FPGAのマクロブロックを対象とした配置概略配線同時処理手法

井上 大輔　戸川 望　柳澤 政生　大附 辰夫

早稲田大学理工学部電子・情報通信学科
〒169-8555 東京都新宿区大久保3-4-1
E-mail: inoue@ohtsuki.comm.waseda.ac.jp

FPGAのマクロブロックとは，FPGA上で複数の論理ブロックがまとまって1つの機能を実現するブロック集合である．マクロブロックはFPGAの周波数を上げるためやFPGAの論理ブロックの使用率を上げるのに不可欠である．このマクロブロックのレイアウトでは，単純な論理ブロックの交換やレイアウト領域の分割による配置配線は適用できない．本稿では，このようなFPGAのマクロブロックを対象として，トップダウン分割とボトムアップ結合を組み合わせた配置概略配線同時処理手法を提案する．提案手法は，トップダウン分割段階とボトムアップ配置概略配線段階の2段階で構成される．トップダウン分割段階ではマクロブロック間の配線を考慮しながら階層的にレイアウト領域を分割する．ボトムアップ配置概略配線段階では，階層的にマクロブロックを配置概略配線しながら結合する．計算機実験により手法の有効性を評価した結果を報告する．

キーワード　FPGA，マクロブロック，配置概略配線同時処理，トップダウン分割，ボトムアップ結合

A Simultaneous Placement and Global Routing Algorithm for FPGAs with Macro-Blocks

Daisuke INOUE　Nozomu TOGAWA　Masao YANAGISAWA　Tatsuo OHTSUKI

Dept. of Electronics, Information and Communication Engineering
Waseda University
3-4-1 Okubo, Shinjuku, Tokyo 169-8555, Japan
E-mail: inoue@ohtsuki.comm.waseda.ac.jp

A macro-block of FPGAs is a set of preplaced and prerouted logic-blocks which can imprement a logic function such as an adder or a multiplier. Macro-blocks are indispensable to increase the clock freqency and also logic-block utilization of an FPGA chip. This paper proposes a simultaneous placement and grobal routing algorithm for FPGAs with macro-blocks. The algorithm consists of top-down partitioning and bottom-up combining. The top-down partitioning phase is based on hierarchical bipartitioning of a layout region and a set of macro-block. If there exist connections between bipartitioned macroblock sets, pseudo-pins are introduced to perserve the connections. In this phase rough information for macro-block placement and global routing can be obtained. The Bottom-up combining phase combines partitioned layout regions and macro-blocks and determines datailed placement. The experimental results demonstrate the efficiency and effectiveness of the algorithm.

Key Words *FPGA, macro-block, simultaneous placement and routing, top-down partitioning, bottom-up combining*

1 まえがき

FPGA（Field-Programmable Gate Array）は，現在では回路のプロトタイプ，エミュレータに用いられることが多い．しかし，近年システム LSI の汎用性を高める工夫として FPGA を LSI チップのコアとする試みが注目されている．例えば画像処理システムでは，同じ圧縮伸長処理を実行したとしても機器の種類ごとに入出力の信号の規格が異なる．このため，システム LSI を構成するときに汎用的なインタフェース回路を FPGA を使って構成すれば，機器メーカーの開発期間の短縮や在庫管理の効率化などに効果を発揮する．システムの特定機能を実行するコプロセッサでは，機器を使う場面に合わせて，異なる機能を持ったコプロセッサを FPGA で実現すれば，システム LSI に汎用性を持たせることが可能になる．

FPGA にはマクロブロックと呼ばれる，複数の論理ブロックを使用して 1 つの機能を実現するブロックがある．マクロブロックは動作周波数を上げるためや論理ブロックの使用率を上げるために，マクロブロック内部で最適な配置がなされており，加算器，乗算器といった使用頻度の高い回路のマクロブロックを使用することは FPGA の動作周波数を上げるためや，回路全体の論理ブロックの使用率を上げるために不可欠である．

一般に FPGA は論理ブロックおよび配線リソースが少ない．そこで，FPGA のマクロブロックを使用して，論理ブロックの使用率を上げる必要があり，同時に少ない配線リソースでも配線できるように配線混雑度を少なくする必要がある．ところが，配置設計と配線設計を別の段階にすると配置設計段階では配線を含むレイアウト結果を正確に評価することはできず，配線混雑度の最小化は図れない．つまり，マクロブロックを使用した FPGA で配置配線する際に，より配線混雑度を低くするためには，配置設計と配線設計をある程度統合化する必要がある．配置設計と，概略配線設計の統合化を目指した手法として，これまでビルディングブロックを対象とした手法 [2], [3]，スタンダードセルおよび，ゲートアレイを対象とした手法 [4], [6] が提案されている．これらの手法は，いずれも配置設計の処理段階で概略配線設計を取り入れており，その結果として，従来の配置，配線の段階的処理に比較しより良好なレイアウト結果を得ている．FPGA のように規則的な構造を持つデバイスは，文献 [5], [7] のようにその規則性を十分に活用することによりレイアウト設計問題を単純化することができる．文献 [5] では FPGA の Look-Up Table や論理ブロックの交換で配置配線同時処理している．文献 [7] では論理ブロックに回路が割り当てられるまで回路およびレイアウト領域を再帰的に分割し高速な配置概略配線処理を行っている．しかし，マクロブロックを対象としたレイアウトでは複数の論理ブロックをまとめて扱わなければならないため，これらの手法は適用できない．

