設計自動化 34－9
（1986. 10. 21）

# ストレッチ･シンボル対応型レイアウトコンパクタ

山内 貴行　谷 貞宏　神戸 尚志　西岡 郁夫
シャープ株式会社 コンピュータシステム研究所

LSIの大規模化･高密度化に伴い、マスクパターン設計の効率化が重要な問題となっている。コンパクタの使用は、設計期間の短縮と設計ミス防止という点から非常に有効であるが、コンパクション可能なシンボリックレイアウトには制限があり、設計者は一般のマスクパターン設計とは異なったレイアウトを余儀なくされ、コンパクタの長所を十分に生かせない場合が多かった。そこで、形状変更の容易なストレッチ･シンボルの取り扱いを可能とし、シンボルの属性とマスク図形から、素子認識や接続位置の認識を行うことにより、柔軟なレイアウトモデルに対応可能で実用性の高いコンパクタを開発したので報告する。

# A LAYOUT COMPACTOR FOR STRETCHED SYMBOLS

Takayuki YAMANO-UCHI, Sadahiro TANI, Takashi KAMBE, Ikuo NISHI-OKA
Computer Systems Laboratories, SHARP Corporation
2613-1, Ichinomoto-cho, Tenri-shi, Nara, 632 Japan

As the LSI's complexity has been getting larger, the efficiency of designing mask patterns is more critical. The use of Compactor is an effective method to shorten the layout design cycle and reduce design errors. But the symbolic layout that conventional compactors are able to deal with is restricted, so that designers are obliged to design the layout carefully for the usage of Compactor. This Compactor can handle the stretched symbols whose shapes are changed at the placement, and can recognize the layout components and their connected positions.

## 1. はじめに

LSIのレイアウト設計は、自動レイアウトシステムの普及や、シリコン・コンパイラの登場などにより自動化が進んでいる。しかしチップ面積の点などを考えると完全な形では実現されておらず、人手によるレイアウト作業も依然として行われている。人手による設計は非常に時間がかかるものであり、またLSIの急速な大規模化・高密度化に伴い設計工数は増大しており、この分野での設計期間の短縮が急務となっている。

この分野での効率化の手法として、シンボリック・レイアウトとコンパクション技術を組み合わせた設計手法が数多く発表されてきたが[1]、広く実用化したものは少なく、コンパクション技術はもっぱら自動レイアウトツールの一部として用いられてきた。その理由として以下のような点が考えられる。

i)コンパクタが取り扱うことのできるレイアウトデータは、図形のみで構成するレイアウトと比較してかなり制限されたものであり、一つのシンボルの定義で形状が異なる図形を配置したり、シンボル間の接続位置を自由に選択することができない。また、素子は1つのシンボルとして定義する必要があり、そのシンボル内の図形についての位置の融通がきかないため、配置の際に負担が大きい。

ii)従来のレイアウト設計システムでは、通常のレイアウト・エディタと、コンパクションが使用できるシンボリック・レイアウト・エディタが独立して存在し、設計者は二つのエディタを使い分ける必要がある。また、両者の間でデータの変換が必要である。

iii)シンボリック・レイアウト・エディタ上でコンパクタのための接続情報などを作成しているので、一般のレイアウト編集用コマンドの応答速度が遅くなる。

そこで、従来のシステムの弱点を克服し、レイアウト・エディタとの結合性が高く、レイアウトに対する制約の少ない実用型のコンパクタを開発したので報告する。

## 2. システム構成

本コンパクタは超LSI設計支援システム[2]を構成するサブ・システムの一つである。

本システムはLSIの多品種少量生産時代に備え、設計の抜本的効率化を図るために開発されたもので、EWS上で1つのデータ・ベースのもとに、論理設計からマスク・パターンの検証に至るまでの設計の各段階を支援するプログラム群から成り立っている。
(図1)

設計者はEWS上ですべてのCADツールを操作利用でき、効率的に設計を行うことができる。EWSは互いにネットワークを通じて接続されており、ネットワーク中のスーパーミニコンピュータがネットワーク管理、セル・ライブラリ管理などを行う。さらに超高速コンピュータが大規模データの処理をバッチジョブで行う。

