社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2006－SLDM－124 (21)
2006／3／17

# クラスタ段数最小化を目的としたLUT型FPGAのパッキング手法

勝木 裕二† 松永 裕介††

† 九州大学 大学院システム情報科学府
〒816-8580 春日市春日公園 6-1
†† 九州大学 大学院システム情報科学研究院
〒816-8580 春日市春日公園 6-1
E-mail: †{katsuki,matsunaga}@c.csce.kyushu-u.ac.jp

**あらまし** FPGA 上に回路を実装する場合，回路を LUT(Look-Up Table) で被覆した後，各 LUT をクラスタと呼ばれる論理ブロックへとまとめるパッキングという作業を行う．ここで，クラスタ間の配線遅延について考えた場合，回路の配置配線が行われた後でないと実際の遅延を厳密に求めることは難しい．そこで，パッキングの段階では各クラスタ間の配線遅延を一定とした遅延モデルが用いられる．この場合，回路の遅延はパッキング後のクラスタの段数によって決定される．本稿では，クラスタ段数の最小化を目的とした FPGA のパッキング手法を提案している．提案手法は，LUT の段数を最小化する既存のマッピングアルゴリズムを基にしており，“ラベル付け” と “パッキング” という二つの工程を経てクラスタ段数が最小となる回路を生成する．ベンチマークを用いた既存アルゴリズムとの比較実験では，平均でクラスタ段数が約 30 %削減されていることを確認した．

**キーワード** FPGA，EDA，パッキング，遅延削減

# A minimum cluster depth packing algorithm for LUT-based FPGA

Yuji KATSUKI† and Yusuke MATSUNAGA††

† Graduate School of Infomation Science and Electrical Engineering,Kyushu University
6-1 Kasuga-koen,Kasuga,Fukuoka 816-8580 JAPAN
†† Faculty of Infomation Science and Electrical Engineering,Kyushu University
6-1 Kasuga-koen,Kasuga,Fukuoka 816-8580 JAPAN
E-mail: †{katsuki,matsunaga}@c.csce.kyushu-u.ac.jp

**Abstract** When the circuit is implemented on FPGA, the circuit is covered with LUT(Look-Up Table) and pack their LUTs into a logical blocks called cluster. The process is called packing. When thinking about the wiring delay between clusters,it is difficult to estimate an actual delay strictly if placement and routing of the circuit has not been done yet. Then, the delay model by whom the wiring delay between each cluster is assumed to be constant is used in packing. In this case, the delay of the circuit is decided depending on a cluster depth. In this paper, it proposes a minimum cluster depth packing algorithm for FPGA. The proposal algorithm refers an existing mapping algorithm that minimizes a LUT depth and it makes the circuit which the cluster depth is minimized through two processes “labelling” and “packing”. In the comparison experiment with an existing algorithm that used the benchmark circuit, it was confirmed that a cluster depth had been reduced about 30 % by the average.

**Key words** FPGA, EDA, Packing, Delay reduction

## 1. はじめに

現在のシステム LSI(Large Scale Integration) 市場において，短期間で多品種少量の製品を送り出す必要性が高まっている．この要求を満たすために，FPGA(Field-Programmable Gate Array) [1] を利用するという手段がある．FPGA とは，チップ上にあらかじめ論理モジュールや配線リソースを用意し，チップ製造後であっても回路の書き換え修正ができる再構成可能な論理デバイスである．

スタンダードセルや MPGA(Mask-Programmed Gate Ar-

ray) で回路を実装する場合，回路ごとに専用のマスクを設計/製作する必要がある．そのため，回路を設計してからチップを製造するまでに約6週間から8週間の期間を必要とする [2]．また，マスクの設計/製作にかかる費用は製造されるチップの個数に関わらず一定である．よって，チップが大量に生産される場合はチップ1個当たりにかかるマスク費用が少なくなりチップの価格を下げることが出来るが，少量生産の場合チップ1個当たりにかかるマスク費用が多くなるためチップの価格が高くなってしまうといった問題がある．