本稿では，FPGA のマクロブロックを対象として配置処理と概略配線処理を統合化したレイアウト設計手法を提案する．提案手法は，レイアウト領域を階層的に分割しブロックの配置目標を決めるトップダウン分割段階と，階層的にマクロブロックを配置概略配線しながら結合するボトムアップ結合段階の 2 段階からなっている．トップダウン分割段階では，回路全体の概略配線を考慮しながらマクロブロックを概略配置し，分割線上に仮想的なブロックを導入することによって，概略配線径路を保持し，ボトムアップ結合段階の概略配線に利用する．回路全体の概略配線を考慮することにより配線混雑度を低減する．ボトムアップ結合段階では，トップダウン分割段階で求めたマクロブロックの概略配置位置や概略配線径路を利用し，階層的にマクロブロックの位置を決めながらマクロブロック同士を配線する．ボトムアップにマクロブロック同士を結合することによりマクロブロックを扱うことができ，同時に配線混雑度を低減する．本手法を計算機上に実装し，評価実験を行った結果について報告する．

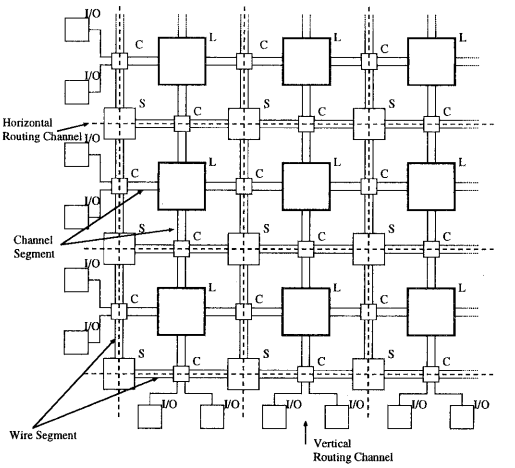

図 1. FPGA.

2 問題の定式化

2.1 FPGA レイアウトモデル

文献 [1], [8] をもとに図 1に FPGA レイアウトモデルを示す．図 1において，L は論理ブロック，C はコネクションブロック，S はスイッチブロック，I/O は外部入出力端子を表す．点線によって区切られた 1 区画を単位格子と呼ぶ．各論理ブロックは，入出力端子を介して隣接するコネクションブロックによりトラックに接続される．論理ブロックの入出力端子は上下左右に存在する．図 2では $I_1, ..., I_{13}$が論理ブロックの入力端子となり，$O_1, ..., O_4$ が論理ブロックの出力端子となる．スイッチブロックは，一種のスイッチボックスであり，プログラムによって水平トラックと垂直トラックを接続する．図 3にスイッチブロックの内部構造を示す．コネクションブロックは論理ブロックの入出力端子と水平・垂直トラックを接続する．図 4にコネクションブロックの内部構造を示す．図において点線は論理ブロックの入出力端子，実線は垂直トラックを表す．外部入出力端子は FPGA の内部と外部との間でデータを入出力する端子で，最外周のコネクションブロックに 2 個接続している．

2.2 マクロブロック

マクロブロックとは，1 つまたは複数の論理ブロックの集合で，1 つのマクロブロックで加算器などの 1 つの論理を実現する．マクロブロック内の論理ブロック位置，およびマクロブロック内の論理ブロック同士の配線径路は予め決まっている．そのため，マクロブロックの形状や端子位置は変更できない．本稿では Xilinx の FPGA ツール XACT のマクロブロック [9] を使用している．図 5に 8 ビット加算器マクロブロックを示す．このマクロブロックは横方向に 1，縦方向に 4 の単位格子を包含している．太線はマクロブロック間の配線を表している．

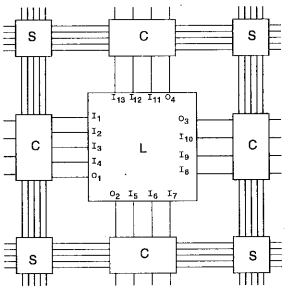

図 2. 論理ブロックの端子位置.

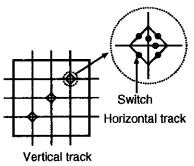

図 3. スイッチブロック.

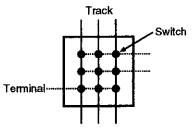

図 4. コネクションブロック.

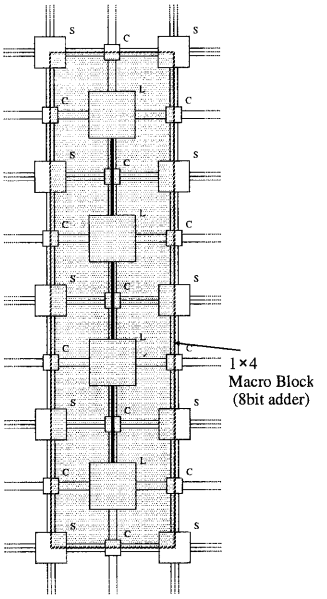

図 5. マクロブロック.

