再構成可能デバイスMPLD/SePLDにおける設計アルゴリズムについて

谷川 一哉[1,a)]　弘中 哲夫[1,b)]

概要：本研究室で開発している再構成可能デバイス MPLD(Memory-based Programmable Logic Device)/SePLD(Selector-based Programmable Logic Device) 用のテクノロジマッピング，配置アルゴリズムについて紹介する．再構成可能デバイス MPLD/SePLD では，FPGA とは異なり，プログラム可能な配線資源を持たず，論理ブロック間を直接接続したアレイ構造を持つ．そのため，配置アルゴリズムでは単純に論理ブロックに論理セルを配置するだけでなく，論理ブロックを配線として使用する事を考慮した配置が必要になる．また，再構成可能デバイス SePLD の論理ブロックでは，3 入力までの任意の論理関数を実現できるが，4〜6 入力までの論理関数は一部の論理関数のみが実現できるという制約がある．そのため，SePLD 用のテクノロジマッピングでは，その制約を考慮したマッピングアルゴリズムが必要となる．本稿では，SePLD 用のテクノロジマッピング，および，MPLD/SePLD 用の配置アルゴリズムについて評価を行い，現状の問題点について考察する．

Design Algorithm for Reconfigurable Device MPLD/SePLD

Tanigawa Kazuya[1,a)]　Hironaka Tetsuo[1,b)]

Abstract: In this paper, we will explain about technology mapping and placement algorithm for the reconfigurable device, Memory-based Programmable Logic Device(MPLD) and Selector-based Programmable Logic Device(SePLD), that have been developed in our laboratory. MPLD/SePLD don't have programmable routing resource unlike FPGAs and have array structure that consists of directly connected logic blocks. Therefore, a placement algorithm is required to consider not only placement of logic cells into logic blocks but also using logic blocks as routing resources. The logic block in SePLD can implement any 3-inputs logic functions, but there is a limitation on implementation for 4 to 6 inputs logic function. Therefore, the limitation needs to be considered on technology mapping. In this paper, we evaluate the algorithms and describe the problems of these algorithms.

1. はじめに

近年では, FPGA (Field Programmable Gate Array) などの再構成可能デバイスが広く利用されており, 我々の研究室ではその 1 種として, MPLD (Memory-base Programmable Logic Device)[1] や SePLD(Selector-based Programmable Logic Device)[2] と呼ばれる新型の再構成可能デバイスの研究開発を行なっている．MPLD/SePLD は FPGA と同様に論理回路を再構成することができるデバイスであるが，その構造が異なる．FPGA は LUT とスイッチブロック，コネクションブロックなどによって構成され，それらの間を接続するための再構成可能な配線資源が用意されている．それに対し，MPLD/SePLD は，論理を構成する論理ブロックを平面上に配置し，それらのブロック間を専用の配線で直接配線した構成をとる．この構成により，MPLD/SePLD では，全体の面積における論理領域の割合を高め，一定面積内に構成できる論理回路の規模を大きくする事を狙っている．

このような再構成可能デバイス MPLD/SePLD を効率的に使用するには，効果的な設計アルゴリズムが必要にな

[1] 広島市立大学
HCU, Asaminami, Hiroshima 731–0134, Japan
[a)] tanigawa@hiroshima-cu.ac.jp
[b)] hironaka@hiroshima-cu.ac.jp

る．SePLD の論理素子 SLE では，3 入力までの論理関数であれば任意の論理関数が実現できるが，4〜6 入力までの論理関数の場合，一部の論理関数しか実現できないという制約がある．そこで，SePLD 用のテクノロジマッピングでは，シャノン展開を用いてマッピング不可能な論理関数をマッピング可能な入力数の少ない論理関数に分解するアルゴリズムを使用した．

また MPLD/SePLD では論理ブロックが直接接続された構造をとるため，配置対象の論理回路に応じて，論理ブロックを配線として使用する事が必要である．それを実現するために，配置アルゴリズムに SA 法を使用し，コスト関数において，配線の混雑度を考慮したコスト関数を使用した．そこで本稿では，再構成可能デバイス MPLD/SePLD 用に開発しているこれらのアルゴリズムを紹介し，それらの評価結果，および，今後の課題について述べる．

