卒業論文

題目

# マルチポートメモリ生成ツールの<br>メモリマクロ自動配置アルゴリズムの改良

指導教員

近藤 利夫 教授

2015年

三重大学 工学部 情報工学科
コンピュータアーキテクチャ研究室

櫻田 賢大(411821)

# 内容梗概

近年，実行するアプリケーションの特性に適したスーパースカラコアを複数個組み合わせたヘテロジニアスマルチコアプロセッサの研究が注目されている．アプリケーションの特性に適したコアで処理することにより，現在広く用いられているホモジニアスマルチコアと比べ，計算性能の向上，省電力化が期待されている．しかし，スーパースカラコアの設計には膨大な時間がかかるため，現在スーパースカラコアの自動設計を行うツールセット FabScalar が提案されている．また，スーパースカラコアでは，レジスタファイルやレジスタリネーミングテーブル，分岐予測用の履歴テーブルなど，様々なマルチポートメモリが使用されている．マルチポートメモリの設計は非常に複雑なため，全てのマルチポートメモリを設計するには膨大な時間が必要となる．そこで，先行研究により，マルチポートメモリの自動設計ツールが提案・開発されている．しかし，現ツールのレイアウト設計における配置アルゴリズムは配線効率を考慮しておらず，消費電力や設計面で問題がある．本論文では，現ツールの配置アルゴリズムを改良し，両アルゴリズムによる設計結果を比較することで，改良アルゴリズムが有効であることを評価する．

# Abstract

Recently, several computer architecture trends suggest that now is the time to automate their development. These trends are the growing interest in heterogeneous multi-core processor which is higher performance and lower power than single generic core. but, in order to develop Superscalar processor, it needs several internal multi-port memories development. then, multi-port memory auto-design tool is suggested. this tool implemented no effective placing algorithm. This work is to improve the algorithm and to evaluate this algorithm.

# 目 次

# 図 目 次

# 表 目 次

# 1　はじめに

## 1.1　背景

近年，実行するアプリケーションの特性に適したスーパースカラコアを複数個組み合わせたヘテロジニアスマルチコアプロセッサの研究が行われている．アプリケーションの特性に適したコアで処理を行うことにより，現在広く用いられているホモジニアスマルチコアと比べ，性能向上，省電力化が期待されている．しかし，スーパースカラコアの設計には膨大な時間が必要となるため，現在スーパースカラコアの自動設計を行うツールセット FabScalar が提案されている．スーパースカラコアでは，レジスタファイルやレジスタリネーミングテーブル，分岐予測用の履歴テーブルなど，様々なマルチポートメモリが使用されている．FabScalar では，これらのマルチポートメモリを DFF で設計する．そこで，DFF による設計から SRAM による設計へ置き換えることで，低消費電力化及び回路規模の縮小が可能である．また，各々のメモリ構成はスーパースカラコアによって異なるため，ヘテロジニアスマルチコアプロセッサでは，使用するメモリの種類が増加する傾向がある．したがって，ヘテロジニアスマルチコアプロセッサでは設計するメモリの数が増加し，全種のメモリを設計するのに膨大な時間が必要となる問題がある．この問題を解決

するため，任意のポート数のマルチポートメモリを自動設計するツール RAMMAGE(RAM macro generater) が提案されている．しかし，現ツールのレイアウト生成時の配置アルゴリズムは配線効率を考慮していないため，冗長な配線により消費電力の増加に繋がる．

## 1.2 研究目的

スーパースカラコアを自動設計するツールセット FabScalar[1] と併用することにより，ヘテロジニアスマルチコアプロセッサの設計時間を短縮するマルチポートメモリ自動設計ツール RAMMAGE に対して，配線効率を考慮した配置アルゴリズムの実装，評価を行う．

