設 計 自 動 化 83－11
1997. 2. 14)

スキュー制御クロックネットワークの構成

井上 一紀　　高橋 篤司　　梶谷 洋司

東京工業大学 電気・電子工学科
〒 152 東京都目黒区大岡山 2-12-1

梗概： 各レジスタへのクロック信号の到達時刻を指定することにより，回路のクロック周期をゼロスキュー配線による最小クロック周期よりも小さくすることが可能である．しかし，各レジスタのクロック信号の到達時刻を要求された値に指定するクロックネットワークを構成するスキュー制御配線が新しい問題として浮上してくる．本研究はスキュー制御配線の実現のために，トポロジーの選択，配線遅延の制御，中間バッファの挿入を行う．この手法により，総配線長を過度に増加させることなく，このようなクロックネットワークを構成することができることを実験結果によって示す．

Skew Control Clock Network Routing

Kazunori INOUE , Atsushi TAKAHASHI and Yoji KAJITANI

Department of Electrical and Electronic Engineering,
Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo 152, Japan

abstract: If the clock arrival time to each register is controlled, the clock-period in a circuit can be shorter than the minimum clock-period in Zero-Skew Routing. To realize the idea, we propose the Skew Control Routing to construct a clock network such that the clock arrival time to each register is set to the required value. We generate a good topology of the clock network, control the propergation delays on wire, insert intermediate buffers. Experimental results show that this method constructs clock network without excessive wire length.

1　はじめに

近年のVLSIの大規模化，高集積化に伴い，クロック信号の分配に関して様々な問題点が指摘されている．単相クロックを用いる同期式回路は，回路中のすべてのレジスタに同一周期のクロック信号が同時に到着することを前提としている．しかし無策なクロック信号の分配はクロックスキューと呼ばれる遅延時間のばらつきを生じさせる．これはクロック周期の短縮の妨げになり，最悪の場合には回路の誤動作を引き起こす．このため，同期式回路の高速動作を実現するために，回路中のすべてのレジスタにクロック信号が同時に到達するクロックネットワークを設計するゼロスキュー配線手法によってクロック配線を行うことが一般的であった [1, 2].

しかし，製造上の制約や誤差によって厳密にゼロスキューを実現することは不可能である．従って事実上は，クロックスキューをある一定の範囲内に抑えることにより誤動作を防ぐ手法が採られている [3]. それに対して，同一周期のクロック信号が同時刻にレジスタに到達することを要求しない準同期式回路構成がある．この構成において各レジスタへのクロック到着時刻を自由に指定することが可能であると仮

定すると, 同期式回路の最小クロック周期以下で回路を動作させることが可能である [4].

本研究は準同期式回路上で各レジスタにクロック信号が, クロック時差と呼ばれる基準時刻からの差で指定された任意の時刻に到達するクロックネットワークを構成することを目的とする. その実現方法は様々考えられるが, ここではクロック信号を1つのクロックソースから2分木を用いて分配するクロックネットワークを構成するスキュー制御配線手法を提案する.

2分木によるスキュー制御配線とゼロスキュー配線はよく似た性質を持っている. その点を踏まえて, ゼロスキュー配線手法である Clustering-Based Algorithm [2] を応用したスキュー制御配線手法を提案する.

2 クロック配線

クロック配線手法として一般的なゼロスキュー配線手法を示し, 提案手法であるスキュー制御配線手法を定義する.

2.1 ゼロスキュー配線

クロックソース s_0, クロックピンの集合 $S = \{s_1, s_2, \cdots, s_n\}$ が与えられたとする. この時, クロックソースから, すべてのクロックピンへの遅延時間が等しいクロックネットワークをゼロスキュー配線と呼ぶ.