本稿の構成を以下に示す．まず次節では再構成可能デバイス MPLD/SePLD について説明し，続いて，3 節ではそれらのデバイス用のテクノロジマッピングと配置アルゴリズムを紹介する．4 節では，3 節で紹介したアルゴリズムの評価結果を示し，5 節で本稿についてまとめる．

2. 再構成可能デバイス MPLD/SePLD

本節では再構成可能デバイス MPLD/SePLD について説明する．最初に，両デバイスに共通する基本構成について説明した後，MPLD/SePLD を特徴づけている論理ブロックである MLUT(Multiple-output Look-Up Table), SLB(Selector based Logic Block) について説明する．

2.1 基本構成

MPLD/SePLD は，LB(Logic Block) と呼ばれる論理ブロックを AD 対と呼ばれる LB 間専用配線によって接続することで構成されている．**図 1** には例として AD 対数 6 の基本構成を示している．MPLD/SePLD では，AD 対数と LB 同士を繋ぐ AD 対の接続パターンよってアーキテクチャが変化する．MPLD/SePLD は，LB の作り方によって異なるアーキテクチャになっており，MPLD では LB に MLUT と呼ばれる LB を使用し，SePLD では LB に SLB と呼ばれる LB を使用する．MLUT/SLB の詳細については，次項で説明する．

2.2 MLUT

MLUT の基本構成を**図 2** に示す．一般的なメモリは図 2 に示すようにアドレス線 (addr) とデータ線 (data) を持っており，読み込み時にはアドレス線に書き込みたいデータのアドレス値を入力することでメモリの各アドレスに書き込まれたデータを出力する．MLUT として用いるメモリはアドレス線とデータ線双方の幅が等しいメモリを用いる．また MLUT のデータ線 (data) の出力には，マルチプレクサがついており，DFF に保存するか，あるいは，スルーして出力するか選択することができるが，図 2 では省略している．

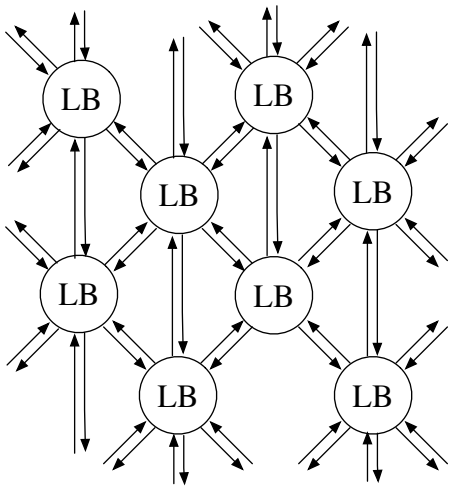

図 1 MPLD/SePLD の基本構成
Fig. 1 Basic Structure of MPLD/SePLD

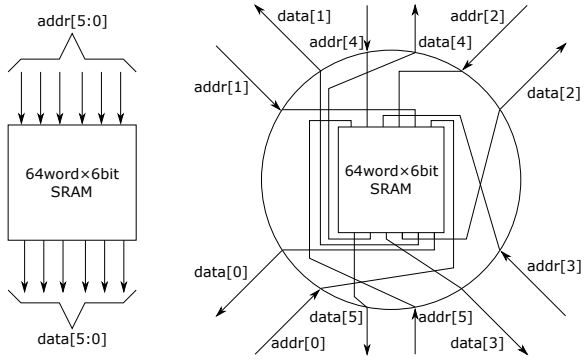

図 2 AD 対 6 の MLUT の構成
Fig. 2 Structure of MLUT with 6 AD pair

AD 対 n の MLUT は n 入力 n 出力の LUT という構成であるため，入力変数が共通な n 個の論理変数から構成される論理関数を n 個マッピングする事ができる．