一方 FPGA を用いた場合，あらかじめ製造されたチップに対して回路の構成情報をダウンロードすることで回路を実装する．よって回路ごとに専用のマスクを設計/製作する必要が無く，あらかじめチップが用意されていれば数分で所望の回路を実装することが可能である．また，同一のチップを複数の回路に対して利用することができることから，多品種少量生産に適しているといえる．

FPGA の問題点の一つとして，動作速度が低速であることが挙げられる．FPGA では各配線上に抵抗の高いプログラマブルなスイッチが配置されており，セルベース LSI や MPGA と比べて配線遅延が大きく，また回路遅延のうち配線遅延が支配的となっている．これが FPGA の動作速度を低下させる主な原因となっている [2]．論理ブロック間の配線において，通過するスイッチが多くなるとその分だけ配線遅延が増大するため，通過するスイッチを出来るだけ少なくして FPGA の動作速度を向上させる必要がある．この問題に関して筆者は，FPGA の CAD(Computer Aided Design) の面からの改善を試みている．

多くの FPGA は，LUT(Look-Up Table) を基本構成要素とした構造をとっている．そのため，以下では LUT 型の FPGA について議論することにする．FPGA に回路を実装するための CAD フローは以下の通りである．まず，ハードウェア記述言語等で記述された回路記述からゲートの接続で表現された論理回路を生成する．次に回路の論理最適化を行った後，回路を LUT で被覆する．この作業をテクノロジマッピングという．そして各々の LUT をクラスタと呼ばれる複数の LUT のかたまりからなる論理ブロックにまとめ，論理ブロック間の接続で表現された論理回路を生成する．この作業をパッキングという．最後に論理ブロックを FPGA 上に配置し，各論理ブロック間の配線を行う．

パッキングの段階で回路遅延を削減することを考えた場合，回路遅延を削減するためにはクラスタの段数を少なくすればよい．ここで言うクラスタの段数とは，クラスタからなる回路におけるクリティカルパス上のクラスタの個数のことである．クリティカルパスとは，回路の外部入力もしくは回路内部のフリップフロップの出力から，回路の外部出力もしくは回路内部のフリップフロップの入力までの経路の中で，最も遅延が大きいパスのことである．クラスタ内部の接続はクラスタ外部 (クラスタ間) の接続よりも高速であるので，出来るだけ多くのクリティカルパス上の LUT を一つのクラスタにまとめ，クラスタ間の接続を減らすことで回路遅延を削減することが出来る．しかし，パッキングでは任意の LUT を同一のクラスタにまとめることはできないという問題がある．これは，同一のクラスタで用いられる LUT の異なる入力の本数はある値以下でならなければならないという制約があるためである．

本稿では，このような制約下でクラスタの段数が最小となるパッキング手法を提案する．提案手法では，LUT の段数が最小となるような既存のマッピングアルゴリズムをクラスタへのパッキングに応用することで，クラスタ段数の最小化を実現している．本稿では，ベンチマーク回路に対して提案手法と既存のパッキングアルゴリズムを用いてそれぞれパッキングを行い，パッキング後のクラスタ段数を比較する実験を行った．

本稿の構成は以下の通りである．2章で FPGA の構成及び FPGA の CAD フローについて述べる．3章で提案手法についての解説を行う．4章で提案手法の評価実験について述べる．5章で本稿をまとめる．

## 2. FPGA

### 2.1 FPGA の構成

現在多くの FPGA は，LUT(Look-Up Table) を基本構成要素とした構造をとっている．図1に，FPGA の構成として一般的なアイランド型アーキテクチャの構成図を示す．図1において，CLB(Clustered Logic Block) は論理ブロック (クラスタ)，S はスイッチブロック (S ブロック)，C はコネクションブロック (C ブロック)，I/O は I/O ブロックをそれぞれ表している．CLB は N 個の BLE(Basic Logic Element) で構成されており，BLE は k 入力の LUT とフリップフロップとで構成されている．図2，3に，BLE 及び2個の BLE を内部に含む CLB の構成図を示す．