システムは論理設計支援部と、レイアウト設計支援部に大別され、さらにレイアウト設計は、自動レイアウトツールと、人手設計支援ツールに分かれている。コンパクタは、レイアウト・エディタ、DRC、ERCなどの人手設計支援ツールの一つとして位置付けられ、特にレイアウト・エディタとの連携により人手設計を強力にサポートする。

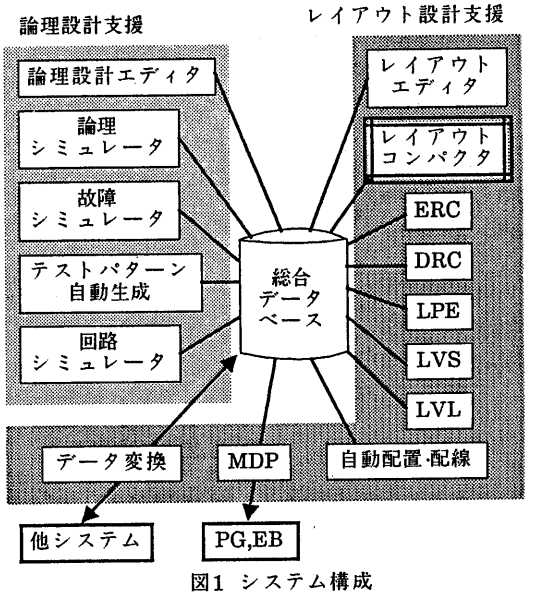


図1 システム構成

## 3.レイアウト･モデル

### 3.1 モデル定義

本コンパクタが取り扱うことのできるレイアウト･モデルを定義する。

シンボルとは一つ以上のマスク図形の集合である。マスク図形は幾何学的情報の他にレイヤー情報を持つ。

シンボルを定義する際に論理的な意味を与える。(例えば、配線、コンタクト、トランジスタ･ゲートなど)

配置時に形状変更の可能なシンボルをストレッチシンボル(以下Sシンボル)と呼ぶ。

素子とは回路構成上機能を果たす複数のマスク図形の集合である。素子は一つのシンボル、またはシンボルの重ね合わせによって定義する。

シンボルの重ね合わせによって素子を構成する場合、論理的な意味と図形のレイヤーの組み合わせを記述する。例えば、論理的な意味がトランジスタ･ゲートで5層の図形を持つシンボルと、論理的な意味が配線で7層の図形を持つシンボルの組み合わせをトランジスタ素子と定義する。

コンパクションの際にSシンボルを縮めるかどうかはそのシンボルの論理的な意味によって決定される。例えば配線と定義されたSシンボルは縮め、トランジスタ･ゲートと定義されたSシンボルは配置時の大きさを保存することができる。

### 3.2 シンボルの種類

シンボルには次の三種類がある。(図2)

(A) シンボルを配置する際に、1つの入力点によってその形状が決定されるもの。
(1点型シンボル)

(B) シンボルを配置する際に、2つの入力点によってその形状が決定されるもの。
(2点型シンボル)

(C) シンボルを配置する際に、2点以上の入力点によってその形状が決定されるもの。
(多点型シンボル)

2点型･多点型のシンボルがSシンボルである。

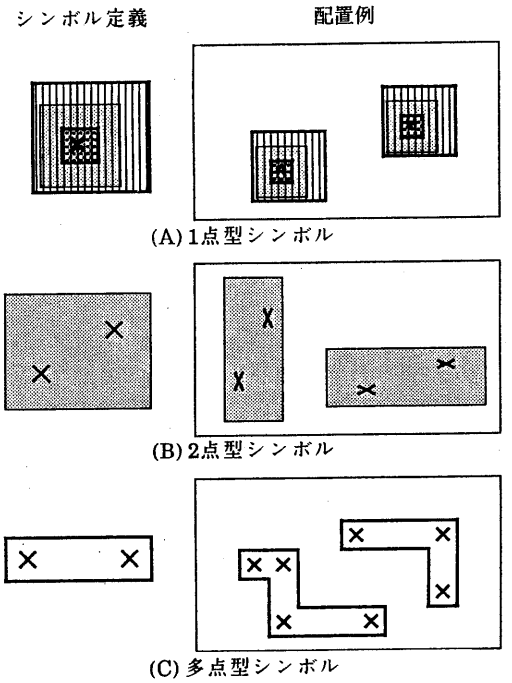



図2 シンボルの種類

### 3.3 シンボルのマスク領域算出

本コンパクタは、素子の認識や、接続の判定、スペーシング･ルールによる位置制約の決定、接続しているシンボル間の相対位置関係の変更を行う際にシンボルが実際に持つマスク図形の領域を利用する。シンボルの入力点に対するマスクの存在領域をシンボルを登録する際に自動的に算出する。

