設 計 自 動 化 76－4
（１９９５．７．２０）

既配線の押し退け可能性を考慮したゼネラルエリア配線手法

林 中也　奥田 亮輔　中尾 博臣　寺井 正幸　佐藤 興二

三菱電機　システムＬＳＩ開発研究所
〒664 伊丹市瑞原 4-1

本論文では新しいゼネラルエリア配線手法を提案する。ゼネラルエリア配線では通常、１ネットずつ配線する手法が用いられるが、各ネットを配線する際、既に引かれている配線が邪魔になり、配線できないことが起こる。この時、既配線と短絡を起こす経路をまず見つけ、この既配線を移動させて短絡をなくす手法がしばしば用いられる。短絡は touch（接触）と cross（交差）に分類される。新手法の特長は、“押し退け可能性の高い touch（cross）”と“可能性の低い touch（cross）”という概念を新たに導入し、可能性の低い touch（cross）を起こす経路より可能性の高い touch（cross）を起こす経路を見つけるようにしたことである。ゼネラルエリア配線のベンチマークとして有名な Burstein's difficult switch box 問題に適用した実験で、本手法は学会発表されている他の手法以上の高配線率を達成した。

A General Area Router Considering the Possibility of Shoving Aside

Chuya Hayashi　Ryosuke Okuda　Hiroomi Nakao　Masayuki Terai　Koji Sato

System LSI Laboratory, Mitsubishi Electric Corporation
4-1, Mizuhara, Itami, Hyogo, 664 Japan

This paper presents a new general area router. A typical approach to the general area routing problem is to route one net at a time. The main problem is that nets routed later may be blocked by nets routed earlier. In this context most current routers find a path, allowing conflicts (short circuits) with previously routed wires, and move (e.g., shove, rearrange) the wires to eliminate the conflicts. The conflicts can be classified into two types, *touches* and *crosses*. We introduce a new conception of *touch (cross) with high possibility of shoving* the previously routed wire and *touch (cross) with low possibility of shoving*. The novelty of our router is that a path causing touches (crosses) with high possibility of shoving is found prior to a path causing touches (crosses) with low possibility of shoving. Experimental results on Burstein's difficult switch box, which is a famous benchmark for the general area routing, show that the router obtained higher completion than other published routers.

1　はじめに

本手法が扱うゼネラルエリア配線問題は、グリッドベースであり、形状、層の数に制限のない領域内で、任意の場所に置かれた複数の端子間を結ぶことである。ゼネラルエリア配線問題では通常、1ネットずつ配線する手法が用いられるが、各ネットを配線する際、既に引かれている配線が邪魔になって配線できないことが起こる。

この時、次の3つの手法がよく用いられる。（1）邪魔になっている既配線を引き剥がして、この既配線は全く引かれていないものとして繋ぎたいネットの経路を探索して配線し、引き剥がした配線を配線し直す手法（[5], [6], [9]）、（2）邪魔になっている既配線を局所的に移動させ、配線するためのスペースを空けてから配線経路を探索する手法（[5]）、（3）既配線と短絡を起こす経路をまず見つけ、（短絡を起こしてでもその経路で結ぶ手法も含め）後に経路上の既配線を移動させて短絡をなくす手法（[1], [2], [3], [4], [7], [8]）。

手法（1）では邪魔な既配線を引き剥がした後、この既配線の再配線を全く考慮せず繋ぎたいネットを配線するので、この時に引いた配線が邪魔で、引き剥がした配線を配線し直すことができない可能性が高い。また、手法（2）では既配線を局所的にしか移動させないので、しばしば配線のためのスペースを空けることができない。この手法は、第1の手段として用いられるもので、失敗した場合には、手法（1）が用いられる。手法（3）は邪魔な既配線を引き剥がす前に経路を見つけるので、既配線の存在を考慮して（例えば、既配線との短絡の数が最小となる様に）経路を見つけることができ、既配線の移動により短絡がなくなり配線を完成できる可能性が高い。

手法（3）は大きく2つに分けられる。1つは、配線経路を見つけるときに既配線との短絡を "touch" と "cross" に区別するもの（[1], [2]）、他の1つは区別しないものである（[3], [4], [7], [8]）。