2.3 配置概略配線問題

論理ブロックが配置される場所をスロットと呼ぶ．隣接する2つのスイッチブロック間にあるトラックの集合をチャネルと呼ぶ．チャネルあたりに必要なトラック数を配線混雑度と呼ぶ[1]．ネットの概略配線径路はチャネルの並びによって表す．配置すべき論理ブロックの中で外部入力および外部出力を与えるものを外部入出力端子とする．論理ブロック間の結線要求をネットとし，ネットの集合をネットリストと呼ぶ．

定義 1 FPGA を対象としたマクロブロック配置・概略配線問題とは

(1) 論理ブロック集合及びネットリスト

(2) スロット集合およびマクロブロックの形状，端子位置

が与えられたとき，

(a) マクロブロックが配置されるスロット位置

(b) ネットが経由するチャネル

(c) ネットの端子位置

をネットリストにしたがい，配線混雑度の最小化を目的として決定することである．このとき，外部入力，外部出力は外部入出力端子に割り当てる．

3 アルゴリズムの概要

3.1 用語の定義

アルゴリズムに先立ち，用語を定義する．

部分領域: 隣接した単位格子の集合によって構成される矩形領域.

部分領域境界: 部分領域の外側の境界辺.

カットライン: 一つの部分領域を二つの部分領域に分割する水平または垂直の線分．図 1の格子に沿って引かれる．

仮想端子: カットラインによって分割された二つの部分領域にまたがるネットの接続を保つために，カットラインまたは部分領域の境界線上に置かれる仮想的な端子.

クラスタ: マクロブロックの集合あるいはクラスタの集合．単一のマクロブロックもクラスタと呼ぶ.

グループ: レイアウト領域全体を縦のカットラインで分割した部分領域に割り当てられるマクロブロックの集合．外部入出力端子とマクロブロックの関係に基づき全てのマクロブロック集合をその部分集合に分割することによって，グループが生成される.

目標領域: トップダウン分割段階において，各クラスタが割り当てられた部分領域．各クラスタは必ずしも目標領域内に収まるとは限らない.

割り込み領域: 目標領域内にマクロブロック m が収まらないとき，Cは目標領域以外の部分領域も配置に使用する．目標領域外の部分領域内の m が配置される可能性がある領域を m の割り込み領域とする.

配置配線パタン ボトムアップ結合段階ではクラスタはクラスタ内のマクロブロックの配置位置情報およびマクロブロック間の配線情報を持つ．このマクロブロックの配置位置情報とマクロブロック間の配線情報を配置配線パタンと呼ぶ.

マクロブロックの面積: マクロブロックを包含する最小矩形領域が包含する単位格子の数.

部分領域の面積: 部分領域が包含する単位格子の数.

クラスタ Cの論理ブロック数 $S(C)$: マクロブロック m の面積を $S(m)$ とする．このとき，

$$S(C) = \sum_{m \in C} S(m)$$

と定義する.

[1] 本稿では，トラックがコネクションブロックを貫通する構造をとっているため，配線混雑度はコネクションブロックを通過するトラックの最大数となる.

Step 1. 各マクロブロックをクラスタリングする．

Step 2. 外部入出力端子を配置する．

Step 3. 生成された木のルートのクラスタ C_r と，レイアウト領域全体を部分領域 R_r とし，(C_r, R_r) として待ち行列 Q_P に挿入する．

Step 4. Q_P から節点 (C, R) を取り出す．Q_P が空なら，終了する．

Step 5. R を R_a, R_b に分割するカットラインの方向と位置を決める．

Step 6. $C = \{C_a, C_b\}$ から C_a, C_b を取り出し，C_a を R_a に，C_b を R_b に仮配置して，配線混雑度を求める．

Step 7. Step 6 とは逆に C_a を R_b に，C_b を R_a に仮配置して，配線混雑度を求める．

Step 8. Step 6 と Step 7 の混雑度を比較し，混雑度が少ない割り当て方にクラスタを割り当て，仮想端子を部分領域の周辺に割り当てる．

Step 9. C_a がさらに分割可能であれば，(C_a, R_a) または (C_a, R_b) を待ち行列 Q_P に挿入する

Step 10. C_b がさらに分割可能であれば，(C_b, R_a) または (C_b, R_b) を待ち行列 Q_P に挿入する

Step 11. Step 4 へ．

図 6. トップダウン分割段階のアルゴリズム．

部分領域境界の端子集合: 部分領域境界上のチャネルのいずれかを経由することが決まったネットの仮想端子集合．

部分領域内部の論理ブロック集合: 部分領域内部のスロットが目標領域になった論理ブロック集合．

3.2 基本アルゴリズム

トップダウンに回路を再帰的に分割するだけではマクロブロックの形状を考慮できず，ボトムアップにマクロブロックを結合させていくだけでは，回路全体を考慮した概略配線はできない．マクロブロックを対象とした配置概略配線同時処理手法は，概略配線を考慮しながらマクロブロックの目標領域を割り当てるトップダウン分割段階および，その目標領域内に収まるようにブロックを再帰的に結合させていくボトムアップ結合段階から構成される．トップダウン分割段階では，回路全体の概略配線を考慮しながらマクロブロックの目標領域を決定し，仮想端子を配置することで配線径路を保持する．ボトムアップ結合段階では，目標領域内に収まるようにマクロブロックを配置し，配線混雑度の最小化しながらクラスタ間を配線しながら階層的に結合する．

各段階をそれぞれ解説する．

3.2.1 トップダウン分割段階

マクロブロックが外部入力端子か外部出力端子のどちらに近いかや，マクロブロック間の共有するネットの多さによって，マクロブロック集合から 2 分木を生成し，その木をもとに，概略配線を考慮しながらレイアウト領域を階層的に分割する．