1点型のシンボルの場合、シンボルの形状は一定であり、シンボルのレイヤー毎にマスク図形の上下左右方向の存在領域が記憶される。(図3)

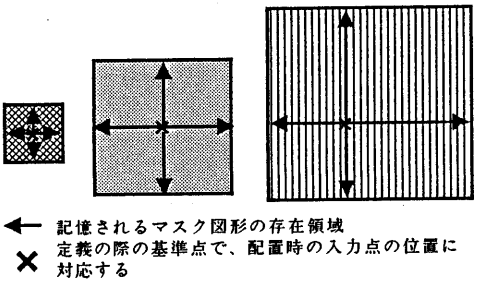



図3 1点型シンボルのマスク領域

2点型シンボルの場合、第一点からのマスクの存在領域と第二点からのマスクの存在領域がそれぞれ別に記憶される。(図4)

また、マスクの存在領域が他の入力点の位置によって決定される場合は、その方向も同時に記憶される。

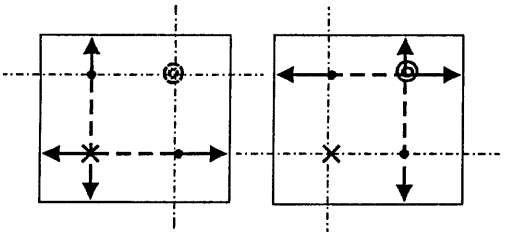



図4 2点型シンボルのマスク領域

多点型のシンボルの場合、入力点の種類が始点、終点、中間点に分類され、それぞれ別に記憶される。(図5)

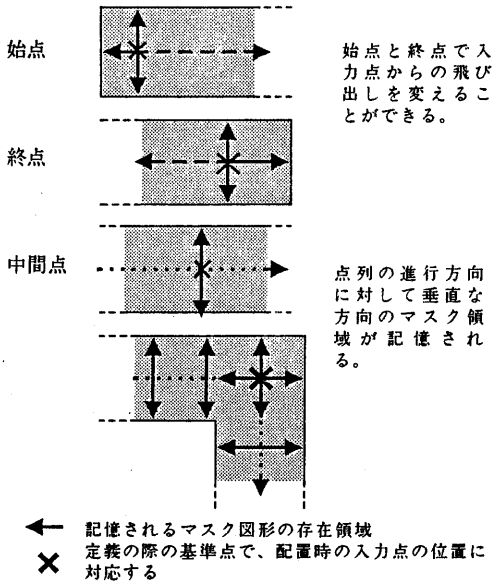



図5 多点型シンボルのマスク領域

この入力点に対するマスク領域の算出により、配置時に形状が決まるSシンボルに対しても、ポリゴン形状の図形を扱うことなく、高速にマスク図形に応じた接続認識、スペーシング値の決定が可能である。

本コンパクタは以上のレイアウト・モデルに対応可能で、特徴として次の点が挙げられる。

i) 一つのSシンボル定義で形状の異なる図形を配置可能。

ii) 複数のシンボルを用いた素子の構成による配置時の負担の軽減。(図6)

iii) シンボル間の接続位置は、マスク図形の接続する位置で配置可能。

iv) 一つのレイアウト・エディタからすべての操作が可能。

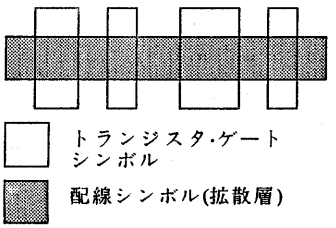



図6 複数シンボルによる素子の配置

## 4.テクノロジー・ファイル

コンパクタが使用するスペーシング・ルールは、テクノロジー毎にあらかじめ準備されているテクノロジー・ファイルに記述されている。テクノロジー・ファイルには、そのプロセスに対するデザイン・ルールが納められているが、コンパクタはその中から各層間および特定のシンボル間のスペーシング・ルール、素子の構成上のルール(例えば組み合わせるべきシンボルの論理的意味付けとレイヤー、MOSトランジスタの場合のゲートに対する拡散のとび出し距離など)といった、必要な項目のみ取り出す。

