学位請求論文

速度 電力最適化 CMOS 集積回路の 高密度レイアウト手法に関する研究

１９９３年１２月２０日提出

指導教官: 浅田 邦博 助教授

東京大学 大学院 工学系研究科
電子工学専攻
張 洪明
学籍番号: 17105

目次

図一覧

表一覧

第 1 章　序論

マイクロ プロセッサおよびメモリ (DRAM) が誕生して以来、集積回路の集積度は年平均 50% 越えるスピードで向上し、ついに３００万トランジスタを集積した Super Sparc が発表された [1]。２０００年ごろには 4G ビット DRAM や 5 千万素子を集積した 2000 MIPS 性能のマイクロ プロセッサの出現が見込まれている。このように高集積化、高性能化が進む集積回路を支援するレイアウト自動設計には新たな課題が生じている。

1.1　集積回路レイアウト自動設計の現状と本研究の背景

サブミクロン レベルの微細化技術によって、デバイス素子の遅延が数十ピコ セコンドに達し、信号遅延の支配的な要因はデバイス素子内部の遅延から素子間配線の遅延に変りつつある。このため、信号遅延を考慮した大規模集積回路のレイアウト自動設計が必要になってきた。信号遅延を考慮したレイアウトでは、与えられたタイミング制約によってセルを配置するというパフォーマンス ドリブン レイアウト (Performance Driven Layout) が知られている [2, 3, 4]。これらの手法は、タイミング制約を守ることができるが、信号遅延が最小となるレイアウトを発見することができない。タイミング要求は本来、与えられるものではなく、回路に固有なものである。よって、本研究では、クリティ

カル パスの探索と遅延時間の解析によってタイミング要求を求め、求めたタイミング要求にしてがって、信号遅延が最小となるようにレイアウトする方法について研究した。

一方、集積回路の大規模化、高速化、特に、RISC アーキテクチャの導入により、回路の動作周波数が増加するため、消費電力は大きな問題となっている。低消費電力化に関する研究はすでに各設計、製造部門で繰り広げられている。消費電力は回路の物理的な構造に関係する部分が多いので、消費電力の削減を考慮したレイアウト自動設計がこれから注目される新しい課題になるであろう。このような背景で、本研究では、論理シミュレーションによって各ノードの充放電を調べ、頻繁に充放電される節点につながっているセルを近くに配置するという消費電力削減手法について研究した。

集積回路の設計自体はそれほど難しい問題ではないが、最適な集積回路の設計は非常に難しい問題である。このため、集積回路の設計では、いくつかの部分設計に分けて、それぞれの部分設計において最適化を行なう。代表的な部分設計は、システム設計 (System Level Design) あるいはアーキテクチャ設計 (Architectural Level Design) [5, 6, 7]、機能設計 (Register-Transfer Level Design) [8, 9, 10, 11, 12, 5]、論理設計 (Logic Level Design) およびレイアウト設計 (Physical Level Design) が知られている。このような分け方にしたがって、これまでのレイアウト設計の目標は、ほとんど面積と計算時間を減らすこととされている [13]。しかし、信号遅延と消費電力は回路の物理的な構造と密接に関連している部分が多く、レイアウト設計の段階でそれを考慮しないと、最適な集積回路が設計できるとは言えない。

信号遅延を考慮したレイアウトは最近注目され始め、研究されている。このようなレイアウト手法は、遅延時間が最小となるレイアウトを発見することができる。このときの最小遅延時間はデバイス テクノロジー、回路の構造と接続などによって決められる。よって、遅延時間をさらに短縮するためには、回路の構造、接続などを変える方法がある。これはよく知られていることであるが、これを取り入れることのできる設計システムは限られている。特に、現在