図1は1層の配線問題の例で、細線で区切られた小さなマスが1つの配線グリッドを表す。今、ネットB, Cの既配線が存在し、さらに、ネットAを配線したいとする。破線1の経路はネットCの配線とcrossを起こす経路、2の経路はネットBの配線とtouchを起こす経路である。2の経路を見つければ、ネットBの配線を右側に押し退けるだけで、ネットAを見つけた経路で配線することができる。一般に、touchを起こす経路を見つけた場合、邪魔になっている既配線を押し退けるだけで配線できる可能性が高いが、crossを起こす経路を見つけた場合には、その経路で配線しようとすると、既配線を大幅に移動させなければならず、移動が成功する可能性が低い。また、既配線を他の層に移動させることも考えられるが、層の数が限られている場合にはその可能性は低い。よって、touchとcrossを区別し、crossよりtouch

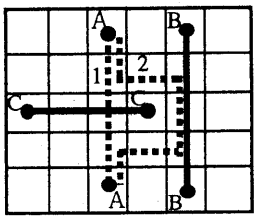

図1　touchとcross

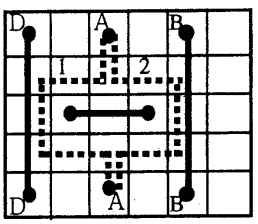

図2　押し退け不可のtouchと押し退け可のtouch

を起こす経路を優先的に見つける方が正しい経路を見つけられる可能性が高いと言える。

一般にtouchを起こす経路は数多くある。例えば、図2の問題の場合、touchとcrossを区別しtouchを優先しただけでは（一度の経路探索では）1の経路か2の経路のどちらを見つけるかは分らない。2の経路を見つければ、ネットBの配線を右側に押し退けるだけで、ネットAを見つけた経路で配線できるが、1の経路を見つければ、（この経路を変更しない限り）他の配線をどの様に変更してもネットAとの短絡を無くすことはできない。即ち、押し退けが成功する確率を最大にする経路を選ぶことが重要である（これは[2]でも指摘されている）。しかし、著者の知る限り、過去にその様な工夫をした手法はない。そこで、著者らは“押し退け可のtouch”（押し退けられ

る可能性の高い既配線との touch）、“押し退け不可の touch”（押し退けられる可能性の低い既配線との touch）という概念を導入した。図２の１の経路は押し退け不可の touch、２の経路は押し退け可の touch を起こす経路である。押し退け不可の touch より押し退け可の touch を起こす経路を優先的に見つけることにより、従来何度も経路探索を繰り返し、（例えば、一度見つけた経路を見つけにくくして）数多くの経路を順次見つけることにより初めて正しい経路を見つけることができた問題に対して、経路探索の回数を減らすことができ、計算時間を短縮することができる。また、実用的な時間では正しい経路を見つけられず解けなかった問題も解くことができると期待できる。

図３に示す問題では、touch を起こす経路は数多く存在するが、正しい経路は破線１の経路のみである。一般に、配線すべきネットの数が多いほど、また配線が混雑するほど、touch を起こす経路の数は飛躍的に多くなる。従って、押し退け可の touch を起こす経路を優先的に見つけることによる、計算時間短縮の効果は大きくなると言える。

本論文では、cross より touch を起こす経路を優先的に見つける配線手法に、上で述べた押し退け不可の touch を起こす経路より押し退け可の touch を起こす経路を優先して見つける機能を付加した手法を述べる。この機能を付加することにより、上で述べたように高速配線、実用的な時間内での配線における高配線率が期待できる。４章では、有名な配線問題である、Burstein's difficult switch box でその効果を実証する。

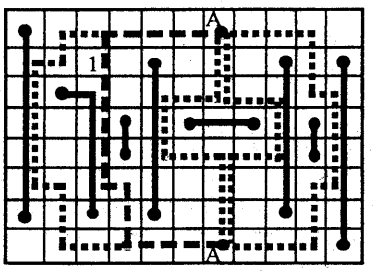

図３　touch を持つ経路が数多くある例

2　配線問題の定義

本論文で扱う配線領域は、グリッドベースであることだけが仮定され、配線層の数、配線領域の形状、端子の形状、位置に制限のない、ゼネラルエリア（general area）である。また、配線領域内に任意の形の障害物があってもよい。ここで、解くべき配線問題は、端子の集合が与えられ各端子にネット名が対応付けられている時、同一ネット名に対応付けられているすべての端子を（短絡なしに）繋ぐことである。