クロックネットワークはクロックソースをルートとし, クロックピンをリーフとする2分木であるとする. ゼロスキュー配線では, 任意の内部点 vから v をルートとする部分木 T_vに含まれるすべてのクロックピンへの遅延時間は等しい. 従って T_vの負荷容量の総和を $C(v)$, vの子を v_1, v_2とし, vから v_1, v_2への配線長をそれぞれ l_1, l_2, 単位長さあたりの抵抗値を r, 容量値を c とすると, vから T_vに含まれるクロックピンへの遅延時間 $t(v)$ は図1に示すπモデルを用いて以下のように表せる.

$$t(v) = rl_1\left(\frac{cl_1}{2} + C(v_1)\right) + t(v_1) \quad (1)$$

$$= rl_2\left(\frac{cl_2}{2} + C(v_2)\right) + t(v_2) \quad (2)$$

$$C(v) = C(v_1) + C(v_2) + c(l_1 + l_2)$$

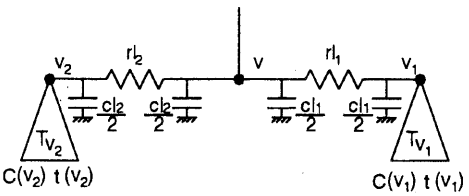

図 1: πモデル

各クロックピン s_iは負荷容量 C_{L_i}を持っており, $t(s_i) = 0, C(s_i) = C_{L_i}$であるものとして扱う. また, 点 v_1上にバッファの挿入が行われた場合には, $C(v_1) = C_{buf}$, $t(v_1) = t(v_1) + D_{buf}$とする. ここで, C_{buf}, D_{buf}はそれぞれバッファの入力容量, 遅延時間である.

2.2 スキュー制御配線

スキュー制御配線では, ゼロスキュー配線と異なり, 任意の内部点 vから部分木 T_vに含まれるすべてのクロックピンへの遅延時間は等しいとは限らない. そこで, 任意の内部点 vから部分木 T_vに含まれるクロックピン s_iまでの遅延時間を $t_{s_i}(v)$ とする.

クロックソース s_0, クロックピンの集合 $S = \{s_1, s_2, \cdots, s_n\}$ 及び, 各クロックピンに対してクロック時差 $dif(s_i)$ が与えられたとする. ただし一般性を失わず, $0 \leq dif(s_i) < C_P$ (C_Pはクロック周期) とする. 任意のクロックピンの対 s_i, s_jに対し, $t_{s_i}(s_0) - t_{s_j}(s_0) = -(dif(s_i) - dif(s_j))$ となるクロックネットワークをスキュー制御配線と呼ぶ. この時, クロックソース s_0を出発したクロック信号が, $dif(s_k) = 0$が割り当てられたクロックピン s_kに到達するまでの時間を t_{st}とすると, s_0から各クロックピン $s_i \in S$への遅延時間 $t_{s_i}(s_0)$ は, $t_{s_i}(s_0) = t_{st} - dif(s_i)$ となる.

クロックネットワークが2分木である時, $t(v)$ を vからクロックピン s_iまでの遅延時間に, s_iのクロック時差 $dif(s_i)$ を加えた時間 $(t_{s_i}(v) + dif(s_i))$ とすると, スキュー制御配線においても式 (1),(2) は成立する.

以降, クロックネットワークは2分木であるものとして扱う.

2.3 クロックネットワークのトポロジー

ゼロスキュー配線では, クロックネットワークのより良いトポロジーを生成するために, 様々な手法が検討されているが, トップダウンパーティションによる方法と, ボトムアップマッチングによる方法に大別される.

ゼロスキュー配線では総配線長の最小化, 遅延時間の最小化を目指し, チップ上で隣接するクロックピンを接続する. しかし, スキュー制御配線ではチップ上で隣接するクロックピンが, 一般に大きなクロック時差を持っているので, 単純に隣接するクロックピンを接続してしまうと, そのクロック時差の調整のために大きな迂回配線が必要となる. これは, 総配線長を大きく増加させる原因となる. 従って, スキュー制御配線では, ゼロスキュー配線では考慮しなかった各クロックピンのクロック時差に依存したトポロジー生成が重要な意味を持つ. 従って, 後者のボトムアップマッチングによる手法が適していると考えられる.