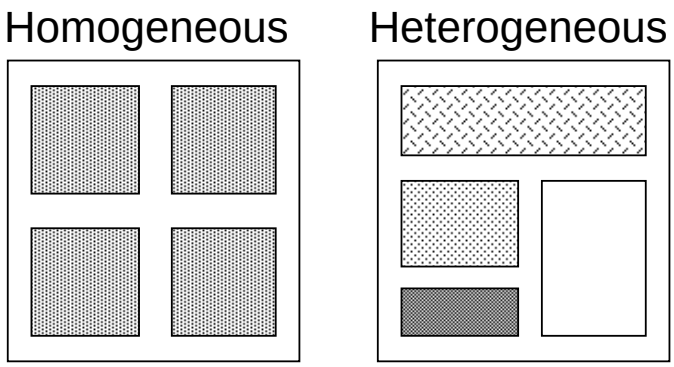


図 1.1: ホモジニアスマルチコアとヘテロジニアスマルチコア

# 2 関連研究

## 2.1 FabScalar

FabScalar はスーパースカラコアをフェッチ幅や演算器の数等のパラメータから自動設計できる設計ツールである．FabScalar が提供するスーパースカラコアのレジスタ転送レベル (RTL: register transfer level) のマルチポートメモリ部分は，スーパースカラコアの構成によって必要なポート数が異なり，エントリ数がパラメータ化されているものもある．しかし，一般的な FPGA や ASIC の設計フローにおいて利用されるメモリコンパイラは，3 ポート以上のメモリの設計ができない．そのため，ポート数とエントリ数の組み合わせが異なるメモリを全て手動で設計する必要があるので非常に時間がかかる．そこで，FabScalar が提供するスーパースカラコアのマルチポートメモリ部分のレイアウトの自動設計を行うツールである FabMem[2] が提案されている．

## 2.2 FabMem

FabMem はスーパースカラコアのマルチポートメモリをトランジスタレベルで設計し，レイアウトを自動生成するツールである．1-Read 1-Write のメモリセルに対して複数のワード線やビット線を多重化させることで

マルチポートメモリを生成している．トランジスタレベルで設計されたマルチポートメモリのレイアウトは設計するプロセス技術にトランジスタが依存するため，プロセス技術ごとに全てのトランジスタを再設計する必要があり移行に時間がかかる．1-Read 1-Write の 2 ポートメモリの多重化による設計の場合では，2 ポートメモリを移行するプロセス技術に対応しているメモリに変更するだけで移行出来るので汎用性に優れた設計が出来る．そのため，プロセス技術の移行が容易な 1-Read 1-Write の 2 ポートメモリの多重化によるマルチポートメモリの RTL の自動生成とレイアウトの自動設計を行う RAMMAGE が提案，開発されている．

## 2.3 RAMMAGE：RAM macro generator

RAMMAGE はスーパースカラコアのマルチポートメモリの RTL の自動生成と，レイアウトの自動生成を行う．図 2.2 は RAMMAGE を用いたマルチポートメモリの設計フローである．パラメータファイルより Read ポート数や Write ポート数等のパラメータを得た後，マルチポートメモリの RTL を自動生成する．生成されるマルチポートメモリの RTL は，一般的な FPGA や ASIC の設計フローにおいて利用されるメモリコンパイラで生成可能な 1-Read 1-Write メモリの多重化によるマルチポートメモリの定義がされている．次に，生成された RTL を論理合成することで生成

されるネットリストを用いて，生成するマルチポートメモリを構成している 1-Read 1-Write メモリの自動配置を行うことで LSI の設計に必要なレイアウト設計を実現している．設計者は設計されたレイアウトをハードウェアマクロとして利用することが出来るため，各メモリのレイアウト設計を行う時間を大幅に短縮することが可能となる．