3　配線アルゴリズム

3.1 で基本となるアルゴリズムを述べ、3.2 以降でアルゴリズムの詳細を述べる。

3.1 基本アルゴリズム

既配線と cross を起こす経路は３次元的にみれば、既配線の上を越えるか下に潜るかで２通りに分かれるにせよ、touch を起こす経路とみることができる。そこで、今まで touch と呼んでいたものを xy-touch、cross を z-touch、これらを共に touch と呼ぶことにする。xy-touch と z-touch を区別する理由は、１章でも述べたように配線領域内で x 方向、y 方向のグリッド数に比べ、z 方向のグリッド数が非常に少ない場合に有効であると考えるためである。

本論文で提案する手法は、１ピンペア（同一ネット名を持つ端子のペア）ずつ配線する逐次配線手法であり、各ピンペアの配線を終えたときには、短絡は存在しないようにする。

結ぶピンペアの順序は、近いピンペアから結ぶように決められる。ここで、近いとはピンペアを結ぶための最短経路長が小さいという意味である。ただし、本論文で言う経路長とは、経路の長さそのものではなく、経路の長さに配線層別、方向（縦横）別、またはビアの種類別の重みを掛けた値である。また、最短経路とは配線領域内の障害物を避ける全ての経路のうち、最短のものを表す。つまり、結ばねばならない全ピンペアについて、そのピンペアを結ぶための最短経路長を計算し、小さいものから結んでいく。最短経路長を計算するためには迷路法を用いる。ただし、配線領域が特殊な場合（例えば、スイッチボックス）には、ピンペアの座標から簡単に計算することができ、迷路法を用いる必要はない。

遠いピンペアから順にピンペアをスタックに入れる（近いピンペアから順に取り出すことになる）。ここで、３端子以上のネットについては全てのピンペアをスタックに入れる必要はない。あ

るピンペアに対して、複数のより近いピンペアを結ぶことによってそのピンペアが結ばれるならば、そのピンペアはスタックに入れない。例えば、ピンペア (p1,p2) と (p2,p3) をスタックに入れる場合、(p1,p3) はスタックに入れない。

1ピンペアを結ぶためのアルゴリズムを、簡単のため1層配線の図を用いて説明する（図4）。ここで、

（経路のコスト）＝（経路長）＋Ct×（押しのけ可の xy-touch の回数）＋Cut×（押しのけ不可の xy-touch の回数）＋Cc×（押しのけ可の z-touch の回数）＋Cuc×（押しのけ不可の z-touch の回数）

と定義しておく（回数は touch を起こすグリッドの数）。Ct, Cut, Cc, Cuc は正の定数で、z-touch を持つ経路より xy-touch を持つ経路を見つけるように　Ct＜Cc, Cut＜Cuc を満たすようにする。また、押し退け不可の touch より押し退け可の touch を持つ経路を見つけるように Ct＜Cut, Cc＜Cuc を満たすように設定する。

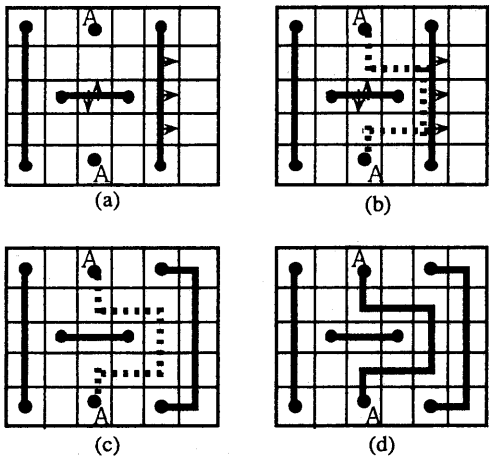

図4　本手法による配線の例

まず最初に、押し退け可能マーク設定処理を行う（図4a、設定処理の詳細は 3.3 節）。ここでは、既配線の存在する全グリッドの内、既配線を押し退け可能であるグリッドに、押し退け可能な方向に矢印を付ける。次に、迷路法を用いてコスト最小の経路を見つける（図4b）。上で設定した押し退け可能マーク（矢印）を基に、押し退け不可の touch より押し退け可の touch を持つ経路を優先的に見つけることができる。次に、経路上の既配線を押し退ける（図4c）。ここでは、経路上の、既配線の存在するグリッドについて1グリッドずつ空けていく（[2]）。押し退けることができれば、配線は成功（図4d）。