3 スキュー制御配線手法

前述のように, ゼロスキュー配線とスキュー制御配線は, クロック時差を 0 とするか否かの違いでしかない. このことから, ゼロスキュー配線手法をそのまま適用することによってスキュー制御配線手法が実現できると考えられるが, そのままでは迂回配線があまりにも大きく実用にならない. よってゼロスキュー配線のアルゴリズムに手を加えて, スキュー制御配線のアルゴリズムを考案するのが妥当である.

ここでは再帰的なマッチングをベースとするゼロスキュー配線手法である Clustering-Based Algorithm [2] に修正を加え, 後述するバッファ挿入条件を満たし, かつ総配線長最小化を目指すスキュー制御配線手法を提案する.

3.1 Algorithm [SC]

ここでスキュー制御配線アルゴリズム [SC] を提案する. このアルゴリズムは 2 つのアルゴリズム MakeTopology, Embedding から構成される.

アルゴリズム MakeTopology は, クロックピンを要素とする集合からボトムアップ方式で再帰的に, 要素数が 1 となるまでマッチングを行い, マージ操作を実行し, クロックネットワークのトポロジーを生成する. ここでクロックネットワークの内部点の位置の候補は点または線分となるので以降, 候補集合線分と呼ぶ. アルゴリズム Embedding は, 生成されたトポロジーに従い, クロックソースからクロックピンに向かう順序で, クロックネットワークの内部点の位置を候補集合線分上の 1 点から選択する. この時, 決定された親の点の位置から最短距離で配線できる点を選択する. このようにトップダウン方式でレイアウトの決定を行う. 以上の手順を合わせて DME(Deferred Merge Embedding) 法と呼ぶ.

図 2にクロックソース s_0, クロックピンの集合 $S = \{s_1, s_2, s_3, s_4\}$ からクロックネットワークを構成する例を示す. 図 2(a) のようにクロックピン s_1から距離 2, s_2から距離 3 である候補集合線分 v_5を生成し, s_1, s_2が v_5を経由して接続されることを確定する. この操作をマージと呼ぶ. 同様に s_3, s_4から v_6をマージする. その後, 候補集合線分 v_5, v_6からそれらの候補集合線分 v_7をマージする. ここで, 候補集合線分が 1 つになったので, トポロジーは生成された. 次に, 図 2(b) のようにトポロジーに従って s_0からレイアウトを決定する.

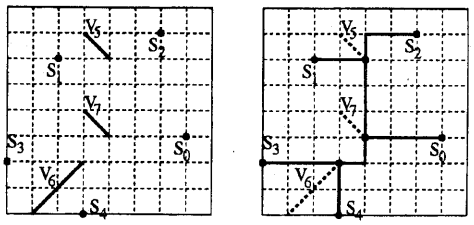

図 2: 候補集合線分の生成及びレイアウトの決定

3.1.1 MakeTopology

図 3にトポロジー生成の手順の疑似コードを示す. Kは再帰の各段階でのクロックピンまたは内部点の候補集合線分の集合である. アルゴリズムの開始時にはクロックピンの集合 Sとする.

関数 MINCOST は, 集合 Kを点集合 Vとする後述するマージングコストグラフ $G(V, E)$ を作成する.

関数 MID は $a \leq c$ の時に, 以下に示すように a, b, c

```
Algorithm MakeTopology

K = S;
while(|K| > 1){
    G(V,E) = MINCOST(K);
    number = MID(1, |K|/k, |K| − 1);
    for(l = 0; l <number; l++){
        e_ij = PICKUP(G(V,E));
        if(e_ij != NULL)
            MERGE(v_i, v_j);
        if(|E| < 1) break;
    }
}
```

図 3: アルゴリズム MakeTopology

のうち中間の値を返す.

$$MID(a,b,c) = \begin{cases} a & a \geq b \\ b & a < b < c \\ c & b \geq c \end{cases}$$

従って, 1つのマージングコストグラフからマッチングする候補集合線分の対の数 number は $1, |K|/k, |K|-1$ の中間値となる. ただし, kは定数で, $2 \leq k \leq 4$ 程度の値である.