図 6にトップダウン分割段階のアルゴリズムの概要を示す．ここで Step 2 の外部入出力端子の配置は単純に，隣接する 2 辺に外部入力を与える外部入出力端子を，その他の 2 辺に外部出力を与える外部入出力端子を配置するものとする．以下，マクロブロックのクラスタリング (Step 1.) カットラインの方向と位置の決定 (Step 5.)，ブロック集合の分割方法 (Step 6, 7.) について説明する．

マクロブロックのクラスタリング

文献 [8] のマクロブロックは表 1のように縦長の形状である．そこで，この形状を活かしてレイアウト領域を縦のカットラインで分割し，外部入力端子に近いマクロブロックは左に，外部出力端子に近いマクロブロックは右に配置することにより，FPGA 内でデータを左から右に流れるようにする．データの流れる方向が混在し配線混雑度が増加することを防ぐために，データの流れを配置に反映し，データの流れる方向を 1 つに決めることで配線混雑度の低減が期待される．

表 1. マクロブロックの形状．

ブロック名	logic	width	height
ACC16	Accumelator	1	10
ACC8	Accumelator	1	6
ADD16	Adder	1	9
ADD8	Adder	1	4
ADSU16	Adder / Subtracter	1	10
ADSU8	Adder / Subtracter	1	5
COMP16	Comparator	3	2
COMP8	Comparator	2	1
COMPM16	Comparator	1	9
COMPM8	Comparator	1	5
CUP16	-	1	9
CUP8	-	1	5
D7SEG	Decoder	3	2
D7SEGM	Decoder	3	2
DEC2-4E	Decoder	1	2
DEC3-8E	Decoder	2	2
ENCPR8	-	2	2
MUX16-1	Multiplexer	2	3
MUX8-1	Multiplexer	2	2
RD16H	-	1	8
RD8H	-	1	4
RM128X4	RAM	5	5
RM128X8	RAM	5	9
RM64X4	RAM	2	6
RM64X8	RAM	4	6
RM32X4	RAM	2	2
RM32X8	RAM	2	4
RM32X2	RAM	1	1
RM32X4	RAM	1	2
RM32X8	RAM	2	2
RS16P	-	2	5
RS8P	-	1	5

各マクロブロックが外部入力と外部出力のどちらに近いかを求める．外部入出力端子およびマクロブロックを節点，その間のネットを枝とするグラフ G_I を考える．グラフ G_I で外部入力端子から外部出力端子まで幅優先探索し，全てのマクロブロックの端子の外部入力から経由した最小マクロブロック数を求める．同様に，G_I で逆に外部出力から外部入力まで幅優先探索し，全てのマクロブロックの端子の外部出力まで経由する最小マクロブロック数を求める．ここで，マクロブロック a の入出力係数 $I(a)$ を

$$I(a) = \sum_{p \in i(a)} \frac{L_i(p)}{|i(a)|} - \sum_{q \in o(a)} \frac{L_o(q)}{|o(a)|} \quad (1)$$

とする．ここで，$i(a)$ はマクロブロック a の入力端子の集合，$o(a)$ はマクロブロック a の出力端子の集合，$L_i(p)$ は端子 p が外部入力から経由したブロックの数，$L_o(q)$ は端子 q が外部出力まで経由するブロックの数を表す．この値が小さいほど外部入力に近く，大きいほど外部出力に近いといえる．この入出力係数によってマクロブロック集合をグループ $G_1, ..., G_n$ に分割する．G_1 の目標領域は最も左のレイアウト領域に割り当て，G_2 の目標領域は G_1 の右のレイアウト領域に，G_n の目標領域は最も右のレイアウト領域に割り当てる．この作業によって，外部入力端子に近いマクロブロックは左に，外部出力端子に近いマクロブロックは右に配置され，データが左から右に流れるようになる．表 1において

BEGIN

全マクロブロックを入出力係数でソートしてリスト L_iに挿入;

$i := 1$;

$G_i = \emptyset$;

$s := 0$;

while($L_i \neq \emptyset$){

L_iからマクロブロック m を取り出す;

if($\frac{3 \cdot i}{W_l} S(M_{ALL}) < s$){

$i := i + 1$;

$G_i = \emptyset$;

}

$G_i = G_i \cup m$;

$s := s + S(m)$;

}

END

図 7. 入出力係数によるマクロブロックの集合への割り当ての手順.

幅が 3 より大きなマクロブロックは一部の RAM を除いてないため，グループ G_1から G_{n-1}の目標領域はレイアウト領域全体の中で横方向の単位格子数 3 とし，G_nは 2 から 4 とする．各グループ内のマクロブロックの論理ブロック数と目標領域の面積との比を等しくするため，入出力係数 Iの値が小さいマクロブロックから G_1に割り当ていく．この手順を図 7に示す．M_{ALL}は全マクロブロック集合，W_lはレイアウト領域全体の横方向の単位格子数を表す．図 9(a) ではマクロブロック集合は 5 つのグループに分割された．G_1内のマクロブロックの入出力係数が最も小さい．G_5内のマクロブロックの入出力係数が最も大きい．