経路上の既配線を押し退けることに失敗した場合には、押し退けることのできなかった既配線を引き剥がしてでも、経路上のグリッドを空ける。そして、この経路で今引こうとしていた配線を引いた後、引き剥がした配線を引き直す。

全ピンペアを結ぶためには、前記の方法で全ピンペアに順序をつけ、1ピンペアずつ結ぶ。以下に、全ピンペアを結ぶ手順を示す。

[配線アルゴリズム]

step 1: 結線すべきピンペアを遠いピンペアから順にスタックに登録。

step 2: 押し退け可能マーク設定処理（詳細は 3.3 節）。

step 3: スタックから最後に登録されたピンペアを取り出し、そのピンペアに対しコスト最小の経路を見つける。

step 4: 見つけた経路上に既配線がなければ step 8 へ。

step 5: 見つけた経路上の既配線の押し退けを試みる。

step 6: 経路上の既配線をすべて押し退けられれば step 8 へ。

step 7: 経路上の既配線を引き剥す。これにより新たに結線する必要が生じたピンペアをスタックに登録。

step 8: スタックに登録されているピンペアがなければ、成功で終了。

step 9: step 2 へ。ただし、step 2 ～ 9 のループの回数が上限を越えれば、失敗で終了。

3.2 押し退け可能の定義

以後、東、北、上とはそれぞれ配線領域内の x 軸方向（右）、y 軸方向、z 軸方向（上の層への方向）を表し、西、南、下はそれぞれ東、北、上と逆の方向を表す。

g を既配線の存在するグリッド、d を東西南北上下の一つの方向とする。以下の5条件を全て満たすとき、g 上の既配線を d 方向に押し退け可能であると言う。

1）グリッド g の d 方向に隣接するグリッドが

存在する（このグリッドをｈとする）。

2）グリッドｇに端子が存在しない。

3）ｇからｄと反対の方向に配線が出ていない。

4）ｈに、配線が存在しないかｇ上の配線と同じネットに属する配線が存在するかまたは、ｈ上の既配線をｄの方向に押し退け可能である。

5）ｄと垂直な方向（例えば、ｄが東なら南北上下）でｇから配線が出ている全ての方向（例えば、ｇから南北に配線が出ているときは南と北）について、その方向へｈから１グリッド離れたグリッドに、配線が存在しないかｇ上の配線と同じネットに属する配線が存在するかまたは、そのグリッド上の既配線をｄの方向に押し退け可能である。

上記の４）、５）の条件から解るように、押し退け可能であるとは、再帰的に定義される。

例えば、図５に示すように配線が引かれているとする。東向きの矢印は、その矢印の置かれているグリッド上の既配線を東方向に押し退け可能であることを表す。この例でｇ上の既配線を東向きに押し退け可能であるかを考える。上記の１）～４）の条件は明らかに満たされている。また、５）の条件もグリッドh2に配線がなくグリッド

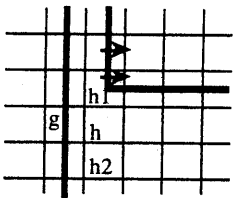

図５　押し退け可能の定義

h1上の既配線は東向きに押し退け可能であるので、満たされている。よって、ｇ上の既配線を東向きに押し退け可能である。

3.3 押し退け可能マーク設定処理

押し退け可能マーク設定処理では、既配線の存在するグリッドの内、そのグリッド上の既配線を押し退け可能であるグリッドについて、押し退け可能な方向に矢印をつける。この矢印を押し退け可能マーク（以後、マーク）と呼ぶ。ここでは、マーク設定処理の方法を述べる。マークを最初に設定する際は、既配線が全く存在しない状態であるので、全グリッドについてマークがつけられでいない状態にする。以下では、マークの２回目以降の設定方法を述べる。

マークを設定する時、前回設定したときから変化した領域を含む一部の領域だけについて、マークを設定し直せばよい。この設定の方法を以下に述べる。ここでは、東向きの矢印をつける場合についてのみ説明するが、６方向について同様に行う。また、各グリッドの位置を３次元の座標で表わす。ただし、隣り合うグリッドの間隔を１として、座標値はすべて整数となるようにする。