2.3 SLB

SLB の構成を**図 3** に示す．SLB は SLE6(6-input Selector Logic Element) が 1 個，出力用クロスバ，構成情報用メモリ (図 3 内の黒い四角) によって構成されている．出力にクロスバを設けることで，入力を任意の箇所に出力することができる．

SLE6 は既存研究である MUX4LE[3] を改良した論理素子である．SLE6 の構成を**図 4** に示す．SLE6 は入力用クロスバ，構成情報用メモリ (図 4 内の黒い四角)，4 個の 2 入力セレクタ，1 個の 4 入力セレクタから構成される．構成情報用メモリからの出力を 2 入力セレクタで選択し，それらの出力を 4 入力セレクタで選択することで論理を構成する．また，MUX4LE と同様に SLE6 は入力にクロスバ

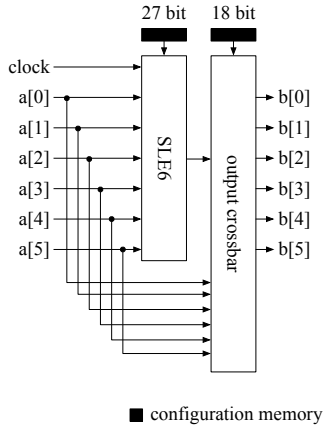

図 3 SLB の構成
Fig. 3 Structure of SLB

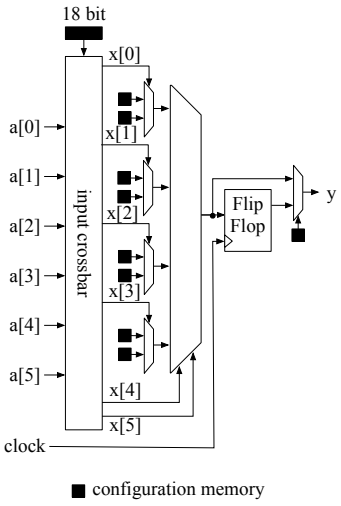

図 4 SLE6 の構成
Fig. 4 Structure of SLE6

を設置することで入力の位置制約を無くし，入力箇所を自由に選択可能にしている．このような構成にすることにより，論理関数の実現能力はそのままに，MUX4LE よりも少ないトランジスタ数で構成することができる．

SLE6 では，既存研究の MUX4LE と同様に，「任意の 3 入力までの論理関数」と「いくつかの 4 から 6 入力までの論理関数」をマッピングできる．一般的に，6 入力の論理関数はシャノン展開により，以下のように変形できる．

$$f(x_5,...,x_0) = \overline{x_1}\ \overline{x_0} f_0(X) + x_1\overline{x_0} f_1(X) + \overline{x_1} x_0 f_2(X) + x_1 x_0 f_3(X) \quad (1)$$

この時，SLE6 では $f_0(X)...f_3(X)$ までの各論理関数が 1 変数しか持たない場合のみ，論理関数をマッピングできる．例えば，$f_0(X)...f_3(X)$ の各論理関数が共通して x_2 を入力として持つ場合，任意の x_2 の値に対して，(x_0, x_1) の値の組み合わせ全てとの論理積を取っているため，任意の 3 入力の論理関数が実現できる．同様に 4 入力の論理関数を実現する場合には，$f_0(x_2)$，$f_1(x_2)$，$f_2(x_3)$，$f_3(x_3)$ というような場合が考えられるが，論理変数 x_2，x_3 に対して，(x_0, x_1) の値の全ての組み合わせは実現できないため，実現可能な論理関数に制限がある事になる．

```
Input:    lut_list     // list of LUTs in a target circuit
Output: sle_list     // list of SLEs converted from LUTs

sle_list.clear();
while ( lut_list != {} ) {
   new_lut_list.clear();
   foreach lut in lut_list {          // take one LUT from LUT list
      // check whether a LUT can be mapped to one SLB
      if( canBeMappled2SLB(lut) ) {
          sle_list.add(lut);                  // add the LUT to SLE list
      } else {
          // decompose a LUT to three LUTs by Shannon decomposition
          lut0, lut1, lut2= decompose_expr(lut)
          new_lut_list.add(lut0);  // add decomposed LUTs to LUT lists
          new_lut_list.add(lut1);
          new_lut_list.add(lut2);
      }
  }
  lut_list = new_lut_list;
}

return sle_list;
```