主流となっているセル ライブラリ設計方式の設計システムでは、論理回路の等価変換を行おうとしても、セル ライブラリに登録したものに限って変換が可能である。つまり、その探索空間はライブラリに限られている。ライブラリの制約をなくし、探索空間を広げるため、本研究では、任意な論理回路に対しても、高速でコンパクトなレイアウトが自動設計できるモジュール ジェネレータを製作した。

1.2 本研究の概要と本論文の構成

本論文では、面積、速度、電力の3つをキーワードとして、レイアウト自動設計と最適化手法について議論する。

第2章では、レイアウト面積、信号遅延、消費電力などを削減することができるソース - ドレーン共有型レイアウトについて検討する。CMOS 集積回路のセルレイアウトの大部分は拡散層、金属配線およびコンタクト窓開けなどのセル内配線で占められている。従って、高密度集積回路のセル レイアウト設計においては、主にセル内配線の面積を考慮してコンパクトなトランジスタの配置を発見することが重要である。セル内配線を減らす効果的な手法として接続関係のあるソース - ドレーンをつなぎ合わせることが知られている。このようなレイアウトスタイルのことをソース - ドレーン共有型レイアウトと呼ぶことにする。ソース - ドレーンの共有によって、拡散領域の面積が削減されたので、拡散容量、そして、信号遅延と消費電力を減らす効果もある。このため、本研究では、ソース - ドレーン共有型レイアウトを2次元に拡張する可能性について検討した。この章では、1つ拡散領域は、最大2つトランジスタに共有されるレイアウトのことを1次元レイアウト、2つ以上トランジスタに共有されるレイアウトのことを2次元レイアウトと言うことにする。検討した結果では、2次元レイアウトは、1次元レイアウトより、最大 20% 平均 10% 程度面積が減少することが可能である。しかし、2次元レイアウトには、トランジスタのゲートは曲がることがあるなどの問題が存在するため、本研究では、1次

元レイアウトのスタイルを採用した。

第3章では、処理順序に依存しない一括処理型トランジスタ ペアリング アルゴリズムについて説明する。CMOS トランジスタ回路を1次元にレイアウトするとき、縦に配置するPタイプとNタイプ トランジスタをトランジスタペアという。同じゲート信号を持つトランジスタが2つ以上存在するとき、トランジスタ ペアの作成処理が必要となる。従来のトランジスタ ペアリング手法には、デュアルなCMOSトランジスタ回路に限定される、PタイプとNタイプのトランジスタのそれぞれのネットワークにおける接続関係を考えていない、ペアリングの結果は処理順序に依存するなどの問題点が存在する。これに対して、本研究では、すべての可能なトランジスタ ペアについて調べ、それぞれのペアの近傍の接続関係を評価して、評価の結果に基づいて、すべてのペアを一括に決定するという一括処理型トランジスタ ペアリング手法を提案した。これにより本手法が非デュアルな CMOS トランジスタ回路を含めて任意な CMOS トランジスタ回路に適用できるという特長がある。つまり、すべての CMOS トランジスタ回路に対して、同じ処理でトランジスタのペアを作成することができる。また、本手法で見つかったトランジスタのペアが必ずしもデュアルな関係を持たず、かつデュアルな関係で作ったペアより最適な場合があることが実験から示された。

第4章では、与えられた CMOS トランジスタ回路をカバーする最小のパス数が求められるコンパチブル ペア探索アルゴリズムについて説明する。CMOS 回路のレイアウト面積の最小化あるいは拡散領域の最小化手法は、トランジスタのゲートを枝、ソース - ドレーンの接続を節点とした2つグラフ（Pグラフと N グラフ）をカバーする枝の順序が同じとなる最小のパス数を見つけることによって実現する。このようなパス探索問題について最初のアルゴリズムは、Uehara らによって提案されたものであり、入力数が偶数であるゲートに仮の (Pseudo) 入力を加えて、実の入力と仮の入力の絡み合うことが最小となるように、入力の順序を変更するというヒューリスティックアルゴリズムである。一方、Maziasz らは、デュアルな直並列 CMOS 回路の接続関係をコンポジショ