前回設定したときから変化したグリッドの集合をＥとする。ここで、変化したとは次のいづれかを意味する。

・配線が存在しなかったグリッドから配線が存在するグリッドに変わった、
・配線が存在したグリッドから存在しないグリッドに変わった、
・存在する既配線のネット名が変わった、
・存在する既配線のネット名は同じで出ている配線の方向が変わった。

さらに、次のように定義する。

E(x)＝（Ｅ内のグリッドのうちｘ座標がｘであるグリッドの集合）。xmax ＝（E(x) が空集合でない最大のｘ）。xmin ＝（E(x) が空集合でない最小のｘ）。

E(xmax) 内のグリッドから調べ、順に西方向へグリッドを調べていく。Ｅ内のグリッドはすべて調べる。また、Ｅ内のグリッドまたは押し退け可能性が前回マークを設定したときと異なるグリッドの西に位置するグリッドとその南北上下のグリッドについても調べる。Ｅ内のグリッド全てについて設定し直し、かつ、押し退け可能性が前回設定した時と異なるグリッドが見つからなければ、終了する。詳しくは次のようになる。Ｃは調べようとするグリッドの集合を表わす。

［押し退け可能マーク設定手順（２回目以降）］

for(x = xmax, C = ϕ ; C≠ ϕ or x≧xmin; x = x-1){

E(x) 内の既配線の存在しない各グリッドｇについてマークを消す。

E(x)∪C 内の既配線の存在する各グリッドｇについて、3.2 １）～５）の条件を全て満たす時のみ東向きの矢印をつける（再帰的な処理は不要であることに注意）。

E(x)∪C 内のグリッドの内、E(x) に含まれるか、押し退け可能性が前回設定した時と異なっ

ているグリッドの集合をC'とする。

C = { (x,y,z) | (x+1,y,z)∈C'}∪
{ (x,y,z) | (x+1,y+1,z)∈C' or (x+1,y-1,z)∈C'}∪
{ (x,y,z) | (x+1,y,z+1)∈C' or (x+1,y,z-1)∈C'}
}

3.4 1グリッドの構造と波の伝搬

本手法では、2つのピンを結ぶコスト最小の経路を見つけるために迷路法を用いる。通常の迷路法では、1グリッドを1セルで構成し、経路探索の際各セルは次の2つの情報を持つ。(1) 迷路法の始点からそのセルまでの経路のコスト。(2) 迷路法における波がそのセルに到達する1つ前のセルがどのセルか。

しかし、この様な迷路法を本手法で用いると、始点sと終点tを結ぶコスト最小の経路（Pと呼ぶ）を見つけることが出来ない場合がある。なぜならば、「sから、P上の任意のグリッドgまでの、Pの部分経路はsとgを結ぶコスト最小の経路である」という通常の迷路法の前提条件が満たされないからである。満たされないのは、g上に既配線が存在する場合で、Pがgで既配線とtouchを起こす方向とsとgを結ぶコスト最小の経路がgで既配線とtouchを起こす方向が異なる場合である。

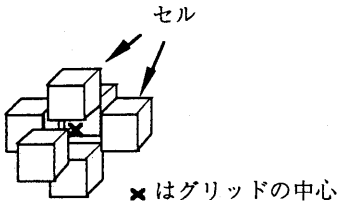

図6　1グリッドを構成する6セル

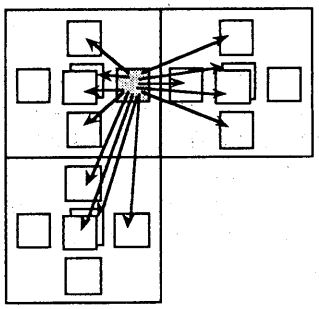

図7　セル間の波の伝搬の例（既配線が存在しない場合）

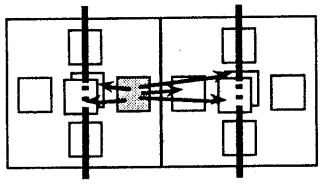

図8　セル間の波の伝搬の例（既配線の存在する場合）

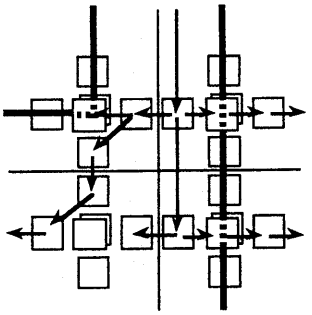

図9　波の伝搬の例

ある。

コスト最小の経路を見つけるため本手法では、迷路法における波が、既配線の存在するグリッドに到達したとき、その波で表される経路が、どの方向から既配線にtouchを起こしたかによって、異なるセルに到達するようにする。