図1において，各 CLB 間の水平，垂直方向には S ブロックと C ブロックを含んだ配線チャネルがひかれている．C ブロックは配線チャネルを通って CLB の入出力ピンに接続されており，S ブロックはチャネルが直交する部分の配線と接続している．図2において，MUX はマルチプレクサ，FF はフリップフロップをそれぞれ表している．k 入力の LUT(k-LUT) は k 入力のマルチプレクサと $2^k$ ビットの SRAM とで構成されており，入力の組み合わせにより SRAM の内容を出力する．BLE はこの LUT の出力もしくはフリップフロップの出力のどちらかを出力とする．図3において，点線で囲まれた部分が CLB である．CLB 内部のそれぞれの BLE は，CLB 外部からの入力及び BLE の出力からのフィードバックを入力としている．

### 2.2 CAD フロー

FPGA を用いた設計の CAD フローを図4に示す [2]．

CAD フローの最初の手順としてまず，ハードウェア記述言語等で記述された回路記述から，ゲートの接続で表現された論理回路を生成する．ここで，テクノロジ非依存のレベルで多段論理回路をモデル化するためにブーリアンネットワーク [3] と呼ばれる有向非循環グラフが用いられることがある．よってこれ以降では，ゲートの接続で表現された論理回路をブーリアンネットワークで表現することにする．この得られた回路から冗長な論理を取り除くことにより論理最適化を行う．その後，回路をあらかじめ決められた入力数以下の部分回路に分解する．

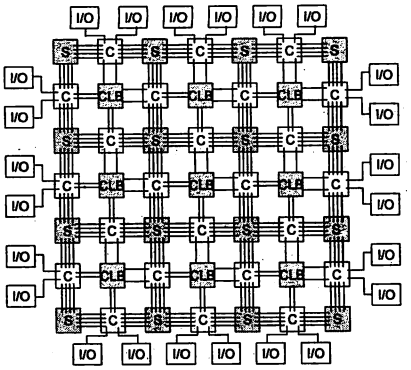


図 1　アイランド型アーキテクチャの構成図

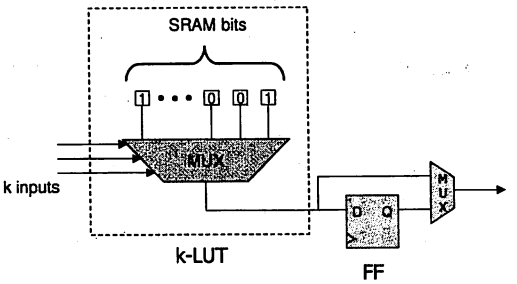


図 2　BLE(Basic Logic Element) の構成図

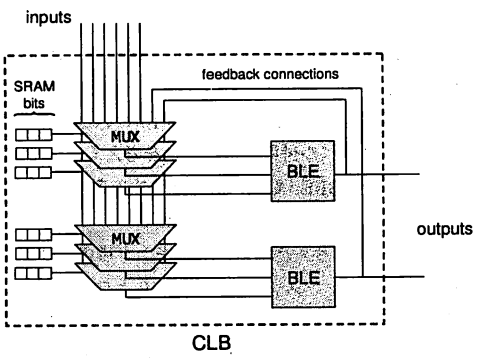


図 3　CLB(Clustered Logic Block) の構成図

この分解を行う理由は，この後回路を LUT で被覆することを考えた場合，元の回路に LUT よりも大きな入力数をもつゲートが存在しているとそのゲートを LUT で被覆することができなくなってしまうからである．次に，入力として与えられた回路を LUT で被覆するテクノロジマッピングという作業を行う．各々の部分回路を一つの LUT で実現できるかどうかは，その部分回路の機能とは無関係に入力数のみで判断することが出来る．そして，入力として与えられた LUT，及び FF の接続で表現された論理回路においてまず LUT，及び FF をひとまとめにすることで BLE を生成し，さらに BLE をひとかたまりに