## 5.アルゴリズム

シンボルをグラフの頂点に代表させ、シンボル間の相対位置制約をグラフの有向枝

(以下単に枝と呼ぶ)として与え、そのグラフの最長径路を求めることによって各シンボルの位置を決定する最長径路法[3]を用いているが、Sシンボルの特徴を利用してより細かくコンパクションを行うため、グラフの構成方法に新しい考え方を導入した。簡単のため以下の説明では水平方向左詰めコンパクションを例にとる。

本コンパクタの処理フローを図7に示す。

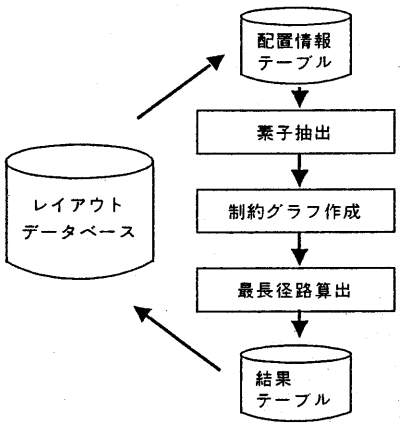



図7 処理フロー

レイアウト・エディタより、レイアウトデータの初期配置情報の出力と、コンパクタの起動を行う。コンパクタはサブ・プロセスとしてエディタと独立に実行されるので、コンパクション中も他のセルに対する修正を行うことができる。次に、コンパクタは素子の抽出、制約グラフの作成、最長径路の算出、結果テーブルの作成を行い、終了の合図をエディタに対して送る。最後に、レイアウト・エディタは結果テーブルを読みこみ、座標値の変更を行う。以下にコンパクタにおける処理手順を示す。

### 5.1 素子抽出

複数のシンボルを重ね合わせて素子を構成する場合には、テクノロジー・ファイルの記述に該当するシンボル間で図形演算を行い、素子を構成するシンボルの組をすべて抽出する。例えば、論理的意味付けの組み合わせがトランジスタ・ゲートと配線で、レイヤーの組み合わせがそれぞれ5層と7層であった場合、トランジスタ・ゲートと定義され5層を持つシンボルと、配線と定義され7層を持つシンボルがすべてリストアップされる。配置情報テーブルとシンボルのマスク領域データより、該当するシンボル内の5層と7層の図形データを算出でき、この図形に対して図形演算を行う。

この手法ではシンボルの論理的意味付けを利用して図形演算の対象となるデータ数を大幅に削減できるので高速に素子抽出が行える。また、素子を構成するマスク・データを別々のシンボルで作成できるので、少数のSシンボルの定義で異なる形状の素子を配置することができレイアウト時間を大幅に削減できる。

### 5.2 制約グラフの作成

#### 5.2.1 グラフ頂点への割り当て

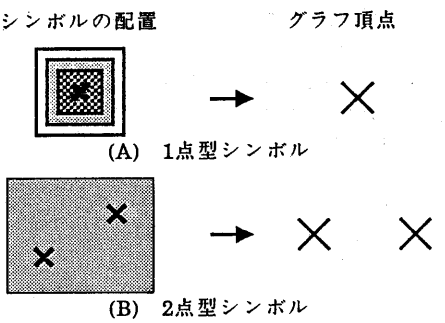
従来の手法では垂直方向に接続されたシンボル群は1つの頂点に割り当てられていた。そのために接続位置や、接続方向が限定されたり、接続されているシンボル間での相対位置の変化ができなかった。そこで、今回の手法では以下の規則で頂点割り当てを行う。(図8)

i) コンパクション領域の左(右)端に対し、$V_{LB}$($V_{RB}$)の頂点を割り当てる。(図9)