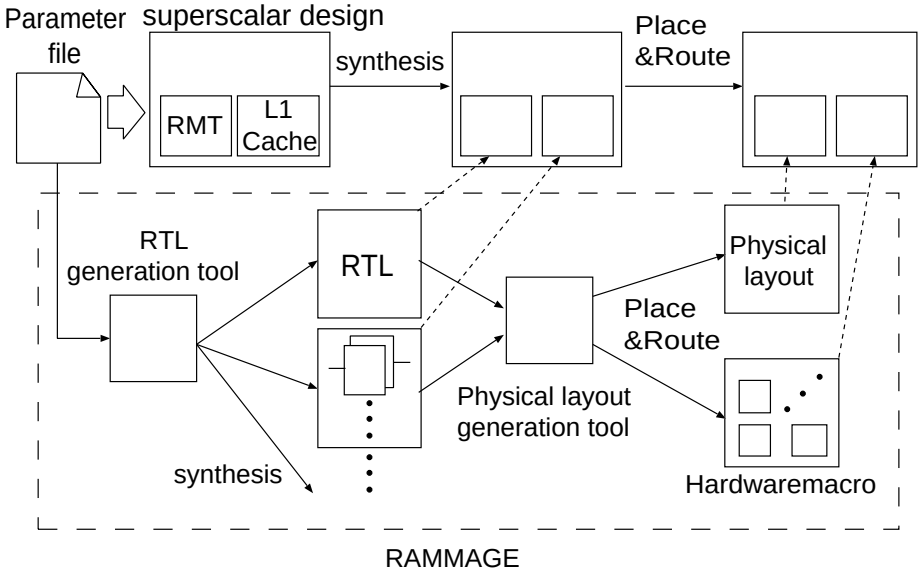


図 2.2: RAMMAGE の設計フロー

## 2.4 一般的な配置配線ツールのマクロの自動配置アルゴリズム

一般的な FPGA や ASIC の設計フローで用いられる，チップ上へ配置されるセルはスタンダードセルとマクロセルがあり，それぞれで自動配置アルゴリズムの研究が行われている．本研究で提案するアルゴリズム

は，マクロセル (メモリマクロ) の自動配置アルゴリズムである．既存研究 [3] で提案されているアルゴリズムでは，異なる構成のマクロセルを 1 つのチップ上に自動配置するためのものである．既存研究によるアルゴリズムは，論理合成後のネットリストを読み込み，マクロの面積やピン数に応じて重み付けを行う．重み付けの結果から，優先度を設け，乱数によるランダム配置を各マクロに対して行う．しかし，本研究で求められるアルゴリズムは構成の同じマクロを１つのチップ上に自動配置するものである．したがって，本研究において，このような既存研究でのアルゴリズムを用いた場合，重み付けの結果が全てのマクロで同等となり，ランダム配置のみが行われる結果となる問題がある．そこで，同一のメモリマクロを複数利用するマルチポートメモリの設計方法に合わせた配置アルゴリズムが求められる．

# 3　マルチポートメモリの設計方法

マルチポートメモリの設計方法には 2 種類あり，それぞれの設計方法の利点及び欠点を示し，自動設計ツール RAMMAGE に採用されている方法について詳しく述べる．

## 3.1　通常のマルチポートメモリの設計

一般的な SRAM の設計と同様にトランジスタによる設計 [4] を指す．典型的な SRAM では 1 ビットを格納するのに 6 個の MOSFET（6T）を使用する．8T や 10T で記憶セルを構成するものでは，読み書きポートを複数実装するために使われ，マルチポート型 SRAM 回路の設計に用いられる．この設計方法での利点，欠点を以下に示す．

- 利点

  トランジスタは小型化高速化が進んでおり，設計される SRAM が小型で，低消費電力である．

- 欠点

  トランジスタ設計で必要となるプロセス技術に変更が行われると，プロセス技術に合わせたトランジスタが新たに設計され，SRAM 内のトランジスタの移行のために膨大な時間が必要となる．また，こ

の設計方法によるマルチポート SRAM は，一般的な FPGA/ASIC の設計フローで利用されるメモリコンパイラで生成ができない．

## 3.2 メモリマクロ多重化によるマルチポートメモリの設計

トランジスタによる設計が行われた複数の SRAM を組み合わせて設計する方法である．組み合わせる際には，配線やスタンダードセルを用いる．この設計方法の利点，欠点を以下に示す．

- 利点

  プロセス技術に変更が行われた場合には，多重化に用いている SRAM をプロセス技術に合わせて設計し，置き換えることで容易にプロセス技術に対応した設計が可能である．そのため，移行に必要な時間の短縮となる．また，一般的な FPGA/ASIC の設計フローで利用されるメモリコンパイラで生成可能な SRAM の多重化により，一般的な設計フローを辿る事が可能となる．