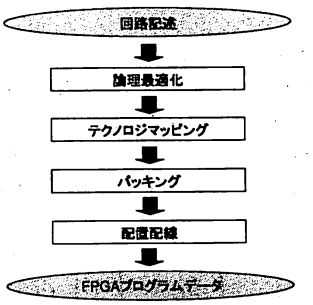


図 4　FPGA の CAD フロー

したクラスタと呼ばれる論理ブロックを作り，クラスタの接続で表現された論理回路を生成するパッキングという作業を行う．ただしここでは制約として，1 つのクラスタにまとめることができる BLE の個数の上限，及びクラスタの入力数の上限が与えられる．これをクラスタ制約という．パッキングは，クラスタ制約を満たす範囲で行われなければならない．最後に論理ブロックを FPGA 上に配置し，各論理ブロック間の配線を行う．配置・配線も回路の遅延を決定する重要な要素であるが，本稿では配置・配線に関しては検討していない．以上の手順を経て，FPGA のプログラムデータが得られる．

## 3.　クラスタ段数最小化手法

本章では，クラスタ段数の最小化を実現するパッキング手法を提案する．提案手法のアルゴリズムでは，LUT の段数最小化を実現するマッピングアルゴリズム “FlowMap” [4] を基に，パッキング後の回路のクラスタ段数の最小化を図っている．以下では，まず提案手法について解説する上で必要となる用語及びパラメータの定義を行った後，提案手法についての説明を行う．

### 3.1　定　　義

本稿では，BLE は内部に FF を含まない，LUT と等価なものとして議論を進める．BLE の接続で表現された論理回路において，BLE ネットワーク $N = (V, E)$ とは各 BLE をノード，各 BLE 間の接続を枝とするブーリアンネットワークと定義する．また，$t \in N$ とし，$t$ から PI ノード $v$ へと辿るパス上にあるノードを $u$ としたとき，$u$ は $v$ の子孫であるという．このとき，$N_t = \{u \in V | u$ は $v$ の子孫 $\}$ を $N$ におけるノード $t$ を根とする部分グラフと定義する．

BLE をクラスタへとパッキングする際，任意の BLE を同一のクラスタへとパッキングすることはできない．これは，BLE をクラスタへパッキングするためには満たさねばならない制約があるからである．これをクラスタ制約という．クラスタ制約には以下の 2 種類がある．

- 一つのクラスタ内に含まれる BLE の個数の上限
- 一つのクラスタの入力数の上限

これ以降では，一つのクラスタ内に含まれるBLEの個数の上限を$N$，一つのクラスタの入力数の上限を$L$という記号で表すことにする．

ここで，クラスタからなる回路のクラスタの集合を$C_{net}$=$\{C_1,C_2,\cdots,C_m\}$とし，クラスタ内のBLEの集合を$C_j$=$\{b_1,b_2,\cdots,b_{n_j}\}$する．クラスタ$C_j$内のBLE $b_i$の入力の集合を$I(b_i)$=$\{u_1,u_2,\cdots,u_k\}$，クラスタ$C_j$の出力の集合を$O(C_j)$としたとき，クラスタ$C_j$における入力の集合$I(C_j)$は次式で表される．

$$I(C_j)=\bigcup_{b_i\in C_j} I(b_i)-\left(O(C_j)\cap\left(\bigcup_{b_i\in C_j} I(b_i)\right)\right) \quad (1)$$