図 5 テクノロジマッピングアルゴリズム
Fig. 5 Technology Mapping Algorithm

3. 設計アルゴリズム

本節では，MPLD/SePLD に論理回路を実現するために必要となる設計アルゴリズムについて説明する．前節で述べたように，SePLD で使用される論理素子 SLE は 6 入力までの任意の論理関数がマッピングできるわけではないため，それを考慮したテクノロジマッピングが必要である．そこで，最初に SePLD 用のテクノロジマッピングのアルゴリズムについて説明する．その後，MPLD/SePLD 特有の基本構成に対応した配置アルゴリズムの説明をする．

3.1 SePLD 用テクノロジマッピング

MPLD の論理ブロックである MLUT は AD 対が n の場合，n 入力の任意の論理関数がマッピングできるため，FPGA 用のテクノロジマッピングをそのまま使用することができる．しかしながら，SePLD の論理ブロックである SLB は 6 入力までの論理関数がマッピングできるが，4〜6 入力までの論理関数でマッピングできる論理関数には制約がある．そのため，4〜6 入力までの論理関数に対して，SLB にマッピングできるかどうか判定し，SLB にマッピングできない場合には，3 入力以下の論理関数に分解する必要がある．

SePLD 用のテクノロジマッピングのアルゴリズムを**図 5** に示す．本アルゴリズムの入力には，ターゲットとする論理回路から $I(4 \leq I \leq 6)$ 入力以下の LUT のネットリストにテクノロジマッピングされた回路を使用する．本アルゴリズムでは，LUT のネットリストから 1 つずつ LUT を抽

```
s = s0;                              // s: current sol., s0: initial sol.
sb = s;                              // sb: best sol.
c = cost(s);                         // c: current cost, cost(s): cost of s
T = T0;                              // T: current temp., T0: initial temp.
M = M0;                              // M: number of loops, M0: initial M

while( T > Tend) {                   // repeat until T reaches the terminal temp.
   loop_cnt = 0;
   while( loop_cnt < M ) {           // repeat neighbor generation
      sn = neighbor(s);              // make a neighbor solution
      cn = cost(sn);                 // calculate the cost of the neighbor

      if( rand() < e^{(c-cn)/T}) {   // accept check ( comparing the costs)
         s = sn;                     // update the current solution
         c = cn;
         update_best_sol(sb, s);     // update the best solution if necessary
      }
      loop_cnt = loop_cnt + 1;
   }
   T = α*T;                          // cooling
   M = β*M;
}

return sb;
```

図 6 配置アルゴリズム
Fig. 6 Placement Algorithm

出し，抽出された LUT が 1 つの SLB にマッピング可能かどうかをチェックしている．もし抽出された LUT がそのまま SLB にマッピング可能であれば，そのまま出力のネットリストに登録する．抽出された LUT が SLB にマッピングできない場合はシャノン展開を用いて 3 つの論理関数に分解する．具体的には，抽出された LUT で表現される論理関数が 6 入力の論理関数である場合，その論理関数を $f(x_0, ..., x_5)$ と表し，x_0 によりシャノン展開すると，

$$f(x_0, ..., x_5) = x_0 f_0(x_1, ..., x_5) + \bar{x_0} f_1(x_1, ..., x_5)$$

と表せ，2 つの 5 入力論理関数 $f_0(x_1, ..., x_5)$，$f_1(x_1, ..., x_5)$ と 1 つの 3 入力論理関数 $f(x_0, f_0(), f_1())$ に分解できる．分解された 3 つの論理関数は，それぞれまた入力の LUT リストに追加され，SLB にマッピング可能かどうかチェックされる．このようにして入力の LUT が全て SLB にマッピングされると本アルゴリズムは SLB のリストを返して，終了する．