ン ツリー (Composition Tree) で表現して、ツリーのリーフからボトム アップ的に処理し、ルートに到達する時、その回路をカバーする最小数のパスが求められるという改良したアルゴリズムを提案した。その他に、直並列型ダイナミック CMOS 回路のパス探索アルゴリズムは Lengauer らによって提案されている。以上のアルゴリズムには直並列回路に限定されるという共通の問題点がある。非直並列 CMOS 回路に関する研究は Ong ら、 Wimer らと Hwang らのアプローチがある。これらのパス探索処理では、トランジスタのペアを 1 つずつ取ることによって作る。計算量を抑えるため、あるコスト関数に基づき、よい結果が得られないと思われるパスを削除する。しかし、このようなアルゴリズムには処理の結果はそのコスト関数および処理の順序に依存するという問題がある。これらの問題を克服するため、本研究では、まず、コンパチブル ペアの概念を導入した。2 つトランジスタ ペアにある P タイプとも N タイプ トランジスタともつなぎ合わせることができるとき、この 2 つトランジスタ ペアはコンパチブルであると呼ばれる。次に、トランジスタの接続関係をコンパチブル マトリクスで表現する。1 つトランジスタ ペアと他のすべてのトランジスタ ペアとのコンパチブル関係を表すものはコンパチブル マトリクスである。これによって、非直並列回路を含めて、任意の CMOS 回路を表すことができるようになった。このコンパチブル マトリクスには、探索する必要のないトランジスタ ペアに関する情報も含まれているので、無駄な探索を避けることができる。すべての探索はコンパチブル マトリクスにおいて行なわれるため、網羅的な探索を行っても、 Maziasz らのアルゴリズムより数倍から数百倍速いという結果が実験から示された。

第 5 章では、一括処理型トランジスタ ペアリング アルゴリズムとコンパチブル ペア探索アルゴリズムを取り入れた CMOS トランジスタ回路モジュール ジェネレータ **GmC** を評価した結果について述べる。モジュール ジェネレータ **GmC** は遅延時間が小さい、消費電力が小さい回路を探すために作られたものである。いろいろな回路に対する実験を通じて、**GmC** を評価した結果は次の通りである。

1. 非直並列、非デュアルな回路を含めて、任意な CMOS トランジスタ回路に対しても適用できる。

2. 与えられた CMOS トランジスタ回路にデュアルなオイラー パスが存在すれば、必ずそれを見つけることができる。

3. 典型的な CMOS トランジスタ回路について、GmC によって見つかったパスの数は理論上の最小値に達した。

4. 従来の手法より数倍から数百倍の速さで最適解を見つけることができる。

また、実験の結果から次の結論が得られた。与えられた CMOS トランジスタ回路をカバーする最小のパス数を求める問題において、最適なトランジスタ ペアが見つからなければ、網羅的なパス探索アルゴリズムがあっても、その回路をカバーする最小のパス数を見つけることができない。

第6章では、モジュール ジェネレータ GmC を基本道具として、集積回路の信号遅延の最小化手法、および信号遅延を考慮したレイアウト設計手法について述べる。信号遅延の最小化手法として、モジュール ジェネレータ GmC を用いた論理回路の再合成手法と論理回路の物理的な構造を考慮し、論理的に等価である接続を入れ換える手法がある。信号遅延を考慮したレイアウト自動設計では、クリティカル パスの探索と遅延時間の解析によってタイミング要求を求め、求めたタイミング要求にしてがって、信号遅延が最小となるようにレイアウトを行なう。

第7章では、モジュール ジェネレータ GmC を基本道具として、消費電力の削減手法と消費電力を考慮したレイアウト自動設計手法について述べる。消費電力の削減手法として、クリティカル パス以外の回路に対して、モジュール ジェネレータ GmC を用いた論理回路の再合成手法と論理回路の物理的な構造を考慮し、論理的等価である接続を入れ換える手法がある。消費電力を考慮したレイアウト自動設計手法では、論理シミュレーションによって各ノードの充放電を調べ、頻繁に充放電される節点につながっているセルを近くに配置し、配線容量、そして、消費電力が最小となるようにレイアウトを行なう。

参考文献

[1] Abu-Nofal et al. "A Three-Million-Transistor Microprocessor". *The 1992 IEEE International Solid-State Circuits Conference (ISSCC), Digest of Technical Papers*, pages 108–109, 1992.