式(1)右辺の第二項は，クラスタ内部のBLEの入力の集合と出力の集合の共通部分である．クラスタ内部のBLEの出力がクラスタ内部のBLEの入力となっている場合，この入力はクラスタ内部の信号線となるためクラスタの外部入力とはならない．このとき，クラスタ制約は以下の式で表される．

$$\forall C_j\in C_{net},\ n\leqq N,\ |I(C_j)|\leqq L$$

### 3.2 提案手法

提案するパッキング手法は，2つの段階に分類される．第一段階はラベル付け，第二段階はパッキングである．以下ではそれぞれの段階について説明する．

提案手法ではまず，入力として与えられるBLEネットワークにおいて，各ノードにラベルという値をつけることから始める．ノード$t$におけるラベルという値は，BLEネットワーク内のノード$t$を根とする部分グラフ$N_t$においてクラスタ段数最小となるようにクラスタへとパッキングしたときの，ノード$t$が含まれるクラスタの段数を表す値である．ノード$t$のラベルは$l(t)$という記号で表すことにする．ラベルは入力側のノードから順に全てのノードに付けられる．

ラベル付けアルゴリズムでは，BLEネットワークに関してPIノードからトポロジカル順にラベル付けを行う．各PIノード$u$に関して，PIノードのラベルは$l(u)$=0とする．ラベル付けを行うノードを$t$とし，ノード$t$を根とする部分グラフを$N_t=(V_t,E_t)$とする．この$N_t$において，$B$=$\{b_1,b_2,\cdots,b_n\}$を最大のラベル$p$を持つノードの集合とする．ノード$b_i$の入力の集合を$I(b_i)$，$B$の出力の集合を$O(B)$とすると，$B$をクラスタにしたときの外部入力の集合$I(B)$は以下の式で表される．

$$I(B)=\bigcup_{b\in B} I(b_i)-\left(O(B)\cap\left(\bigcup_{b\in C_j} I(b_i)\right)\right)$$

このとき，ノード$t$の入力の集合を$I(t)$とすると，ノード$t$を$B$とクラスタへとパッキングしたときの，ノード$t$を内部に含むクラスタの外部入力$I(B_t)$は以下の式で表される．

$$I(B_t)=I(t)-(O(B)\cap I(t))$$

以上より，ノード$t$を集合$B$の全てのノードと同一のクラスタへとパッキングするために満たさねばならないクラスタ制約は以下の式で表される．

$$|B|+1\leqq N \quad (2)$$

$$|I(B)|+|I(B_t)|\leqq L \quad (3)$$

式(2)は，一つのクラスタ内に含まれるBLEの個数の上限の制約である．ノードの集合$B$に対してラベル付けを行うノード$t$を加えた集合の絶対値が$N$以下であれば，この制約は満たされる．式(3)は，一つのクラスタの入力数の上限の制約である．集合$B$の入力の集合及びノード$t$の入力の集合からパッキング後にクラスタ内部の信号線として扱われる入力を取り除いた入力の総和が$L$以下であれば，この制約は満たされる．

以上より，クラスタ制約が満たされる場合と満たされない場合では，ノード$t$のラベル$l(t)$は以下のように付けられる．

- クラスタ制約が満たされる場合　：$l(t)=p$
- クラスタ制約が満たされない場合：$l(t)=p+1$

クラスタ制約が満たされる場合，ノード$t$はノードの集合$B$と同一クラスタへとパッキングすることが出来る．このとき，ノードの集合$B$の入力側にあるノードの最大のラベルは$p-1$であるので，$B$と同一クラスタへとパッキングすることが出来るノード$t$のラベル$l(t)$は$p$と付けられる．クラスタ制約が満たされない場合，ノード$t$はノードの集合$B$と同一クラスタへとパッキングすることが出来ないので，ラベル$l(t)$は$l(t)\geqq p+1$である．一方，ノード$t$の入力ノードの最大のラベルは$p$であるので，$l(t)\leq p+1$である．以上より，クラスタ制約が満たされない場合のラベル$l(t)$は$l(t)=p+1$と付けられる．

