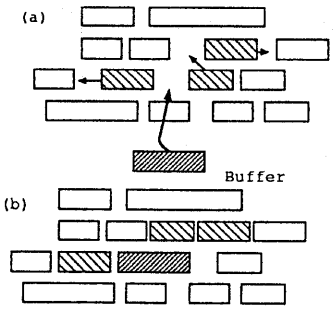


図 4: バッファ挿入

ゼロスキュー併合とは、ある分岐点 v の下位にある2つの2分木の等遅延配線が既に行われており、下位の2分木内での配線遅延、負荷容量が既知の場合にその2つの2分木を結ぶ配線と上位への分岐点 v を決定することをいう。

図6はゼロスキュー併合における π 型モデルを表している。l_1, l_2 は分岐点 v から各下位の部分木までの配線長、r, c はそれぞれ配線の単位長あたりの配線抵抗と配線容量とする。また、C_1, C_2 は各部分木の配線容量、t_1, t_2 は各部分木内での配線遅延である。

このとき分岐点 v 以下を等遅延にするためには、Elmore 遅延モデルによる下の式を満足する必要がある。

$$rl_1(cl_1/2 + C_1) + t_1 = rl_2(cl_2/2 + C_2) + t_2$$

この式を満足する l_1, l_2 で $l_1 + l_2$ を最小とするものを求め、そこから分岐点 v の座標を決定している。

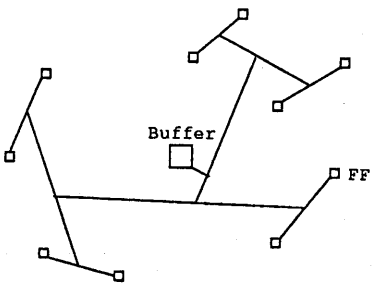


図 5: クロックツリー

以上の議論は配線層ごとに配線抵抗や配線容量が異なる場合にも拡張することが可能であり、その場合でも上と同様にゼロスキュー併合を行なって、等遅延配線を行なうことができる。

さらに、今回の報告のようにバッファの挿入を伴うクロックツリーにおいては各バッファ内でのスキューの

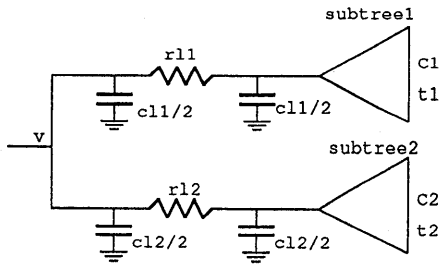


図 6: π モデル

最小化と同時に、同一レベルでのバッファ相互でスキューを最小化することが必要になる。これを実現するために、最大の遅延を持つバッファに遅延値を合わせる目的で他のバッファで等遅延配線の2分木の根の部分で迂回配線を行なっている (図 7)。しかし、この迂回配線は配線長や遅延値の増加につながるため、最小にすることが必要である。このために、クラスタリングの段階で付加容量を見積って、遅延時間の見積もりとそれによるクラスタリングの最適化を行なうことによって迂回配線の量を減らしている。

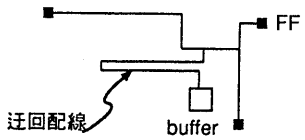


図 7: 迂回配線

3.6 クロックツリーの構成

前節ではクロックネットに関して2分木の接続関係が与えられた時に等遅延配線を行なう方法に関して述べた。一般に任意の2分木の接続関係に対して前節のアルゴリズムで等遅延配線を行なうことによってスキューを0にすることは可能である。しかし配線長や遅延値は2分木の構造に大きく依存することになる。

配線長や遅延値を最小にする2分木の構造の作成方法に関しては [3] で提案された2分木を構成するノード (FF) 群の中で相互に最も近い位置にあるノード (FF) 対から順にボトムアップに併合を行なって2分木を構成する方法が効果的である。

ランダムに配置されたノード群の中から最も近い位置にあるノード対を見出すアルゴリズムについても多数

提案されているが[5]、ここでは、バケット法によってノード間の近さの関係を表すグラフを作成し、それを基にしてボトムアップのノードの併合を行なうアルゴリズム[4]を採用した。このアルゴリズムではFF数に対してリニアなオーダの計算時間で2分木の構造を決定することが可能である。

3.7 2点間ルータ

2点間ルータは、任意の2点間の配線を配線禁止の情報を考慮して行なうことで、前節までで説明した等遅延配線を実際のレイアウト上で実現するものである。

2点間ルータに求められる機能は以下のような物である。

1. 配線で使用する層の指定ができる。
   クロックネットの配線で3層以上の配線層が利用できる場合、同一配線方向の配線に対して利用する配線層に対して複数の選択枝がある。このような場合、より単位長あたりの配線抵抗の小さな配線層を選択的に使ってに配線を行なうことによって配線遅延を最小化することが可能である。

2. 可能な限り最短経路で配線する。
   これは、等遅延配線での配線長の見積もりと実際の配線が大きく異るとスキューの悪化の原因となるためである。

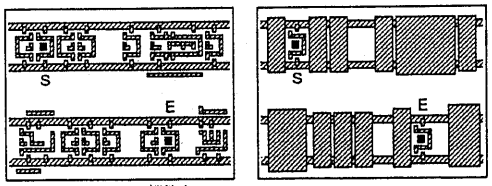
3. 高速に実行できる。
   これは、等遅延CTSでは配線の2分木の各枝ごとに2点間ルータを実行し、したがって2点間ルータの実行回数はFFの数の約2倍と見積もられ、この回数の多さから2点間ルータが等遅延CTS全体の処理時間の大きな割合を占めるからである。