この様にすると、波がセルに到達した時点で、（既配線に沿って何グリッドにもtouchを起こしながら進む場合でも）xy-touchかz-touchか、押し退け可のtouchか押し退け不可のtouchかが分かり、始点からそのセルまでの経路のコストを計算することも出来る。

図6に示すように1グリッドを6セルで構成する。各セルはグリッドの中心に対し、東西南北上下の位置に置かれる。これらのセルを東ーセル、西ーセル、…、下ーセルと呼ぶことにする。迷路法で経路を探索する際、波はこのセルを通過して伝搬していくが、あるグリッドgに属するdーセル（dは東西南北上下のいずれか）から移ることができるセルは図7に示すように以下のいづれかのセルである。ここでdと逆の方向をrとする。

1）グリッドgに属するfーセル。ただし、f≠d，r。
2）グリッドgのdの方向に隣接するグリッドに属するfーセル。ただし、f≠d。
3）グリッドgのhの方向（h≠d，r）に隣

接するグリッドに属するｆ－セル。ただし、
ｆ≠ｈ，ｒ。

ただし、図８に示すように既配線が通過するセルに移ることはできない。配線は常にグリッドの中心同士を結ぶように引かれる。セルは迷路法で経路を探索する際にのみ用いられ、実際に配線を引く際には用いられない。経路探索の後、セル同士を結ぶように配線を引くのではなく、それらのセルの属するグリッドの中心同士を結ぶように引く。

迷路法で波が既配線の存在するグリッドに属するｄ－セル（ｄは東西南北上下のいずれか）に到達すると、その波で表現される経路はそのグリッドで既配線に touch を起こしたとみなされる。ここで、そのグリッドにｄと逆の方向に矢印が付いていれば押し退け可の touch、付いていなければ押し退け不可の touch を起こしたとみなす。また、ｄが東西南北のいづれかなら xy-touch、上下のいづれかなら z-touch を起こしたとみなす。そして、押し退け可の touch か押し退け不可の touch か、xy-touch か z-touch かによって経路のコストを計算する。

4　実験結果

本手法を EWS（Sparc Station 2）上で実行した結果を以下に示す。

当社ゲートアレイ用マクロセル（300 Tr 以下の回路）の内から配線の難しいと思われるセル７セルを選び、押し退け可能マーク設定処理を行った場合と行わない場合（Ct＝Cut, Cc＝Cuc と設定）で、配線に要する CPU 時間を測った（表１）。配線は２層で行った。表から分かるように、押し退け可能性を考慮することによって、（１セル例外はあるが）高速性、高配線率を得ることができた。

表１　押し退け可能性を考慮する効果

セル	ネット数	ピン数	area	配線に要する CPU 時間（分） ①押し退け可能マーク設定処理あり	②押し退け可能マーク設定処理なし	比率 ②／①
A	30	138	49x12	5.5	fail (200)	---
B	20	86	32x12	5.3	fail (120)	---
C	20	90	33x12	fail (144)	14.2	---
D	44	214	75x12	39.4	59.2	1.50
E	44	224	79x12	14.6	40.0	2.75
F	72	386	136x12	7.8	43.7	5.58
G	72	370	128x12	14.0	33.7	2.41

fail は配線失敗、( ) 内は配線に失敗するまでの時間を表す

表２　Burstein's difficult switch box への本手法及び他の手法の適用結果

	area	本手法	Touch and Cross[1]	Codar [3]	Packer [4]	Mighty [5]
burst	22×15	○	○	○	○	○
comp-burst	22×14	○	?	○	○	fail
	21×14	○	?	○	?	fail
	20×14	○	?	?	?	?