図5にラベル付けの例を示す．A～EはBLEを，BLEの横の数字はそのBLEのラベルを，a～kはそれぞれ異なる信号線を表している．図5において，$N$=4，$I$=10というクラスタ制約を与えてBLEにラベル付けを行うことを考える．図5(a)ではラベル付けを行うBLEはA，最大のラベルを持つBLEはB，C，Dである．AとB，C，Dの BLEを一つのクラスタにパッキングすることを考えた場合，パッキング後のクラスタ内のBLEの個数は4，クラスタの入力数は10となり，これはクラスタ制約を満たしている．よってAはB，C，Dと同一のクラスタにパッキングすることが可能であるので，Aのラベルは2と付けられる．図5(b)についても同様に考える．ラベル付けを行うBLEはA，最大のラベルを持つBLEはB，C，D，Eである．AとB，C，D，Eの BLEを一つのクラスタにパッキングすることを考えた場合，パッキング後のクラスタの入力数は9となりこれは制約$L$を満たしているが，クラスタ内のBLEの個数は5となり，これは制約$N$を満たさない．よってこの場合，Aのラベルは3と付けられる．

提案手法のラベル付けアルゴリズムの擬似コードを図6に示す．

次に，先ほど求めたラベルの値を基にクラスタへのパッキングを行う．BLEネットワークにおいて出力側のノードのうちラベルの値が大きいものからパッキングを始め，そのノードを根とする部分グラフにおいて同じラベルを持つノードを同一のク

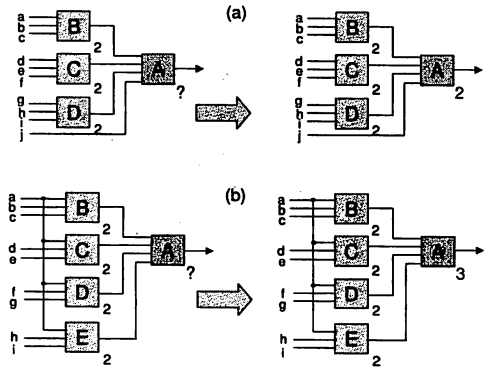


図 5　ラベル付けの例

```
labeling_algorithm (Graph G){
    G の各節点を入力から出力方向へトポロジカル順で整列
    for 各 PI ノード v に関して {
        l(v) := 0;
    }
    for G 内の各ノード t に関して {
        N_t : ノード t を根とする部分グラフ;
        p := max { l(u)|u ∈ N_t };
            B : N_t のうち，ラベル p を持つノードの集合;
        I(B) = ⋃_{b_i∈B} I(b_i) − ( O(B) ∩ ( ⋃_{b_i∈B} I(b_i) ) );
        I(B_t) = I(t) − (O(B) ∩ I(t));
        if |B| + 1 ≦ N かつ |I(B)| + |I(B_t)| ≦ L
            l(t) := p;
        else
            l(t) := p + 1;
    }
}
```

図 6　ラベル付けアルゴリズム

ラスタへとパッキングしていく．

パッキングアルゴリズムでは，BLE ネットワークに関して先ほど求めたラベルの値を基に，PI ノード側のノードのうち最大のラベルを持つノードから処理を行う．パッキングはラベルの値ごとに行われ，同一ラベルのノードに関して全てパッキングが終わった後，次のラベルのノードに関してパッキングを行う．これをラベルが 0(PI ノード) になるまで繰り返す．

パッキングを行うノードのラベルを $p$，ラベル $p$ を持つノードの集合を $T$ とする．パッキングは $T$ 内のノードに関して出力からのトポロジカル順に行われる．パッキングを行う $T$ 内のノードを $t$ とし，ノード $t$ を根とする部分グラフを $N_t$ とする．この $N_t$ において，$B$ をラベル $p$ を持つノードの集合とする．そしてノード $t$，ノードの集合 $B$ を一つのクラスタへとパッキングし，クラスタ $C$ を生成する．このとき，$T$ を $T-B$ と更新し，パッキングが行われたノード $b_i$ に関しては，ノード $b_i$ を根とする部分グラフに関するパッキングが行われないようにする．集合 $T$ 内の全てのノード $t$ についてパッキングが終了した後，ラベル $p-1$ のノードに関してパッキングを行う．