これらの課題を満たすために、ラインサーチ法を基本アルゴリズムとして用い、その中で探索線分の優先順位づけを行なって1, 2 を実現し「アウトライン機能」を用いることによって3 を実現した。

この中で特にアウトライン機能に関して述べる。2点間ルータでは、配線禁止や端子、既配線といった図形の数がルータの処理時間に大きな影響を与える。特に大きな配線領域で配線を行なう場合にこれが顕著となるため、処理する図形数を減らす必要がある。アウトライン機能は、配置された機能ブロックを単位として、配線禁止の情報の多い1層部分に関して、

- 配線の始点と終点の近辺のブロックが含んでいる図形はそのまま扱う。
- それ以外のブロックが含んでいる図形は、それらを囲む一つの矩形にまとめて扱う。

という処理を行なうものである。



(a)アウトライン機能なし　(b)アウトライン機能あり

図 8: アウトライン機能

例えば、図8(a)で、■で表されている始点Sと終点Eを結ぶとすると、図中の全ての矩形をそのまま扱ったのでは処理する図形数の増大を招くので、同図(b)のように始終点の周辺を除いてはブロックを囲む矩形にまとめてしまう。通常、配線は始終点の周辺以外でブロックの内部を通る必要はないので、このように禁止や端子図形を扱っても配線品質に問題はない。

この機能によって、長い配線を行なう時の処理時間やメモリ使用量を押えることができた。

4 適用結果

本論文の提案の方式をGA/CBICのレイアウトに適用した結果を示す。

4.1 スキューの評価

表1は、CMOSゲートアレイのデータに対して提案の方式で3段のバッファを挿入し、等遅延配線を行なった結果のスキューの評価結果である。最終段のFFまでのクロックスキューは概ね100ps以下で、これはLSIの高速動作ために期待される性能を満足する。スキューが生じる原因としては等遅延配線において、配線禁止図形の考慮が行なわれておらず、2点間ルータで実際に配線した際に予期しない迂回が生じるためである。

表 1: スキューの評価

	FF数	遅延(ps) 最小	遅延(ps) 最大	skew
data1	816	1142	1193	51
data2	3546	1978	2067	89
data3	14616	2495	2575	80

表2は、このうちFF数14616のデータ(data3)での挿入バッファの段ごとのルートからの遅延とスキューの評価結果である。

表 2: 挿入バッファの段ごとのスキューの評価

ツリーのレベル	バッファ (FF) 数	遅延 (ps) 最小	遅延 (ps) 最大	skew
挿入バッファ (1 段)	4	675	677	2
挿入バッファ (2 段)	16	1416	1419	3
挿入バッファ (3 段)	256	2000	2034	34
FF	14616	2495	2575	80

データ　FF 数 14616 個

4.2 等遅延配線と配線量の増加

表 3は、FF 数 3546 のデータ (data2) での等遅延配線のスキューの削減に対する効果を示す。ある回路に対してバッファ挿入のみ同等に行なって等遅延配線の有無によってスキューの削減の効果を評価した。スキューの削減の効果はあるものの配線長の増加が生じている。一般に等遅延配線では配線長の増加とそれによる収容性の悪化が起こるのでレイアウト前の収容性の見積りの際には注意が必要となる。

表 3: 等遅延配線の効果と配線量の増加

	スキュー	配線長 クロック	配線長 全体
等遅延配線なし	415[ps]	8969 (100)	463196 (100)
等遅延配線あり	89[ps]	12675 (141)	473984 (102)

データ　FF 数 3546 個

4.3 アウトライン機能

表 4は、3.7節で述べた 2 点間ルータのアウトライン機能の効果である。処理時間と使用メモリ量を比較しているが、特に使用メモリの削減の効果が大きい。

表 4: アウトライン機能の効果

アウトライン	処理時間	使用メモリ
機能なし	15h	370MB
機能あり	7h	45MB

データ　FF 数 5649 個
マシン　95 Mips RISC workstation

5 まとめ

本報告では等遅延配線による配線容量の見積りをバッファ挿入のためのクラスタ作成の際に行なうことによって、負荷容量がバッファ間で均等になるようにすることを特徴としたバッファ挿入を伴う等遅延 CTS のアルゴリズムを紹介した。

さらに配線禁止を考慮する 2 点間ルータの機能を加え、それを実際の LSI のレイアウト設計に実用化して、その結果からこの等遅延 CTS のアルゴリズムの有効性を示した。

参考文献

[1] P. Penfield and J. Rubinstein, "Signal delay in RC tree networks," *Proc. of the 19th Design Automation Conference*, 1981

[2] R. S. Tsay, "Exact Zero Skew," *Proc. of the 1991 International Conference on Computer Aided Design*, 1991, pp.336-339

[3] M. Edahiro, T. Yoshimura, "Minimum Path-Length Equi-Distant Routing," *Proc. of 1992 IEEE Asia-Pacific Conference on Circuits and Systems*, 1992, pp.41-46

[4] M. Edahiro, "An Efficient Zero-Skew Routing Algorithm," *Proc. of the 31st Design Automation Conference*, 1994, pp.375-380

[5] M. Edahiro, "A Clustering-Based Optimization Algorithm in Zero-Skew Routings," *Proc. of the 30th Design Automation Conference*, 1993, pp.612-616

[6] 枝廣, "スキューをゼロにする配線における遅延最小化," 信学技報, VLD93-53, 1993, pp.15-22

[7] 高野、南、"クロックスキュー最小化のための付加容量均一化クラスタリング手法," 情報処理学会第 *43* 回全国大会, 3R-9, 6-203

[8] S. Pullela, N. Menezes, J. Omar, and L. T. Pillage, "Skew and Delay Optimization for Reliable Buffered Clock Trees", *Proc. of the 1993 International Conference on Computer Aided Design*, 1993, pp.556-562