ii) 1点型のシンボルに対して1つの頂点を割り当てる。

iii) Sシンボルの場合、X座標の異なる入力点には別の頂点を割り当てる。

iv) セルに対しては、その外形を示すポリゴン形状のデータに対し、X座標の異なる点にそれぞれ頂点を割り当てる。




(A) 1点型シンボル

(B) 2点型シンボル

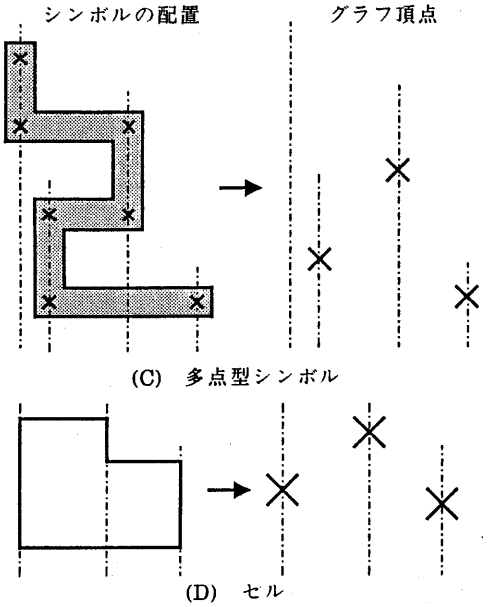



図8 グラフ頂点への割り当て

この規則によって、それぞれのシンボルに対して独立に頂点が割り当てられるので、接続している二つのシンボルを、接続が保証される範囲内で独立に移動させることができ相対位置の変化が可能となる。

5.2.2 グラフの枝の割り当て

(1) 頂点組の選択

位置制約のある2つの頂点の組を選び出すために平面掃引算法を用いている。(図9)

(ステップ1) 各頂点に対して、スペーシング・ルール上の最大値を見込んだY軸方向の領域の上端、下端を算出する。最大値はいかなるシンボルに対してもスペーシング・ルールが守られる値で、コンパクションの結果のY軸方向のエラー防止のために加えておく。

(ステップ2) すべての頂点の上端、下端をY座標の降順にソーティングして点列リストに入れる。

(ステップ3) ワーク・リストを準備する。点列リストから$V_{LB}$、$V_{RB}$を取り出しワーク・リストに挿入する。

(ステップ4) 点列リストから順に頂点を取り出す。その頂点がワーク・リスト内に存在しなければ、ワーク・リストにX座標の小さい順に点列が並ぶように挿入する。そのとき、既にワーク・リストに挿入されている頂点との間で、枝発生の判定(2)(後述)を行う。枝の発生はワーク・リスト上で挿入しようとする位置に近い頂点から判定し、挿入する頂点の示すシンボルと、スペーシング・ルールが存在するすべてのレイヤーについての枝が発生したところでやめる。

(ステップ5) 点列リストから取り出した頂点が既にワーク・リスト内に存在する時はその頂点をワーク・リストから削除する。

(ステップ6) 点列リストから全ての頂点を取り出した時点で頂点組の選択を終了する。

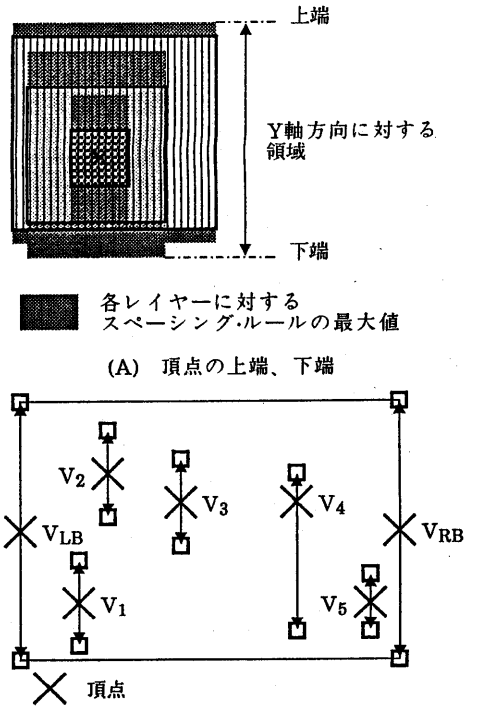



図9 頂点組の選択

(2) 枝の発生と枝に対する重みの付加

2種類の枝を定義する。1つは分離制約枝で、主にスペーシング･ルールによる制約を意味する。他方は接続制約枝で、2つのシンボル間の相対距離を一定以下に保つために用いる。枝の区別は最長径路を求める際に利用する。