関数 PICKUP はマージングコストグラフ $G(V,E)$ から重み最小の枝 e_{ij} を選択し, $G(V,E)$ から e_{ij} を削除する. さらに v_i, v_j が共に Kに含まれているならば, Kから v_i, v_j を削除し, e_{ij} を返す. そうでないならばNULL を返す. PICKUP によって選択された枝 e_{ij} の両端の 2 点 v_i, v_j は, 関数 MERGE によりマージ (3.2節) され, 候補集合線分 vとなり, vは Kに加えられる.

以上を $|K| = 1$ となるまで, Kの要素をマッチングする. ただし, number 回ごと, あるいはマージングコストグラフ $G(V,E)$ の枝数 $|E|$ が 0 となるごとに $G(V,E)$ の更新を行う.

3.1.2 Embedding

図 4にレイアウトの決定の手順の疑似コードを示す. 唯一の Kの要素である候補集合線分上の点で, クロックソース s_0から最短距離にある点を v_cとする. 関数 CONNECTROOT はクロックソース s_0と v_cを接続し, v_cを返す. また, 再帰的に呼び出される関数 CONNECT は, vの子 v_1, v_2に対して v_1の候補集合線分の中から vと最短距離にある点を新たに v_1とし, vと v_1を接続する. v_2についても同様の操作を行い, (v_1, v_2) を返す.

```
Algorithm Embedding

v_c = CONNECTROOT(s_0, K);
local_embedding(v_c);

procedure local_embedding(v)

if(v has children){
    (v_1, v_2) = CONNECT(v);
    local_embedding(v_1);
    local_embedding(v_2);
}
return;
```

図 4: アルゴリズム Embedding

3.2 候補集合線分の生成

マッチングによるトポロジーの生成は, マンハッタン平面上のマージングコスト最小の点対を選び, 両者から指定された遅延比を持つ点 (以降, ゼロスキューの用語を用い, 等遅延点と呼ぶ) の候補集合を求めることを再帰的に繰り返すことによって行われる. コストについては, 後述する. 配線長を抑えるためには, 2 点を結ぶ最短経路上に等遅延点の候補集合を置き, それぞれを最短距離で結ぶことが望ましい. 2 点 v_1, v_2からの等遅延点である候補集合線分 vを以下のように定める. v_1, v_2間の距離を l とし, 仮に vを最短経路上にとることができるとすると, v_1と vの配線長 l_1'は式 (1),(2) により以下のように表せる. なお, バッファが挿入される場合には, バッファ挿入後の $t(v_i), C(v_i)$ を用いる.

$$l_1' = \frac{t(v_2) - t(v_1) + rl\,(C(v_2) + cl/2)}{r\,(cl + C(v_1) + C(v_2))} \qquad (3)$$

$0 \leq l_1' \leq l$となれば, 候補集合線分 vは v_1と v_2を結ぶ最短経路上にある点, または傾きが 1 あるいは-1の線分 (マンハッタンアーク) となる [1].

それに対して,負荷容量や遅延時間のバランスが悪いために,$l_1' < 0$あるいは$l_1' > l$となる場合は等遅延を実現するために迂回配線を行う必要がある.$l_1' < 0$となる場合は等遅延の候補集合線分vをv_1上の点とし,vとv_1を結ぶ配線を迂回させ,等遅延を実現する.また同様に,$l_1' > l$となる場合には,vをv_2上にとり,vとv_2を結ぶ配線を迂回させる.以上により,l_1, l_2は以下のようになる.