次に，グループ $G_1, ..., G_n$内のマクロブロックを全てクラスタとして扱い，クラスタ同士を結合し，2 分木を生成する．クラスタ C_a, C_b間の接続度 $E(C_a, C_b)$ を以下の式で算出する．

$$E(C_a, C_b) = \sum_{m_a \in C_a} \sum_{m_b \in C_b} |N(m_a, m_b)| \,/\, S(C_a) \cdot S(C_b)$$

$$C_a, C_b \in G_i (i = 1, ..., n) \quad (2)$$

ここで，$N(m_a, m_b)$ はマクロブロック m_a内の端子とマクロブロック m_b内の端子が共有しているネットの集合を表す．この接続度が高いクラスタ同士を階層的にまとめ，各 $G_1, ..., G_n$内でマクロブロック集合を 1 つのクラスタ $C_1, ..., C_n$にする．この手順を図 8に示す．図 9(b) では C_1を 1 つのクラスタにした後，C_2内のクラスタを結合している．図 9(c) では全てのグループ内での結合が終了している．

$C_1, ..., C_n$同士をクラスタとして階層的にまとめ，図 9(d) のように，全てのマクロブロック集合を 1 つの 2 分木にする．この 2 分木は平衡で，クラスタの前後位置を保っている．

カットラインの方向と位置の決定

部分領域 Rを分割してできる部分領域 R_a, R_bの分割について考える．Step 4 において取り出したクラスタ $C = \{C_a, C_b\}$ を分割した C_a, C_bは，それぞれ R_a, R_bの一方に割り当てられる．Step 5 では分割の方向および Rをどの線で分割するかについて決定する．

カットラインをひく方向は，$C_1, ..., C_n$およびその上位の階層のクラスタの分割では，カットラインは縦のカットラインをひく．その他のクラスタは Rの長辺を分割する方向にカットラインをひく．

BEGIN

$i := 1$;

while($i \leq n$){

$m \in G_i$を全てリスト L_cに挿入;

while($|L_c| > 1$){

$C_a, C_b \in G_i (i = 1, ..., n)$ である全ての C_a, C_b間の $E(C_a, C_b)$ を算出;

$E(C_a, C_b)$ が最大となる C_a, C_bをリスト L_cから取り出す;

$C = \{C_a, C_b\}$ とし，Cを L_cに挿入;

}

$i := i + 1$;

}

END

図 8. 接続度による 2 分木生成の手順.

カットラインをひく境界線の位置を考える．カットラインによって分割された後のクラスタ C_aが割り当てられる部分領域の，カットラインに対し垂直方向の長さ $W_v(C_a)$ を，

$$W_v(C_a) = \frac{S(C_a) \cdot W_v(C)}{S(C_a) + S(C_b)} \quad (3)$$

とする．このカットラインは単位格子の境界線上でなければならないので，$W_v(C_a)$ の値は整数に丸める．

部分領域内部のブロック集合の分割

Step 5 で目標領域 Rの分割する方向と R_a, R_bのどこにカットラインをひくかが決定しており，部分領域の周囲に仮想端子および外部入出力端子が配置されている．Step 6, 7 では，決められたカットラインの方向と位置に基づき，クラスタ C_a, C_bを，部分領域 Rが上下または左右に分割された R_a, R_bに仮配置し，ネット毎に仮想端子を配置しながら部分領域の境界線の配線混雑度が最小になるように配線する．

端子の配置を決定するネットの順番は，端子が存在するクラスタが少ないネット，分割前の境界線上の仮想端子の数が少ないネットの優先順位で仮想端子を配置する．図 11の (a) の中央のブロックを分割する場合を例に仮想端子の配置順序について述べる．C_aはネット a と b の端子を持ち，C_bはネット b と c の端子を持つ．ネットの仮想端子配置順序はネット d は C_a, C_bどちらにも端子が無いので始めに配線し，ネット a, c は C_a, C_bの一方に端子を持つのでその次になるが，分割される領域の境界線上の端子数は，a が 2，c が 3 であるので，ネット a が 2 番目，ネット c が 3 番目となる．最後に C_a, C_b両方に端子を持つネット b の仮想端子を配置する．

各ネット毎に配線混雑度が最小となる径路を探索し，仮想端子をカットライン上とカットラインによって分割される境界領域上，迂回する周辺の部分領域境界上に配置する．部分領域境界ごとに配線混雑度を算出し，配線混雑度の最大値を最小化する．ここでの部分領域境界の配線混雑度 Dは次の式で表す．

$$D = \frac{P_a}{W_a} + \frac{P_b}{W_b} \quad (4)$$

ここで，P_aは分割後に配置された端子の数，W_aは分割後の部分領域境界の長さ，P_bは配置される境界線が未決定の端子の数，W_bは分割前の部分領域境界の長さを表す．図 10では領域 Rを領域 R_a, R_aに分割する．領域 Rの下辺にまだ割り当てられていないネットの端子が 2 つ存在し，領域 R_aの下辺に端子が 1，領域 R_bの下辺に端子が 2 個割り当てられているとき，領域 R_aの配線混雑度

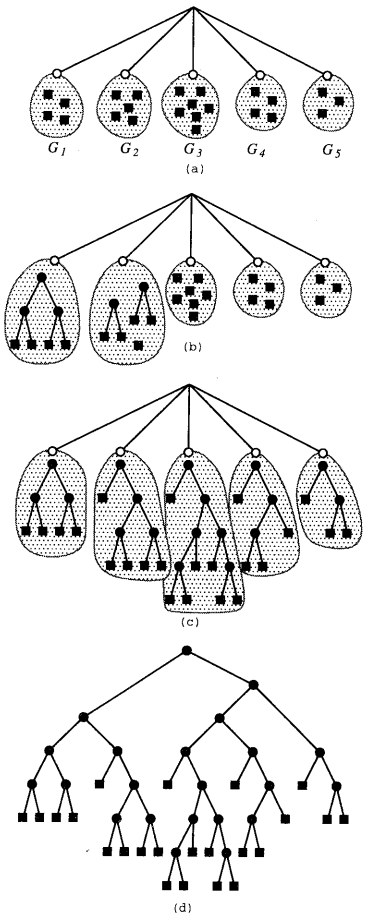

図 9. クラスタ木．白丸はグループ，黒丸はクラスタ，四角はマクロブロックを表す．(a) 入出力係数によってマクロブロック集合を分割．(b) グループ $G_1,...,G_n$ 内のマクロブロック集合を接続度でまとめる．(c)$G_1,...,G_n$ 内のマクロブロック集合を全て木にする．(d) 全てのマクロブロックを 1 つの木にする．