[2] K. Nakao, O. Kitada, M. Hayashikoshi, K. Okazaki, and Y. Tsujihashi. "A High Density Datapath Layout Generation Method Under Path Delay Constraints". *Proceedings of The IEEE 1993 Custom Integrated Circuits Conference (CICC)*, pages 9.5.1–9.5.5, May 1993.

[3] T. Gao, P. M. Vaidya, and C. L. Liu. "A New Performance Driven Placement Algorithm". *IEEE International Conference on Computer-Aided Design, ICCAD-91, Digets of Technical Papers*, pages 44–47, November 1991.

[4] M. A. B. Jackson and E. S. Kuh. "Performance Driven Placement of Cell Based ICs". *Proceedings of the 26th ACM/IEEE Design Automation Conference (DAC)*, pages 370–375, June 1989.

[5] D. E. Thomas et al. *"Algorithmic and Register-Transfer Level Synthesis: The Syetem Architect's Workbench"*. Kluwer Academic Publishers., 1990.

[6] M.C.McFarland and S.J. "Using Bottom-Up Design Techniques in the Synthesis of Digital Hardware from Abstract Behavioral Descriptions".

Proceedings of the 23rd ACM/IEEE Design Automation Conference (DAC), pages 474–480, June 1986.

[7] Y. Nakamura. "An Integrated Logic Design Environment Based on Behavioral Description". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-6(3):1115–1128, May 1987.

[8] S.Davidson, D.Landskov, and P.W.Mallett. "Some Experiments in Local Microcode Comaction for Horizontal Machines.". *IEEE Transactions on Computers*, C-30(7):460–477, July 1981.

[9] E.F.Girczyc and J.P.Knight. "An ADA to Standard Cell Hardware Compiler Based on Graph Grammars and Scheduling.". *Proceedings of the IEEE International Conference on Computer Design (ICCD)*, pages 726–731, October 1984.

[10] A.C.Parker, J.Pizarro, and M.Minar. "MAHA: A Program for Datapath Synthesis". *Proceedings of the 23rd ACM/IEEE Design Automation Conference (DAC)*, pages 461–466, June 1986.

[11] C.Tseng and D.P.Siewiorek. "Automated Synthesis of Data Paths in Digital Systems". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-5(3):379–395, July 1986.

[12] Wakabayashi K. and Yoshimura T. ""A Resource Sharing and Control Synthesis Method for Conditional Branches". *IEEE International Conference on Computer-Aided Design, ICCAD-89, Digets of Technical Papers*, pages 62–65, November 1989.

[13] E.S. Kuh and T. Ohtsuki. "Recent advances in VLSI layout". *Proceedings of the IEEE*, 78(2):237–259, February 1990.