3.2 配置アルゴリズム

MPLD/SePLD で実装したい論理回路はテクノロジマッピングで論理セルにマッピングされた後，配置処理により，論理セルの物理的な位置を決定する．配置処理で使用されるアルゴリズムは SA(Simulated Annealing) 法 [4] をベースとしている．MPLD/SePLD 用の配置アルゴリズムを**図 6** に示す．

コスト関数は SA 法において最も重要な要素であり，コスト関数がいかに良解を表すかで，最終的に得られる解の質が左右される．MPLD/SePLD では論理ブロックが互いに隣接した配置状態にある．そのため論理セルを隣接した論理ブロックに配置することで，論理セルを密集させることができる．一方で，論理セル間の配線を実現するためには，配線のための論理ブロックを適度に確保する必要がある．

MPLD/SePLD 用の SA 法で採用しているコスト関数を以下に示す．

$$cost = p \times length + q \times congestion + r \times nearness$$

ここで $length$ は総配線長，$congestion$ は配線混雑度，$nearness$ は配線領域の確保をそれぞれ評価するもので，p,q,r はそれぞれのパラメータに対する重み付けの定数である．$length$ の値が小さくなる事で，回路全体をより少ない論理ブロックを用いた配置結果になる一方で，論理セル同士が近くに配置されると $congestion$, $nearness$ の値が上昇し，論理セル同士が過度に密集しすぎるのを防ぐようなコスト関数となっている．MPLD/SePLD で採用している SA 法の詳細については，紙面の都合上，論文 [1] を参照して頂きたい．

4. 評価

本稿ではテクノロジマッピングと配置アルゴリズムに関する評価を行う．その後，これらのアルゴリズムにおける問題点についてまとめる．

4.1 テクノロジマッピングの評価

4.1.1 評価内容

SePLD 用のテクノロジマッピングの効果を確認するために，オープンソースで開発されている FPGA 用 CAD ツール VTR[5] で使用されている ABC ツール [6] をテクノロジマッピングに用いて，論理セル数の比較評価を行う．具体的には，ABC ツールで 3 入力までの論理セルにマッピングした結果と，3.1 節で説明した提案アルゴリズムによりマッピングした結果を比較する．提案アルゴリズムの入力となる LUT のネットリストは，ABC ツールにより 4,5,6 入力の LUT にマッピングされたネットリストを使用する．対象とするベンチマーク回路は ISCAS'85, ISCAS'89 ベンチマークである．

4.1.2 評価結果

表 1 にテクノロジマッピングの評価結果を示す．同表において，各項目はテクノロジマッピング後の論理セル数を表す．また，「ABC の 3 入力」とは ABC ツールを用いて 3 入力までの論理セルにマッピングした結果を示し，「提案手法の 4,5,6 入力」は，提案手法のテクノロジマッピングによりマッピングした結果である．

表 1 より，提案テクノロジマッピングでは，ABC ツールで 3 入力までの論理セルにマッピングした結果よりも多くの論理セルを使用してることが分かる．SLB では 3 入力までの論理セルであれば任意の論理関数が実現できるた

表 1 テクノロジマッピングの評価結果
Table 1 Result of Technology Mapping

回路	ABC	提案手法		
	3 入力	4 入力	5 入力	6 入力
C17	5	6	6	6
C432	137	276	271	622
C499	122	206	230	600
C880	218	424	627	752
C1355	122	206	230	600
C1908	194	311	603	809
C2670	421	571	825	804
C3540	659	1039	1830	2536
C5315	904	1356	1923	2273
C6288	952	1473	2984	3569
C7552	1009	1445	2429	3192
s820	169	256	256	449
s832	189	289	289	452
s838	208	341	341	334
s953	269	466	466	828
s1238	370	648	648	1036
s1423	269	465	465	1036
s1488	378	634	634	995
s1494	368	608	608	999
s5378	707	1216	1216	2253
s9234	578	979	979	1753
s38417	8856	11472	11472	17016
s38584.1	8520	12187	12187	20315