選ばれた頂点組に対して枝の発生の判定を行う。頂点組の関係によって、以下のように場合分けし、枝を発生する。

i) 2つの頂点が同一のシンボルを表す時

A) そのシンボルが伸縮可能な場合

枝は発生しない。これによって、配線の折れ曲がり点の相対位置の交換が可能となる。(図10)

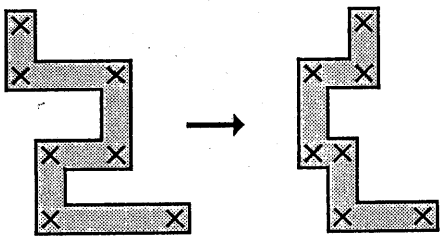


図10 折れ曲がり点の相対位置の交換

B) そのシンボルが伸縮不可能な場合

2頂点間の現状の相対距離を維持するため分離制約と接続制約の2つの枝を発生する。(図11)

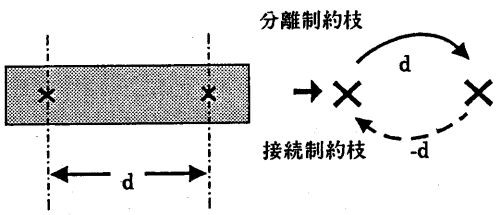



図11 伸縮不可能な同一シンボル

ii) 2つの頂点が異なるシンボルを表す時

C) シンボル間に接続関係がなく、スペーシング･ルールが存在する時

それぞれのレイヤー毎の領域をもとに、必要な距離を算出し、それらの最大値を重みとする分離制約枝を発生する。(図12)

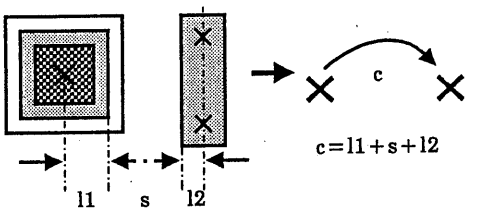



図12 スペーシング･ルールによる制約

D) シンボル間に接続関係が存在し、コンパクション方向に対して接続するレイヤーの一方の図形が、他方を包含している時

相互に接続制約枝を発生し、一方の図形が他方の図形からはみ出ない範囲で移動するように重み付けする。その範囲はマスク領域データより算出する。(図13)

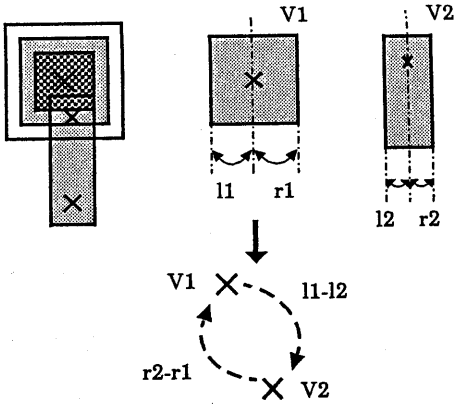



図13 完全に接続するシンボル

E) シンボル間に接続関係が存在し、コンパクション方向に対して接続するレイヤーのどちらの図形も、他方を包含していない時

接続位置の移動によって、最小線幅のルールを犯すことがあるので、B)と同様に現状の相対距離を維持するように枝を発生する。(図14)

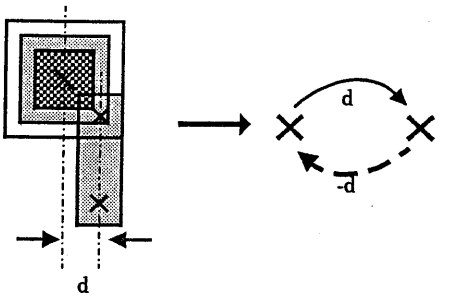



図14　不完全に接続するシンボル

F)　シンボル間にスペーシング･ルールが存在せず、接続関係もない時
　枝は発生しない。

接続の判定は、各シンボルに登録されたマスク領域データにより、簡単に行うことができる。

A)の、折れ曲がり点の相対位置の交換を利用してジョグの発生が可能である。設計者は配線上でジョグを挿入したいポイントを指定する。レイアウト･エディタ上で配線が分割されコンパクション後にはジョグが挿入される。