は $2/5+1/2=9/10$，領域 R_b の配線混雑度は $2/5+2/3=11/10$ となる．

分割する部分領域とその周囲の部分領域を節点，その間の境界線を枝とし，ネット毎に全てのクラスタの端子および，仮想端子を通る配線パタンの中で，仮想端子の配置後の境界線の配線混雑度の最大値が最小となる配線パタンを求め，仮想端子を配置する．ここで配線パタンの探索は，最小スタイナ木問題であるが，節点の数が限られているため，計算量は多くはならない．

図 11の (b) でネット c の径路を探索する．仮想端子を全て通る径路を探索した結果，配置後の配線混雑度が最小になる仮想端子の配置結果は (c) になる．

3.2.2 ボトムアップ結合段階

この段階ではクラスタは配置配線パタンを持つ．このクラスタをトップダウン分割段階でのクラスタの分割と逆順にボトムアップにクラスタ同士を結合，配線し，最終的にマクロブロック集合全体を 1 つに結合する．

トップダウン分割段階で求めた目標領域内にマクロブロックが

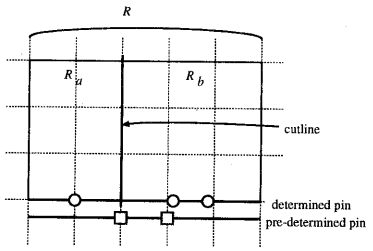

図 10. 回路分割段階での配線混雑度の計算方法

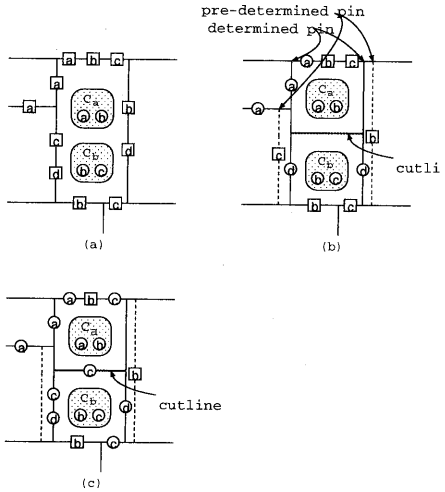

図 11. 仮想端子の配置方法．四角で囲まれた文字はカットラインにより分割された後の仮想端子の配置位置が未決定のネットの仮想端子，丸で囲まれた文字は仮想端子の配置が終了したネットの端子．(a)C_a,C_b の分割前．(b) ネット d,a の仮想端子の配置後，ネット c の最小配線混雑径路探索．(c) ネット c の仮想端子の配置後．

収まるかどうか判定し，収まらない場合は，はみ出してしまう可能性のある隣接する目標領域にはみ出るブロックをうけいれる空間があるか調べる．図 13で，2×3 の目標領域 R_b 内に 1×4 のマクロブロック C_b を挿入する場合，斜線の部分が割り込み領域となる．その後，トップダウン分割段階とは逆に木の葉の部分から目標領域内に収まり，配線混雑度が小さくなるようにクラスタを階層的に結合する．クラスタ C_a,C_b はそれぞれ配置配線パタンを複数持つので，全ての組合せで配置し，クラスタ間を配線混雑度が最小化しながら配線する．配線混雑度が最小となる配置配線パタンをいくつかクラスタに保存し，後の結合時の配置配線パタンとして使用する．

図 12にボトムアップ配置・概略配線段階のアルゴリズムの概要を示す．以下，配置概略配線とその評価 (Step 4.) について説明する．

配置概略配線

Step 3 で取り出された C_a,C_b は配置配線パタンを持っている．配置概略配線段階のアルゴリズムの概要を図 14に示す．この段階は，1.C_a,C_b の配置，2.R 内で C_a,C_b 間の概略配線，3. 配置配線パタンの評価 の 3 段階からなる．配置段階では，与えられた 2 つのクラスタの配置パタンを目標領域内に収まるように全ての

Step 1. 各目標領域と割り当てられたマクロブロックの形状からはみ出る場合は，はみ出る方向とはみ出る単位格子数および他の領域から割り込める空間をもとめる．

Step 2. 領域からマクロブロックがはみ出してしまう方向に割り込める空間が十分かどうか求める．あれば，入れる領域を割り込み領域とする．なければ配置不可能として終了．

Step 3. トップダウン分割とは逆順に，$(C_a, R_a), (C_b, R_b)$ から配置・配線情報を持つクラスタ C_a, C_b と目標領域 R_a, R_b 取り出す．木のルートまで接続が終了したら Step 7 へ．