め，提案手法によるテクノロジマッピングを使用しない方が良い結果となっている．これは提案手法ではマッピングできない論理関数をより入力数が少ない論理関数に分解するだけであったが，分解された論理関数同士を合成して，マッピング可能な論理関数に再構成する必要があったのではないかと考えられる．

4.2 配置アルゴリズムの評価

4.2.1 評価内容

MPLD/SePLD 用配置アルゴリズムの性能評価の為に，配置後に配線が成功した場合の配置にかかった時間，及び，コスト値を評価した．評価環境は以下の通りである．

- プロセッサ: Quad-Core AMD Opteron(tm) Processor 2384 @ 2.7 GHz
- OS : RedHat5
- メモリ: 64GB
- コンパイラ: gcc 4.1.2 (-O2)

また，評価に用いる回路は，ISCAS'85, ISCAS'89 ベンチマーク回路のうち，テクノロジマッピング後の論理セル数が 100 個以上となる回路を用いる．テクノロジマッピングには，1 つ目の評価において，論理セル数が最も少なくなった ABC ツールによるマッピング結果を使用した．また，MPLD/SePLD ではテクノロジマッピング後においては配置アルゴリズムにおける違いはないため，本評価ではターゲットデバイスには MPLD を仮定している．ターゲットデバイスには論文 [1] で使用したアレイサイズが 33×36, 63×60, 93×90 を使用した．

表 2 配置アルゴリズムの評価結果
Table 2 Evaluation Result of Placement Algorithm

回路	ゲート数	論理セル数	MPLD サイズ	論理セルの割合	時間 [s]
c499	202	115	33*36	10%	17
c880	383	151	33*36	13%	20
c1335	546	110	33*36	9%	23
c1908	880	145	33*36	12%	48
c2670	1193	355	63*60	9%	147
c3540	1669	488	63*60	13%	172
c5315	2307	668	93*90	8%	602
c6288	2416	738	93*90	9%	653
c7552	3512	597	93*90	7%	607
s820	289	157	33*36	13%	22
s832	287	161	33*36	14%	23
s838	390	158	33*36	13%	26
s953	395	236	63*60	6%	73
s1238	508	317	63*60	8%	109
s1423	657	305	63*60	8%	92
s1488	653	346	63*60	9%	139
s1494	647	338	63*60	9%	139
s5378	2779	754	93*90	9%	729
s9234	5597	576	93*90	7%	603
s13207	7951	1419	93*90	17%	1294
s15850	9772	1776	93*90	21%	2406

4.2.2 評価結果

表 2 に配置アルゴリズムの評価結果を示す．同表において，論理セル数はテクノロジマッピング後の論理セル数を，MPLD サイズは配置が成功した時の MPLD のアレイサイズを，論理セルの割合は全 MLUT 数に対する MLUT の割合を示す．

最初に論理セルの割合に着目すると，最大でも 21%と低い割合にとどまっている．これは現在の SA アルゴリズムで使用しているコスト関数において

(1) nearness の値によって全ての論理セル間の間に配線のためのスペースを確保している，
(2) congestion の計算において混雑度を Binding Box を用いた概略計算をしており，その結果，混雑度を高く見積もりすぎている，

ためだと考えている．次に実行時間に着目すると，s15850 回路に対して，2406 秒の時間がかかっており，配置アルゴリズムの高速化も必要であると考えられる．

4.3 評価のまとめ

テクノロジマッピングの評価結果より，提案アルゴリズ

ムでは効率的な SLB マッピングが実現できていない事が分かった．効率的なマッピングを実現するためには，提案アルゴリズムに対して，論理関数の分解後，再度，論理関数を合成する事でマッピング後の論理セル数の削減を図る手法などの検討が必要と考えている．

配置アルゴリズムの評価結果より，効率的な配置が行えていない事と，計算時間が膨大になっていることが分かった．効率的な配置アルゴリズムを実現するために，コスト関数の変更 [7] や SOM を用いた配置手法 [8] などを検討している．