パッキングを行っていく段階で図 7 のようにノードが別々のクラスタに重複して含まれる場合がある．このノードを片方のクラスタのみにパッキングした場合，図 7(a) のようにクラスタ段数が増加してしまう．そこで，別々のクラスタに重複して含まれるノードがある場合は，図 7(b) のようにそのノードの複製を作成して両方のクラスタにパッキングされるようにする．このように複製を作成することで，回路全体のノードの個数は増えるがクラスタ段数の増加は防ぐことが出来る．

全てのノードに関してパッキングが完了したら，次は各クラスタに関して，クラスタ同士を併合してもクラスタ制約を満たすものに関してはクラスタ同士を併合するという作業を行う．この作業は回路全体のクラスタの総数を減らすために行われるが，任意のクラスタ同士を併合しても，元はそれぞれが独立したクラスタであるのでクラスタ段数が増加することはない．回路全体のクラスタの集合を $Q$ としたとき，クラスタ $C_i$ に関してクラスタ $C_j$ との併合を試みることを考えると，以下のクラスタ制約を満たせばクラスタ $C_i$，$C_j$ を併合することが出来る．

$$|C_i \cup C_j| \leqq N$$

$$|I(C_i) \cup I(C_j)| \leqq L$$

この併合作業は，$Q$ 内のクラスタ全てについて行われ，上記のクラスタ制約を満たすクラスタ $C_i$，$C_j$ の組がなくなるまで行われる．

提案手法のパッキングアルゴリズムの擬似コードを図 8 に示す．以上の作業を経て，提案手法でパッキングされたクラスタの接続で表現された回路が生成される．

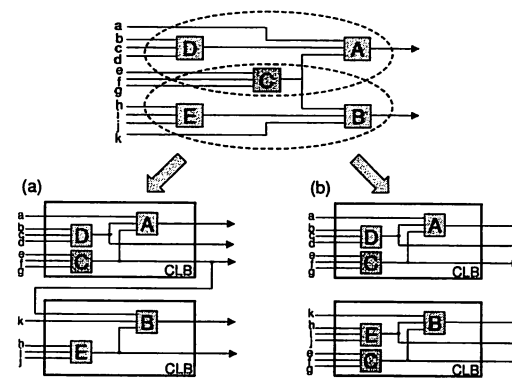


図 7　ノードパッキング

## 4. 実　　験

今回の実験は，11 の MCNC ベンチマーク回路 [6] を用いて行った．このベンチマーク回路を SIS [7] と呼ばれる論理合成ツールで論理最適化を行い，FlowMap を用いて LUT で回路を被覆する．この回路に対して，既存のパッキングアルゴリズムである “T-VPack” [5] 及び提案手法を適用してクラスタへのパッキングを行い，クラスタ段数及び回路のクラスタの総数について比較する．

```
packing_algorithm (Graph G){
    p := max{l(u)|u ∈ G};
    while p ≠ 0 {
    Q := φ;
    T : 出力からのトポロジカル順にソートされた
ラベル p を持つノードのリスト;
        while T ≠ φ {
            t : リスト T の先頭要素;
            N_t : ノード t を根とする部分グラフ;
            B : N_t のうち，ラベル p を持つノードの集合;
            C : クラスタ;
            C := {t} ∪ B;
            Q := Q ∪ {C};
            T := T − C;
        }
    p := p − 1;
    }

    for each C_i, C_j ∈ Q {
        if |C_i ∪ C_j| ≤ N かつ |I(C_i) ∪ I(C_j)| ≤ L{
            C_i := C_i ∪ C_j
            Q := Q − C_j;
        }
    }
}
```