Step 4. 目標領域 R_a と R_b を結合した領域 R内で，C_a, C_b を全配置パターンで配置・概略配線後評価する．

Step 5. 評価が良い配置概略配線パターンを n 個，クラスタ $C = \{C_a, C_b\}$ の節点の結合結果として保存する．

Step 6. 木の節点に (C, R) の情報を保存する．次の節点に移り，Step 3 へ．

Step 7. 配線混雑度が最小の配置・概略配線パターンが出力となる．

図 12. ボトムアップ配置・概略配線段階のアルゴリズム．

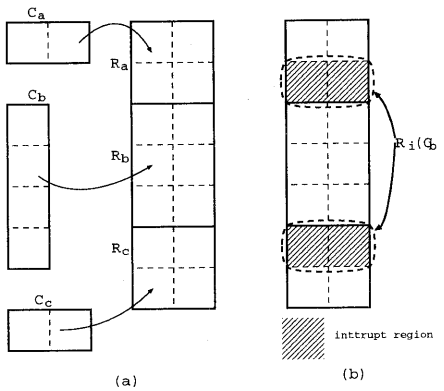

図 13. 割り込み領域．(a)R_aにC_bが割り当てられている．(b)R_bに隣接する領域に割り込み領域が作られる．

組合せパタンを配置する．C_aまたはC_bが1つのマクロブロックである場合は，目標領域内および割り込み領域内で配置可能な全てのパタンをそのクラスタの配置パタンとし，マクロブロック内の配線とあわせて配置配線パタンとする．ただし他への割り込み領域は最小にする．また，他のマクロブロックの割り込み領域は最大にする．

概略配線段階では，配置されたクラスタの端子および概略配線径路情報から配線混雑度を最小化する概略配線径路を求める．径路の探索はネット別に任意の端子を始点，別の任意の端子を終点とし，始点から終点まで配線混雑度が最小となる配線径路を探索し，その径路を概略配線径路とする．多点間ネットは複数の2点間ネットとして，全ての端子を接続するまで径路探索を繰り返す．

この段階での配線混雑度はチャネルを通る配線の本数である．配線径路を求める際の配線可能領域は目標領域 Rと他のクラスタへ飛び出しているマクロブロックの周辺の領域とする．また，R内の他のクラスタ内のマクロブロックからの割り込み領域は配線不可能領域とする．

Step 4.1. クラスタ C_a, C_b を R内で配置する．全パタン配置したら Step 5 へ．

Step 4.2. 配置されたパタンで重みをつけて概略配線する．Step 4.1 へ．

Step 4.3. 全ての配置配線パタンの中で配線混雑度が小さい n 個のパタンを $C = \{C_a, C_b\}$ の配置配線パタンとする．

図 14. 配置概略配線のアルゴリズム．

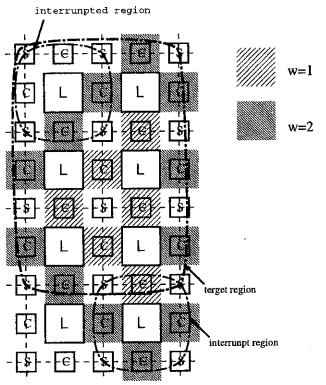

図 15. 配線可能領域と配線の重み．

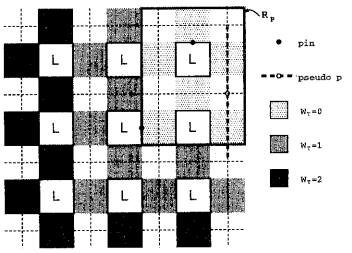

図 16. 配線時の一時的重み．

他のクラスタとトラックを共有することや配線径路が遠回りすることを防ぐことを考慮し，配線に重みをつける．目標領域の他のクラスタとの境界線上のトラックは共有しているため，クラスタの内部のトラックは配線1本あたりの重み $w = 1$ に対して，境界線上のトラックの配線は $w = 2$ とする．他の領域に飛び出しているマクロブロック内部の配線は $w = 1$ とする．図 15に配線可能領域と他のクラスタと共有するトラックの配線の重みについて示す．斜線の領域が配線可能領域である．目標領域の内部は $w = 1$ であるが，他のクラスタと共有する領域は $w = 2$ である．

配線径路は端子と仮想端子が入る最小矩形領域 R_p内の配線混雑度が高いと配線は R_pから出る場合がある．しかし，配線可能領域全体を探索すると計算量が多くなる．探索する領域を少なくするために各ネットの配線時に端子と仮想端子が入る最小矩形領域 R_pから外に径路が出る場合には，径路探索時のみ一時的に R_pからの距離に比例して重みをつける．図 16では配線の対象としている端子および仮想端子は R_p内に収納されている．この配線径路探索時には他の目標領域とのトラックの共有を考慮した配線混雑度の他に図 16内の W_Tの重みが付加される．この重みを付加して配線混雑度が最小となる探索径路を配線径路とする．

C_a, C_b両方に端子が存在するネットを配線する．一方がマクロブロックである場合は端子またはネットは多点間ネットとなるが C_a, C_bともに複数のマクロブロック集合である場合はクラスタ内の同一ネットの端子間は配線されているために，C_a, C_bのそれぞれの任意の端子間を接続すれば，全ての端子を接続できる．つまり2点間ネット配線に置き換えられる．多点間ネットの配線は，