- 欠点

  トランジスタレベルで設計される SRAM に比べ，大型化し易く，消費電力の増加が予想される．

一般的なFPGA/ASICの設計フローで利用できるマルチポートSRAMの設計のため，RAMMAGEは，こちらの設計方法を用いている．

## 3.3 メモリマクロ多重化による設計の詳細

多重化を行うメモリマクロは，2ポートの1-Read 1-Writeメモリである．この2ポートSRAMに対し，段階的に多重化を行うことで，目標の構成のマルチポートSRAMを設計する．多重化の段階は，書込みポートの多重化，読み込みポートの多重化，目標のメモリ容量を満たすための多重化の3段階である．各段階について述べる．

- 書込みポートの多重化

  書き込みポートの多重化の例となるブロック図を図3.3で示す．この多重化処理では，各アドレスの最新の情報を持つメモリを識別するためのハードウェアが追加される．この追加されるハードウェアは，DFFにより実装される小規模なマルチポートメモリである．

- 読み込みポートの多重化

  読み込みポートの多重化の例となるブロック図を図3.4で示す．この多重化処理では，書込みポートへの入力信号は，利用する2ポートメモリで共有するように繋ぎ，読み込みポートはそれぞれで異な

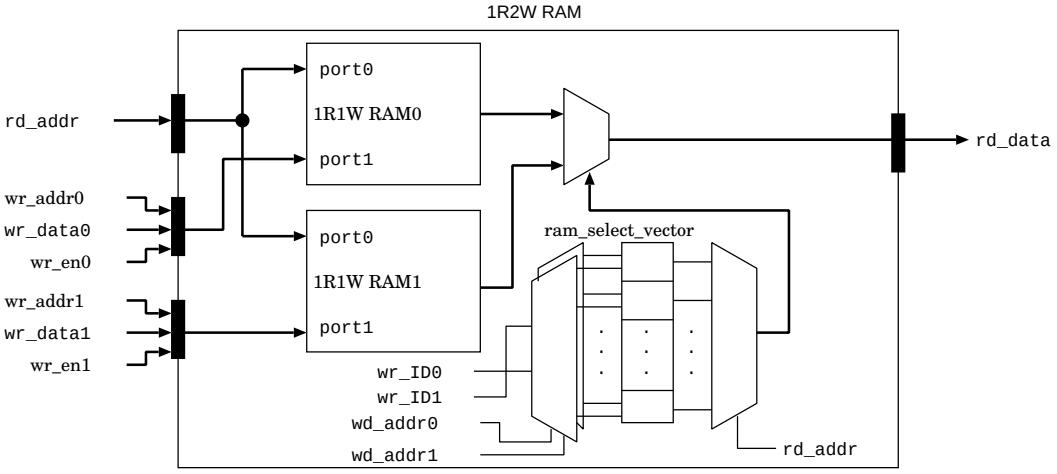


図 3.3: 書込みポートの多重化

るよう繋ぐことで多重化を行う.