### 5.3　最長径路法によるシンボル位置の決定

完成したグラフをもとに、頂点$V_{LB}$から各頂点への最長径路を算出し、その時の径路上の枝の重みの総和を各シンボルのX座標とする。

最長径路の算出は分離制約枝に対する処理と、接続制約枝に対する処理に分けられる。[4]

頂点集合をV、枝の集合をE、分離制約枝の集合をE1、接続制約枝の集合をE2とする。頂点 $V_{LB}$から頂点vへの径路上の枝の重みの総和をL(v)、枝e∈E、e=(u,w)に与えられた重みをC(u,w)とする。

(ステップ1)　グラフG1=(V,E1)に対して頂点$V_{LB}$から各頂点への最長径路を算出する。

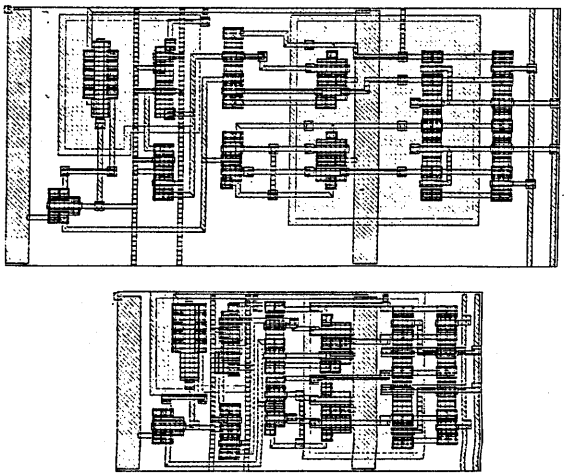
(ステップ2)　e=(v,w)、e∈E2なるすべての枝に対し、

$$L(w) < L(v) + C(v,w)$$
$$\Rightarrow L(w) = L(v) + C(v,w) \qquad (1)$$

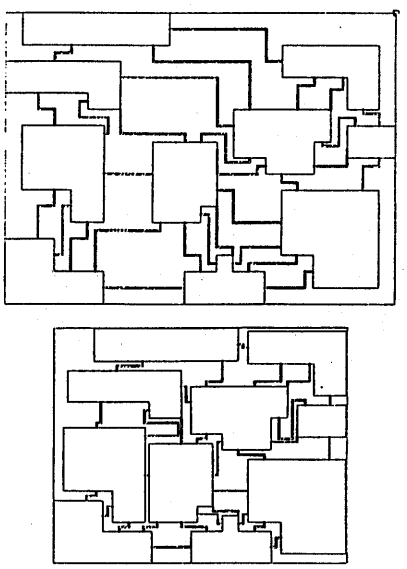
ステップ2で、(1)式の実行が行われたならばステップ1へ戻る。

なおグラフG1=(V,E1)は、その作り方より必ずアサイクリックなグラフとなるので、G1についての最長経路は簡単に求められる。

## 6.結果



(A)　セル内コンパクション



(B)　セル間コンパクション

図15　コンパクション結果

## 7.おわりに

本コンパクタの使用によって、人手によるマスク・パターン設計の期間が短縮され、かつ人手設計に近い面積を得ることができた。

現在、コンパクタによって得られた素子情報や接続情報を利用して、レイアウトと論理の対応付けを行う機能を開発中であり、本手法の様々な応用が考えられる。

[参考文献]

[1] Y.E.Cho,"A Subjective Review of Compaction",Proc.22nd DA Conf. pp.396-404,June 1985.

[2] 谷 他,"超LSI設計支援システム",情報処理学会第33回全国大会予稿集, pp.2261-2262

[3] A.E.Dunlop,"SLIM-The Translation of Symbolic Layouts into Mask Data",Proc 17th DA Conf, pp.595-602,June 1980.

[4] Y.Z.Liao and C.K.Wong,"An algorithm to Compact a VLSI Symbolic Layout with Mixed Constraint", IEEE Trans.on CAD, Vol.CAD-2,No2,pp.62-69,April 1983.

[5] D.G.Boyer and N.Weste,"Virtual Grid Compaction Using the Most Recent Layers Algorithm",Proc. ICCAD-83,pp.92-93,Sept. 1983.