表 2. XACT と提案手法との比較結果

circuit				ours		XACT
name	macro blocks	CLBs	used ratio[%]	density	time[sec]	density
Mult16	166	251	43.6	11	230	14
Mult8	142	190	33.0	10	190	14
Adcomp	3	7	1.2	6	16	7
Add32	2	18	3.1	5	13	5
Rgb2yuv	56	71	12.3	9	117	11

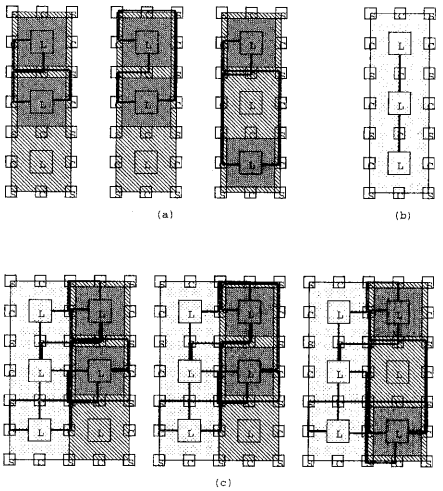

図 17. クラスタの配置と配線．太線は配線径路．(a) クラスタ C_a の 3 つの配置配線パタン．(b) クラスタ C_b．1 つのマクロブロックであるので，配置配線パタンは 1 つである．(c) C_a, C_b 間の配置配線パタンの例．

表 3. Mult16 の配置配線の比較

array size	used ratio [%]	ours	
		density	time[sec]
24 × 24	43.6	11	230
21 × 21	56.9	12	235
20 × 20	62.8	12	235
19 × 19	69.5	13	239
18 × 18	77.4	15	243
17 × 17	86.9	-	-

複数の 2 点間ネットとして配線する．図 17にクラスタ間の配線の例を示す．C_a は 2 つのマクロブロックによって構成されていので，複数の配置配線パタンが存在する．C_b は 1 つのマクロブロックであるので，配置配線パタンは 1 つである．

評価段階では，概略配線段階での重みつき配線混雑度が少ない配置・概略配線パタンを数パタンとりあげ，後の結合段階での候補として木の節点に n 個保存する．実験では $n = 3$ とした．

4 実験結果

提案手法を，MMX Pentium 150MHz 上に C 言語で実装し，計算機実験によって評価した．入力は Xilinx の XACT でマクロブロックを使用し設計した回路を用いた．XACT 上でマクロブロックでない論理ブロックは，1 × 1 のマクロブロックとして配置した．

表 2に XACT の配置配線結果と提案手法を比較しその結果を示す．macro blocks はマクロブロックの総数，CLBs は論理ブロックの総数，used ratio は回路中の論理ブロックの使用率を表す．XACT では論理ブロックが縦 24 列横 24 列の FPGA 上に回路を実装した．提案手法も同様に縦 24 列横 24 列の単位格子上で実験した．Mult16 は 16bit 乗算器，Mult8 は 8bit 乗算器，Adcomp は加算比較器，Add32 は 32bit 加算器，Rgb2yuv は輝度情報色差変換回路である．表 3に MULT16 回路を例にとり，論理ブロックの使用率に関する結果を示す．array size は単位格子の横と縦の列の数を表す．レイアウト領域の大きさを変え，論理ブロックの使用率を変化させた．表 2において，FPGA を対象としたマクロブロックの配置概略配線同時処理手法は XACT に対して配線混雑度を 20%程度低減できた．

表 3では，本手法が論理ブロックの使用率を 80%ほどにできることがわかる．

5 むすび

本稿では，FPGA を対象としたマクロブロックの配置・概略配線同時処理手法を提案した．計算機実験の結果，高速に配線混雑度の低い概略配線結果を得ることができた．

参考文献

[1] S. D. Brown, R. J. Francis, J. Rose, and Z. G. Vranesic, *Field-Programmable Gate Arrays*, Kluwer Academic Publishers, pp. 147–167, 1992.

[2] W. M. Dai, and E. S. Kuh, "Simultaneous Floor Planing and Global Routing for Heirarchical Building Block Layout," IEEE Trans. Comput.-Aided Des. Integrated Circuit & Syst., CAD-6, 5, pp. 828–837, 1987.

[3] H. Muroga, H. Murata, Y. Saeki, T. Hibi, Y. Ohashi, T. Noguchi, and T. Nishimura, "Hierarchical Floorplanning and Detailed Global Routing with Routing-Based Partitioning," Proc. 1990 IEEE Int. Symposium on Circuit and Syst., pp. 1640–1643, 1990.

[4] M. Burstein, S. J. Hong and R. Pelavin, "Hierarchical VLSI Layout: Simultaneous Placement and Wiring of Gate Arrays," Proc. VLSI '93, pp. 45–60, 1993.

[5] S. K. Nag and R. A. Rutenbar, "Performance-Driven Simultaneous Placement and Routing for FPGA's," IEEE Trans. Comput.-Aided Des. Integrated Circuit & Syst., pp. 499–518, 1998.

[6] P. R. Suaris and G. Kedem, "A Quadrisection-Based Combined Place and Route Scheme for Standerd Cells," IEEE Trans. Comput.-Aided Des. Integrated Circuit & Syst., 8, 3, pp.234–244, 1989.

[7] N. Togawa, M. Sato, and T. Ohtsuki, "A simultaneous placement and grobal routing algorithm for FPGAs," in *Proc. ISCAS'94*, pp. 483–486, 1994.

[8] Xilinx, *The Programmable Logic Data Book*, 1994.

[9] Xilinx, *Libraries Guide*, 1994.