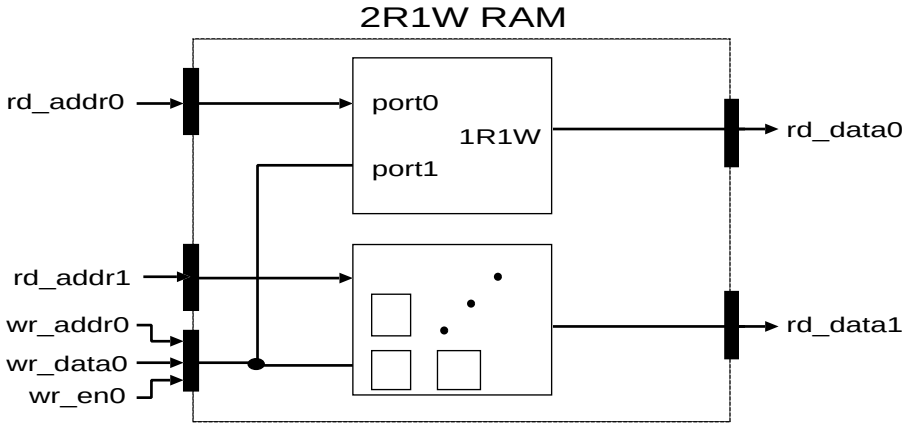


図 3.4: 読み込みポートの多重化

- 目標のメモリ容量を満たすための多重化

  この多重化の例となるブロック図を図 3.5 で示す．メモリの容量はワード数とビット数の積であり，それぞれで必要容量を満たすため

の数量の 2 ポートメモリが必要となる．そのため，図 3.5 では，256 ワード-16 ビットの 2 ポートメモリの設計のため，128 ワード-8 ビットの 2 ポートメモリを 4 つ利用している．設計上のビット幅の整列化制約を回避するため，余剰の上位ビットには 0 詰めを行う設計である．たとえば，9 ビット幅のメモリの設計では，8 ビット幅のメモリを 2 つ利用し，16 ビット幅のメモリを設計する．そして，余剰の上位 7 ビットは 0 詰めが行われる．したがって，整列化制約が無視できる設計が行われる．

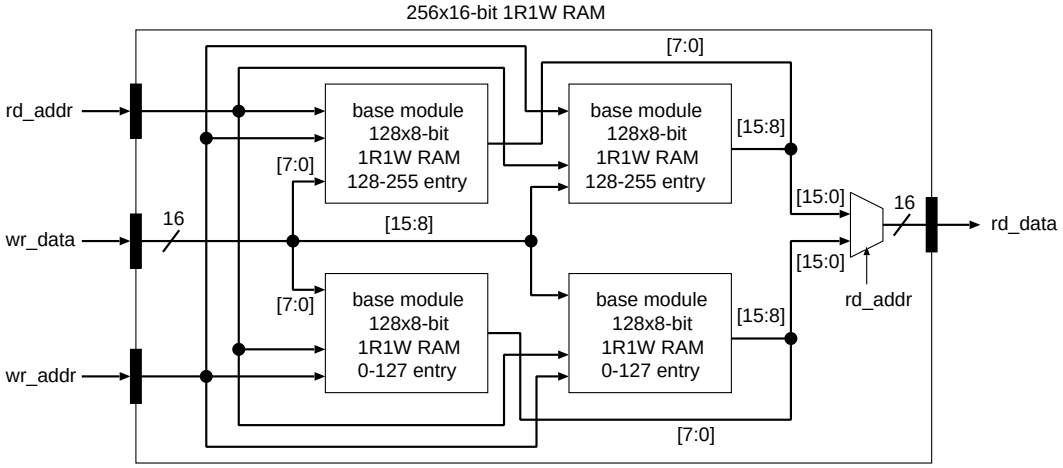


図 3.5: メモリ容量拡張用の多重化

上記の多重化を行うことで必要となる 2 ポート SRAM は，書込み・読み込みポート，メモリ容量拡張のための必要数の 3 つの積で求められる．スーパースカラコアで利用されるマルチポートメモリの例として，書込みポー

ト 4，読み込みポート 4，記憶容量 256x8bit が挙げられる．このマルチポート SRAM を記憶容量 64x8bit の 2 ポート SRAM の多重化により設計すると，64 個の 2 ポート SRAM が必要になる．このように多重化による設計では，2 ポート SRAM が多数利用され，レイアウト設計の工程で，これらを一つのチップ上に配置しなければならない．したがって，機械的な処理により高速で並べるため，自動配置アルゴリズムが求められる．

# 4　提案手法

2ポートメモリの多重化による設計であるため，ネットリストは規則的な階層構造である．提案手法では，この構造を利用し，各メモリの関係性を考慮した配置メモリマクロのリストを作成する．次いで，Algorithm1に沿って，配置する行数・列数を求め，配置を行う．配置の際は，リストの順に求めた行数まで縦方向に配置する．最終行への配置後，次の列へ移動し，先程とは逆の縦方向に配置を行う．以降は全てのメモリマクロを配置するまで再帰的に処理する．また，配線が複雑化する箇所を限定し，消費電力の低減・アクセス時間の短縮のため，メモリマクロの端子の方向は配置される行数に応じて決定する．