$0 \leq l_1' \leq l$の場合

$$\begin{cases} l_1 &= l_1' \\ l_2 &= l - l_1' \end{cases} \tag{4}$$

$l_1' > l$の場合

$$\begin{cases} l_1 &= \frac{\sqrt{(rC(v_1))^2 + 2rc(t(v_2) - t(v_1))} - rC(v_1)}{rc} \\ l_2 &= 0 \end{cases} \tag{5}$$

$l_1' < 0$の場合

$$\begin{cases} l_1 &= 0 \\ l_2 &= \frac{\sqrt{(rC(v_2))^2 + 2rc(t(v_1) - t(v_2))} - rC(v_2)}{rc} \end{cases} \tag{6}$$

このように定められた候補集合線分からも同様の手順でそれらの候補集合線分を計算することができる.

3.3 中間バッファの挿入

クロックネットワークに中間バッファを挿入することは,クロックネットワークの遅延時間に,バッファによるゲート遅延を加えることになる.しかし,ゲートによって負荷容量が遮断されるため,クロックソースあるいは内部点における負荷容量を小さくできるので,クロックネットワーク全体の遅延時間を小さくすることが可能である.また負荷容量の減少は,製造誤差による配線幅の変化によって起こるクロック到達時刻の誤差を小さくし,信頼性を向上させることが可能である.また,クロック配線による消費電力を小さく抑えることも可能である.さらに,中間バッファを遅延素子と考えることは,マージの際の遅延調整において,迂回配線を短くすることも可能にする.また,バッファの遅延時間が大きくなり過ぎることは好ましくないので,トポロジー生成の際に,ある一定の負荷容量を超える候補集合線分に対応する内部点にはバッファを挿入することが必要である.

しかし,バッファの挿入数を大きくとりすぎると,バッファでの消費電力が大きくなり,クロックネットワーク全体の消費電力を増加させることになる.また,製造誤差によるバッファの遅延時間の誤差が大きくなり,クロックネットワークの信頼性を落とすことになりかねない.従って,適切なバッファ挿入数の制御が必要である.

挿入されるバッファのサイズはサイズの集合$W = \{w_1, w_2, \cdots, w_m\}$から選択する.サイズは等間隔で$w_{i+1} = w_i + w_{step}$であるものとする.これにより,$i < j$の時,遅延時間は$D_{buf}(w_i) > D_{buf}(w_j)$となる.

3.4 マージングコスト

内部点の候補集合線分v_1とv_2をマージすることにより増加するクロックネットワークの配線長は,式(4),(5),(6)によって計算されるl_1とl_2の和である.2点間をバッファを挿入することなくマージする場合には,マージングコストを$l_1 + l_2$とする.バッファを挿入することなくクロックネットワークを構成できたならば,マージングコストの総和は総配線長となる.

アルゴリズム[SC]では,2種類のバッファの挿入条件を採用する.1つは負荷容量の遮断のためであり,候補集合線分の負荷容量がC_{limit}以上となった場合に挿入を行う.もう1つは迂回配線長の減少のためであり,こちらは強制ではない.前者の場合はマージングコストを,$l_1 + l_2$とするが,後者の場合バッファの挿入数を制御するために,ペナルティPを導入する.迂回配線長の減少のために,ある候補集合線分にバッファを挿入したとする.バッファの負荷容量をCap,バッファに駆動させる容量の上限値をC_{limit}とし,Pを以下のように定義する.

$$P = \begin{cases} -\log\left(\frac{Cap}{C_{limit}/2}\right) & \text{if } Cap < C_{limit}/2 \\ 0 & \text{otherwise} \end{cases}$$

つまり,$C_{limit}/2$未満の容量を持つ候補集合線分には,バッファの負荷容量にのみ依存するペナルティを付加し,$C_{limit}/2$以上の容量を持つ候補集合線分には,ペナルティを付加しない.マージングコストは$l_1 + l_2 + \beta P$とする.ただしβは定数である.これによって,負荷容量の小さな候補集合線分には,バッファを挿入しにくくし,過剰なバッファの挿入を防ぐ.