○は配線成功、fail は配線失敗、？は不明を示す

配線結果の例を図１０に示す。ただし、この図は本手法で通常の配線をした後、１ネットずつ本手法で配線し直し、迂回配線の削減を行った後の結果である。

ゼネラルエリア配線手法の性能を評価するためのベンチマークとして有名な配線問題である、Burstein's difficult switch box（以後、burst と言う）とcomp- burst に適用した結果を表２に示す。comp-burst とは burst を圧縮して、配線領域を小さくした問題である。本手法では、表に示した４つの問題全てを解くことができた。comp-burst

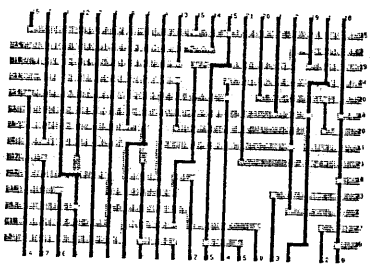

図１１　comp-burst (20x14) の配線結果

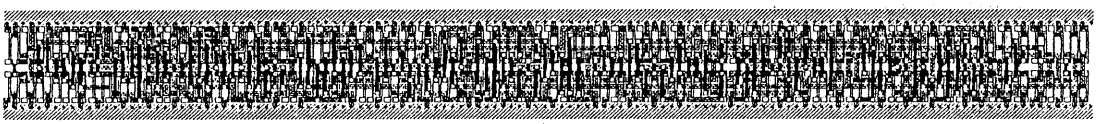

図１０　配線結果の例

(20x14) の配線結果を図１１に示す。

さらに、burst, comp-burst に本手法を押し退け可能マーク設定処理を行わないで適用し、設定処理を行った場合と配線に要する CPU 時間を比較した。結果を表３に示す。comp-burst (21x14), (20x14) については押し退け可能マーク設定処理を行わない場合には、(25分以内には) 配線でき

表３ Burstein's difficult switch box に本手法を適用した場合の、押し退け可能性を考慮する効果

	area	配線に要する CPU 時間 (秒) ( ) 内は引き剥しの回数	
		押し退け可能マーク設定処理あり	押し退け可能マーク設定処理なし
burst	22×15	10.3 (10)	9.1 (7)
comp-burst	22×14	11.1 (18)	10.4 (18)
	21×14	9.7 (18)	fail
	20×14	6.7 (3)	fail

なかったが、行った場合には配線することができた。burst と comp-burst (22x14) に適用した場合には、押し退け可能マークを設定した場合の方がわずかながら配線に多くの時間を要しているが、これはこれらの問題では混雑度が低いため引き剥がし再配線の回数が減らず押し退け可能マーク設定処理の分多くの時間を要するためである。

押し退け可能マーク設定処理を行わない場合には、配線領域が小さいほど引き剥がし再配線の回数が増え配線することができなくなるが、押し退け可能マーク設定処理を行った場合は引き剥がし再配線の回数があまり変わらないことがわかる。

5 まとめ

既配線の押し退け可能性を調べ、その結果を基に配線経路を見つける手法を開発した。実験結果から、本手法によって高速に配線できることが分かった。また、今まで実用的な時間では解けなかった問題を解くことも可能であると期待できる。

[参考文献]

[1] K. Kawamura et al., "Touch and Cross Router," Proc. ICCAD, 1990, pp. 56-59.

[2] H. Bollinger, "A Mature DA System for PC Layout," Proceedings International Printed Circuits Conference, 1979, pp. 85-99.

[3] P. S. Tzeng et al., "Codar: A Congestion-Directed General Area Router," Proc. ICCAD, 1988, pp. 30-33.

[4] S. H. Gerez et al., "Switch box routing by stepwise reshaping. IEEE trans. on CAD, Vol. 8, No. 12, 1989, pp. 1350-1361.

[5] H. Shin et al., "A Detailed Router Based on Incremental Routing Modifications : Mighty," IEEE trans. on CAD, Vol. CAD-6, No. 6, 1987, pp. 942-955.

[6] M. Raith et al., "A New Hypergraph Based Rip-up and Reroute Strategy," Proc. 28th DAC, 1991, pp. 54-59.

[7] F. Rubin, "An iterative technique for printed wire routing," Proc. 11th Design Automat. Workshop, 1974, pp. 308-313.

[8] E. Rosenberg, "A New Iterative Supply/Demand Router with Rip-Up Capability for Printed Circuits Boards," Proc. 24th DAC, 1987, pp. 721-726.

[9] K. Kawamura et al., "Hierarchical Dynamic Router," Proc. 23rd DAC, 1986, pp. 803-809.