# 5　実装方法

Tclを用いて，提案する自動配置アルゴリズムを実現し，既存の配置配線ツールに組み込むことで実装した．自動配置を行うためのパラメータは，レイアウトの最大幅のみで，配線領域などの定義は組み込まれたアルゴリズム上で自動的に行われる．また，設計されるレイアウトが設計規則を満たさない場合には，配線領域を拡張し自動で再設計を行う．

**Algorithm 1** Proposed

```
list ⇐ Get Memory List
Column ⇐ integer(sqrt(list.length))
Row ⇐ list.length/Column
while until Row is Integer do
  Column ⇐ Column − 1
  Row ⇐ list.length/Column
end while
Initialize c and dir
while list is unclear do
  for r ⇐ 1 to Row do
    place(r, c, dir), upside down dir
  end for
  upside down dir, c ⇐ c + 1
  for r ⇐ Row to 1 do
    place(r, c, dir), upside down dir
  end for
  c ⇐ c + 1
end while
```

# 6　性能評価

Rohm 0.18um CMOS プロセス上で，従来及び提案アルゴリズムで生成されたそれぞれのレイアウトで面積，消費電力，アクセス時間の評価を取り，提案アルゴリズムの有効性を検証した．表 6.1，6.2 は，それぞれ 64x8-bit メモリの多重化で設計された 2R2W の 64x32-bit と 1R2W の 64x32-bit のマルチポートメモリの評価である．評価結果から，面積，消費電力及びアクセス時間の全項目で約 10%の改善が確認できる．

表 6.1: 64x32-bit 2R2W メモリでの評価結果

| アルゴリズム | 従来 | 提案 |
|---|---|---|
| 面積 ( μ $m\hat{2}$) | 1.47 | 1.36 |
| 消費電力 ( μ W) | 0.86 | 0.71 |
| アクセス時間 (ns) | 16.95 | 15.97 |

表 6.2: 64x32-bit 1R2W メモリでの評価結果

| アルゴリズム | 従来 | 提案 |
|---|---|---|
| 面積 ( μ $m\hat{2}$) | 0.74 | 0.65 |
| 消費電力 ( μ W) | 0.42 | 0.35 |
| アクセス時間 (ns) | 16.92 | 15.48 |

# 7　おわりに

本研究では，擬似マルチポートメモリ自動設計ツール RAMMAGE のレイアウト設計におけるアルゴリズムの改良を行った．この改良により，設計規則を満たすマルチポートメモリの自動設計が可能となった．今後は，スーパースカラコアに実装し，実用性を検証する必要がある．

# 謝辞

本研究の機会を与えて頂いた 教授 近藤利夫 先生，並びにご指導，ご助言頂いた 助教授 佐々木敬泰 先生，深沢研究員の皆様にも心より感謝いたします.

# 参考文献

[1] N. K. Choudhary，et. al. ”FabScalar: Composing Synthesizable RTL Designs of Arbitrary Cores within a Canonical Superscalar Template ”. ISCA-38, pp. 11-22, June 2011.

[2] T. A. Shah. FabMem: A Multiported RAM and CAM Compiler for Superscalar Design Space Exploration. M.S. Thesis, Department of Electrical and Computer Engineering, North Carolina State University, May 2010.

[3] Akiko Yamada, et. al. ”A New Hieralchical AlgoriIthm of Transistor Placement for CMOS Macro Cell Design ”. 電子情報通信学会総合大会講演論文集 1995 年. 基礎・境界, 134, 1995-03-27

[4] 宮地 幸祐 他, ” 高速，低消費電力 6T-SRAM を実現する電荷の同時注入による不良セルの修復技術”, IEICE Technical ReportICD2012 - 12, pp. 1-2, April 2012