いずれもトポロジー生成の際に候補集合線分にバッファを挿入する.ただし,1つの候補集合線分には高々1個のバッファしか挿入されないものとする.

以上により, マージングコスト M_{cost} は迂回配線長の減少のためにバッファの挿入を行う場合は

$$M_{cost} = l_1 + l_2 + \beta \cdot P$$

となり, その他の場合は以下のようになる.

$$M_{cost} = l_1 + l_2$$

2 点間のマージングコストはこれらのうち最小のものを選択する.

3.5 バッファのサイジング

候補集合線分 v_2 にバッファを挿入することを考える. 挿入するバッファの遅延時間 D_{buf} が以下に示す式を満たせば, 迂回を生じることなく配線できる.

$$D_{buf} \geq t(v_1) - t(v_2) - rl\left(C_{buf} + \frac{cl}{2}\right) \quad (7)$$

$$D_{buf} \leq t(v_1) - t(v_2) + rl\left(C(v_1) + \frac{cl}{2}\right) \quad (8)$$

等号が成立するのは, 式 (4),(5),(6) の再計算により $l_1 = 0$ または $l_2 = 0$ となる時, すなわち v が v_1 または v_2 と重なる時である.

アルゴリズム [SC] では, バッファのサイズを評価に加えていないが, 一般的にバッファのサイズは小さい方が, チップ面積, 消費電力などの点で有利である. 従って, サイズの観点からは, バッファの遅延時間は大きい方が良い. しかし, 式 (8) の右辺を D_{limit} とした時, バッファの遅延時間が D_{limit} を超えると迂回配線が必要となる. よって, 遅延時間が D_{limit} 以下である最小のサイズのバッファを挿入する. 図 5にこのサイジングを示す.

BufferSizing

size = w_1;
while (size < w_m){
    if ($D_{limit} \geq D_{buf}$(size)) **break**;
    size += w_{step};
}
return size;

図 5: バッファのサイジング

最大サイズのバッファを挿入しても遅延時間が D_{limit} を下回らないことがある. その場合でも最大サイズのバッファを挿入することにより迂回配線長を最小化する. また, バッファの遅延時間が式 (7) の右辺を下回ることも考えられるが, 最小サイズのバッファを挿入することによって, 迂回配線長を最小化する. v_1 にバッファを挿入する場合も同様にサイジングを行う.

v_1, v_2 の双方にバッファを挿入する場合は, バッファサイズは, 互いの遅延時間に依存するので, 前述の方法をそのまま適用することはできない. v_1 に挿入するバッファの負荷容量を C_{buf_1}, 遅延時間を D_{buf_1} とする時, v と v_1 を迂回することなく配線するための v_2 に挿入するバッファの遅延時間の上限は以下のように表される.

$$D_{limit} = t(v_1) - t(v_2) + rl\left(C_{buf_1} + \frac{cl}{2}\right) + D_{buf_1}$$

このように, D_{limit} は v_1 に挿入するバッファに依存する.

ここでは, v_1 のバッファのサイズ size1 を最大サイズから順に小さいサイズに変更する. それぞれのサイズにおいて v_2 のバッファのサイズ size2 を最小サイズから順に大きいサイズに変更する. v_2 に挿入したバッファの遅延時間が D_{limit} 以下となった時にサイジングを終了する. これは,v から v_1 への配線長 l_1 を最小化し, v での遅延時間をできるだけ小さくするものである. このサイジングの様子を図 6に示す.