第 2 章　ソース - ドレーン共有型レイアウト スタイルの検討

CMOS 集積回路のセルレイアウトの大部分は拡散層、金属配線およびコンタクト窓開けなどのセル内配線で占められている。従って、セルのレイアウト設計においては、主にセル内配線の面積を考慮してコンパクトなトランジスタの配置を発見することが重要である。セル内配線を減らす効果的な手法として、接続関係のあるソース - ドレーンをつなぎ合わせる方法が知られている。本論文では、このようなレイアウト スタイルのことを**ソース - ドレーン共有型レイアウト**と呼ぶことにする。ソース - ドレーン共有型レイアウトの最大目的は拡散領域の面積を最小にすることである。これによって、信号遅延と消費電力の 1 つ要因 — 拡散容量も最小となる。ソース - ドレーン共有型レイアウトの従来のスタイルでは、1 つ拡散領域は最大 2 つトランジスタに共有される。このようなレイアウト スタイルを 1 次元レイアウトと言うことにする。これに対して、1 つ拡散領域は 2 つ以上のトランジスタに共有されるスタイルを 2 次元レイアウトと言うことにする。この章では、ソース - ドレーン共有型レイアウトを 2 次元に拡張する可能性について検討する。

2.1 CMOS トランジスタ回路のレイアウト手法

CMOS トランジスタ回路をレイアウトに展開する手法として、ゲート マトリクス レイアウト [1] が知られている。図2.1はゲート マトリクス レイアウトの1例である。このレイアウトの上半分はPタイプ トランジスタであり、下半分はNタイプ トランジスタである。同じ入力信号のトランジスタを1列に並べ、1本の縦のポリシリコン線でつなぐ。その以外の接続はメタル線あるいは拡散層で配線する。このようなレイアウトには、1列に配置するトランジスタの数は一致しないと、空き領域が現れてしまうという問題がある。このような空き領域を減らすため、ゲート マトリクス レイアウトで得られたトランジスタの配置情報を基にして、トランジスタを一定幅の領域に詰め込んだ後、3次元迷路法によって配線を行なうレイアウト手法が提案されている [3]。

ゲート マトリクス レイアウトでは、トランジスタのゲートの接続を優先的に決めるため、ソース - ドレーン間の接続はメタル線あるいは拡散層を利用する場合が多い。これは速度と電力に大きな影響を与える。トランジスタのゲートの接続より、拡散領域の削減を優先的に考えるレイアウト手法がソース - ドレーン共有型レイアウトである。このレイアウト手法では、与えられたCMOSトランジスタ回路をカバーする最小のパス数を見つけることによって拡散領域の最小化を実現する。つまり、ソース - ドレーン共有型レイアウトの最大目的は拡散領域の最小化にある。したがって、本研究ではソース - ドレーン共有型レイアウトについて考える。