5. まとめ

本研究室で開発している再構成可能デバイス MPLD/SePLD 用のテクノロジマッピングと配置アルゴリズムを紹介した．SePLD の論理素子 SLE では，3 入力までの論理関数であれば任意の論理関数が実現できるが，4〜6 入力までの論理関数の場合，一部の論理関数しか実現できないという制約がある．そこで，SePLD 用のテクノロジマッピングでは，シャノン展開を用いてマッピング不可能な論理関数をより入力数の少ない論理関数に分解するアルゴリズムを使用した．このような提案テクノロジマッピングに対して，ISCAS ベンチマーク回路を用いて，論理セル数の変化を評価した．この評価結果では，提案テクノロジマッピングを使用する事で，使用している論理セル数が増加し，効率的なテクノロジマッピングが実現できていないことが分かった．

また MPLD/SePLD では論理ブロックが直接接続された構造をとるため，配置対象の論理回路に応じて，論理ブロックを配線として使用する事が必要である．それを実現するために，配置アルゴリズムに SA 法を使用し，コスト関数において，配線の混雑度を考慮したコスト関数を使用した．このような配置アルゴリズムに対して，ISCAS ベンチマーク回路を用いて配置配線結果を評価した．この評価結果より，MPLD/SePLD の全体の論理ブロックのうち使用している論理セルの割合が，最大 21%であり，比較的，低い値となった．また論理ゲート数 9772 ゲートの論理回路 s15850 に対して，配置が完了するまでの実行時間が 2406 秒かかる事がわかり，アルゴリズムの高速化が必要である事が分かった．

今後の課題は，テクノロジマッピングにおいて，より効率的なテクノロジマッピング手法の調査や開発であり，配置アルゴリズムに対しては，より効率的に配置できるようなコスト関数の開発と配置アルゴリズムの高速化などが挙げられる．

参考文献

[1] NAKAMURA, M., INAGI, M., TANIGAWA, K., HIRONAKA, T., SATO, M. and ISHIGURO, T.: A Physical Design Method for a New Memory-Based Reconfigurable Architecture without Switch Blocks, *IEICE Transactions on Information and Systems*, Vol. E95.D, No. 2, pp. 324–334 (online), DOI: 10.1587/transinf.E95.D.324 (2012).

[2] 山本啓輔，谷川一哉，弘中哲夫， 石黒隆：セレクタを用いた小面積な論理ブロック SLB の提案，FIT2016 第 15 回情報科学技術フォーラム， Vol. 第 1 分冊, pp. 15–22 (2016).

[3] Chin, S. A. and Anderson, J. H.: A case for hardened multiplexers in FPGAs, *2013 International Conference on Field-Programmable Technology (FPT)*, pp. 42–49 (online), DOI: 10.1109/FPT.2013.6718328 (2013).

[4] Laarhoven, P. J. M. and Aarts, E. H. L.(eds.): *Simulated Annealing: Theory and Applications*, Kluwer Academic Publishers, Norwell, MA, USA (1987).

[5] : Verilog To Routing, available from (https://verilogtorouting.org)(accessed 2017-07-12).

[6] University of California: ABC: A System for Sequential Synthesis and Verification, available from (https://people.eecs.berkeley.edu/ alanmi/abc/)(accessed 2017-07-12).

[7] 荒瀬郁実，窪田昌史，谷川一哉，弘中哲夫：細粒度再構成可能デバイス MPLD の論理セル配置問題を対象とした SA 法における迷路法を用いたコスト算出方法，信学技報，RECONF2017-15, Vol. 117, No. 46, 北海道，pp. 75–80 (2017).

[8] 田中智大，谷川一哉，弘中哲夫， 石黒隆：細粒度再構成可能デバイス MPLD の IO を考慮した SOM ベース配置手法，信学技報，RECONF2016-30, Vol. 116, No. 210, 富山，pp. 29–34 (2016).