BufferSizing2

size1 = w_m;
while (size1 > w_1){
    size2 = w_1;
    while (size2 < w_m){
        if (D_{limit}(size1) ≥ D_{buf}(size2))
        return size1,size2;
        size2 += w_{step};
    }
    size1 -= w_{step};
}

図 6: バッファのサイジング 2

3.6 バッファの挿入条件

トポロジー生成のマージの際に C_{limit} を超える容量を持つ候補集合線分には，強制的にバッファの挿入を行う．また，迂回配線長を減少させることができる時にもバッファの挿入を考慮する．マージする候補集合線分 v_1, v_2 の負荷容量 $C(v_1), C(v_2)$ によって場合分けを行うが，図7に示す手順に沿ってバッファの挿入及び，マージングコストの評価を行う．

Step 1 負荷容量を評価し，必要であればバッファの挿入 (強制) 及びサイジングを行う．

Step 2 式 (4),(5),(6) より，l_1' 及び l_1, l_2 を算出する．

Step 3 $0 \leq l_1' \leq l$ (迂回配線の必要がない) ならば，$M_{cost} = l_1 + l_2$ として終了．

Step 4 バッファの挿入，サイジングを行い，l_1' を再計算，式 (4),(5),(6) より算出した l_1, l_2 を l_1'', l_2'' とする．

Step 5 $l_1 + l_2, l_1'' + l_2'' + \beta P$ を比較する．$l_1 + l_2$ の方が小さければバッファの挿入を中止し，$M_{cost} = l_1 + l_2$ として終了，そうでなければバッファを挿入し，$M_{cost} = l_1'' + l_2'' + \beta P$ として終了．

図 7: バッファの挿入とコスト

Step 1 における負荷容量の評価によって，次の 3 つの場合を考える．

(a) いずれも C_{limit} 未満となる場合

強制的なバッファの挿入がないので，Step 1 をスキップする．また，Step 3 において，$l_1' < 0$ となる時には遅延素子として v_2 にバッファを挿入することにより，迂回配線の減少を試みる．Step 5 においてコストの評価を行い，バッファの挿入の可否を決定する．$l_1' > l$ となる時も同様に v_1 へのバッファの挿入を試みる．

(b) 一方が C_{limit} 以上となる場合

T_{v_2} の負荷容量が C_{limit} 以上であるとする．Step 1 において，v_2 に強制的にバッファを挿入し，v_2 のサイズを決定する．Step 3 において，$l_1' < 0$ となる場合には，v_1 へのバッファを挿入を試みる．この時，v_2 に挿入されているバッファのサイズは固定されていると考え，v_1 のサイジングを行い，コストの評価を行い，v_1 へのバッファの挿入の可否を決定する．$l_1' > l$ となる場合には，すでに v_2 にバッファが挿入されているので，これ以上のバッファの挿入は行わずに，$M_{cost} = l_1 + l_2$ とする．また，T_{v_1} の負荷容量が C_{limit} 以上となった場合も同様である．

(c) 両方が C_{limit} 以上となる場合

v_1, v_2 双方にバッファを挿入することになる．Step 1 において図6に示すサイジングを行い，コストを計算する．Step 3 において，迂回が生じたとしても，これ以上バッファを挿入することなく終了する．

3.7 マージングコストグラフ

トポロジーの生成の際は，複数の候補集合線分の対を同時にマージする．コスト最小の候補集合線分の対を選択するためにマージングコストグラフ $G(V, E)$ を定義する．これは，各点がただ 1 本の出力枝を持つ重み付き有向グラフである．点 $v_i \in V$ は，トポロジー生成における再帰の各段階でのクロックピン，あるいは候補集合線分である．従って，初期段階では，すべてがクロックピンとなる．点 v_i と V に含まれる v_i 以外のすべての点との間でマージングコストを計算し，v_i と v_j をマージした時のコストがコストが最小となった時，点 v_i から v_j への枝 $e_{ij} \in E$ が結ばれ，その枝の重みは v_i と v_j をマージした時のコストである．