図 8　パッキングアルゴリズム

実験結果を図 9，10 に示す．図 9 は各ベンチマーク回路を T-VPack 及び提案手法でパッキングした時のクラスタ段数を表しており，図 10 はクラスタの総数を表している．

図 9 より，提案手法では 11 のベンチマーク回路全てにおいて T-VPack でパッキングした回路よりもクラスタ段数が削減されていることが確認できた．クラスタ段数の平均削減率は 31.8 %であった．また図 10 より，回路全体のクラスタの総数は 11 のベンチマークにおいて約 20 %から 50 %増加していることが確認できた．

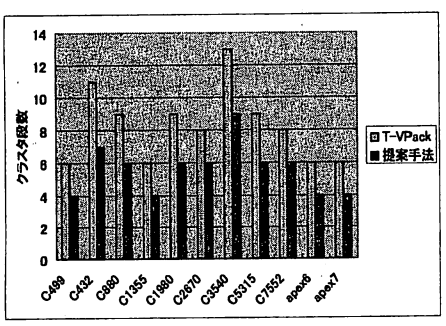


図 9　実験結果 (クラスタ段数)

## 5. おわりに

本稿では，FPGA の回路遅延削減のためにパッキングにおけるクラスタ段数最小化手法を提案し，既存アルゴリズム

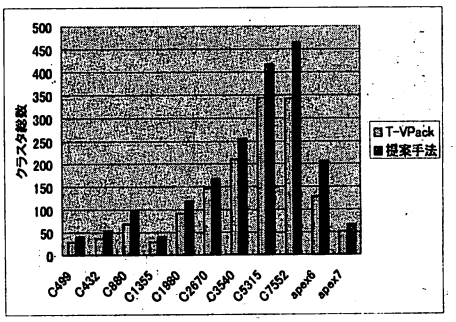


図 10　実験結果 (クラスタ総数)

"T-VPack" との比較実験を行った．実験より，提案手法では T-VPack よりもクラスタ段数を平均で 31.8 %削減することに成功したが，回路全体のクラスタの総数では約 20 %から 50 %増加するという結果を得た．FPGA では，チップが製造された段階でチップ上に実装することのできる論理ブロックの上限は決まっており，実装する回路の論理ブロック数がその上限を超えない範囲であるならば，既存アルゴリズムよりもクラスタ段数が少ない提案手法のほうが回路の遅延削減が見込める．今後は，クラスタ段数とクラスタの総数のトレードオフについて考える必要がある．

**謝辞**

本研究の一部は，科学研究費補助金 (学術創成研究費 (2))(課題番号：14GS0218)，及び文部科学省知的クラスター創成事業研究補助金による．

### 文　献

[1] S.D.Brown,R.J.Francis,J.Rose,and Z.G.Vranesic, "Field-Progurammable Gate Arrays" Kluwer Academic Publishers,1992.

[2] V.Betz,J.Rose,and A.Marquardt, "Architecture and CAD for Deep-Submicron FPGA" Kluwer Academic Publishers,1999.

[3] R.K.Brayton,R.L.Rudell,A.Sangiovanni-Vincentelli, and A.R.Wang, "MIS:A Multiple-Level Logic Optimization System" IEEE transaction of CAD,Vol.CAD-6,No.6.pp.1062-1081.Nov.1987.

[4] J.Cong and Y.Ding, "FlowMap:An Optimal Technology Mapping Algorithm for Delay Optimization in Lookup-Table Based FPGA Designs" IEEE trans.CAD,Jan.1994,pp.1-12.

[5] A.Marquardt,V.Betz,and J.Rose, "Using Cluster-Based Logic Blocks to improve FPGA Speed and Density" ACM/IEEE International Synposium on FPGAs,Feb 1999.

[6] S.Yang, "Logic synthesis and optimization benchmark user guide version 3.0" MCNC,Jan.1991.

[7] E.M.Sentovich et al, "SIS:A System for Sequential Circuit Analysis" Tech.Report No.UCB/ERL M92/41,University of California,Berkeley,1992.