4 実験結果

アルゴリズム [SC] を C++言語により実装し，実験を行った．実験に用いたデータは表2に示すクロックピン数，チップの幅及び高さを満たすもので，クロックピンの位置は乱数によって生成した．また，各クロックピンの負荷容量は 30~80fF の値を，クロック時差は 0, 0.5, 1, 1.5, 2ns のいずれかを乱数によって与えた．各定数は $r = 60\ \mathrm{m\Omega}, c = 0.04\mathrm{fF}, C_{limit} = 2\mathrm{pF}, \beta = 1000$ とした．

表 1はゼロスキュー配線手法であるアルゴリズム [CL] と，スキュー制御配線手法であるアルゴリズム [SC] の比較である．[CL] と [CL+Skew] は同じアルゴリズムを適用しているが，[CL] にはすべてのクロッ

data	Algorithm [CL]		Algorithm [CL+Skew]			Algorithm [SC]			
	wire length	delay	wire length	ratio	delay	wire length	ratio	delay	buffers
r1	147,392	1.82	3,947,102	26.78	13.02	266,514	1.81	4.78	62
r2	294,797	6.07	9,034,421	30.65	51.08	543,183	1.84	6.35	129
r3	366,265	5.85	13,139,596	35.87	83.45	679,889	1.86	6.93	177
r4	727,893	21.75	28,361,307	38.96	198.30	1,310,030	1.80	9.17	350
r5	1,063,507	38.52	45,920,134	43.18	327.47	1,886,817	1.77	5.16	515

表 1: ゼロスキュー配線とスキュー制御配線の総配線長及び遅延時間の比較

data	# of pins	width[μm]	height[μm]
r1	267	6998	7000
r2	598	9401	9313
r3	862	9700	9850
r4	1903	12697	12698
r5	3101	14292	14522

表 2: 実験データ

クピンのクロック時差を 0 としたデータを与えてある. 表中の wire length はクロックネットワークの総配線長で, 単位はμm, delay は [CL] ではクロックソースからクロックピンまでの遅延時間, [SC] では, クロックソースからクロック時差 0 が割り当てられたクロックピンまでの遅延時間であり, 単位は ns である. [CL+Skew] の結果における ratio は [CL+Skew] の総配線長と, [CL] の総配線長との比であり, 同様に [SC] の結果における ratio は [SC] の総配線長と, [CL] の総配線長との比である. また, buffers は挿入されたバッファの数である.

[CL+Skew] の結果から, 0 でないクロック時差の与えられたデータにゼロスキュー配線を適用することは総配線長を大きく増加させることがわかる. しかし, 同じデータに [SC] を適用することにより, 総配線長をクロック時差が 0 の場合の [CL] の総配線長の 1.8 倍程度に抑えることが可能である.

また, 中間バッファの挿入により, クロックネットワークの遅延時間を r1, r2, r3 ではやや増加させるものの r4, r5 では劇的に減少させている. これは, クロックネットワークの内部点での負荷容量の大幅な減少によるものであり, 挿入したバッファに誤差がないとすれば, クロックネットワークの製造誤差等による遅延時間の誤差が生じる可能性を大幅に小さくしていることを意味する.

5 結論

スキュー制御配線アルゴリズム [SC] を提案し, これにより従来のゼロスキュー配線手法の 1.8 倍程度の総配線長で, 0~2ns のクロック時差の存在するクロックネットワークを構成した.

謝辞

本研究を進めるにあたり梶谷研究室諸賢の助言が大きく貢献したことを付記し, 深謝する. なお, 本研究は CAD21 プロジェクトの一部である.

参考文献

[1] T-H. Chao, Y-C. Hsu, and J-M. Ho. Zero skew clock net routing. *Proc. of 29th Design Automation Conf.*, pp. 518–523, 1992.

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

[3] S. Pullela, N. Menezes, J. Omar, and L. T. Pollage. Skew and delay optimization for reliable buffered clock trees. *Proc. IEEE Int. Conf. Computer-Aided Design*, pp. 556–562, 1993.

[4] A. Takahashi and Y. Kajitani. Clock period minimization by clock skew control. *IECE Technical Report VDL95-42*, pp. 85–92, 1995.