2.2 ソース - ドレーン共有型レイアウトの特徴

ソース - ドレーン共有型レイアウトは次の特徴を持っている。

1. レイアウト面積、特に、拡散領域の面積は小さい。

2. 配線数、そして、拡散領域上でのコンタクト窓開けの数は少ない。

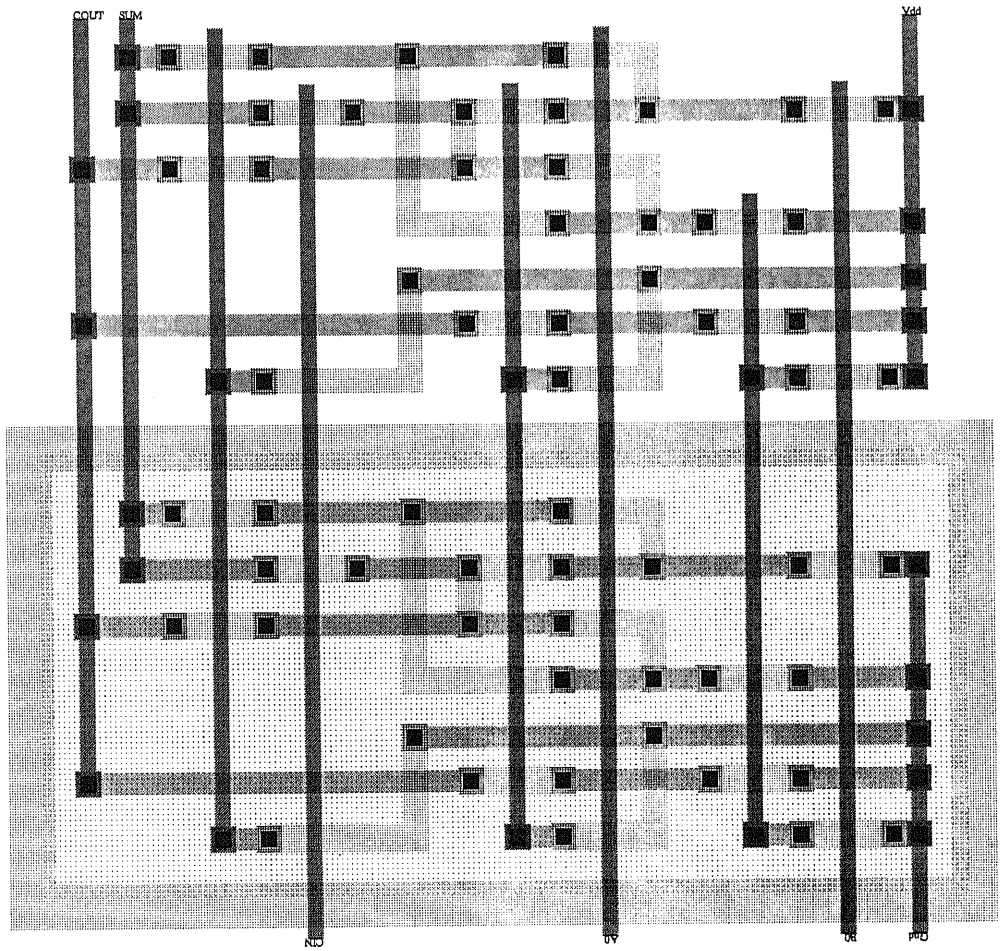


図 2.1: ゲート マトリクス レイアウトの例

拡散領域が小さければ小さいほど寄生容量が小さくなる。コンタクト窓開けの数は少なければ少ないほど信頼性が高くなる。これらの特徴は、最も簡単な論理回路 — 2 入力 NAND ゲートにも見られる。

例えば、図 2.2-(a) の 2 入力 NAND ゲートを CMOS トランジスタで作る場合、図 2.2-(b) のようになる。同じ入力信号を持つトランジスタを 1 列に配置すれば、P タイプ トランジスタと N タイプ トランジスタのゲートは図 2.2-(c) に示すように簡単に接続される。この図では、P タイプ トランジスタは上の行に、N タイプ トランジスタは下の行に配置されている。この配置では、入力信号は b である P タイプ トランジスタのドレーンと入力信号は a である P タイプ トランジスタのソースとは隣の位置にあるが、ノードが違うため、両者の間に設計ルールに決められたセパレーション（拡散領域のギャップ）を挿入しなければならない。ところが、入力信号は a である P タイプ トランジスタをひっくり返して配置すれば、そのセパレーションをなくすことができる（同図 (d)）。このときのレイアウト面積はおよそ 20% 節約される。したがって、トランジスタのソース - ドレーンの拡散領域をつなぎ合わせることによって、面積の小さいレイアウトが得られる。また、ソース - ドレーン共有のないレイアウト（図 2.2-(c)）と比べて、共有のある方は、配線が 2 本、コンタクト窓開けが 3 個、少なくなる。

ソース - ドレーン拡散領域の共有によって、拡散容量も小さくなる。これを示すものは図 2.3のシミュレーション結果である。このシミュレーション結果から、拡散領域が共有された場合（実線）の立ち上がり時間と立ち下がり時間とも速くなったことがわかる。このシミュレーションでは配線容量は考慮していないので、速くなった分は拡散容量が減った分となる。もし、1 個のコンタクトに 1 個分ドレーン - 基板間の容量 C_{BD} が相当するならば、出力ノード z に付いている C_{BD} は、図 2.2-(c) と比べて、図 2.2-(d) の方が 1 つ少ないことになる。

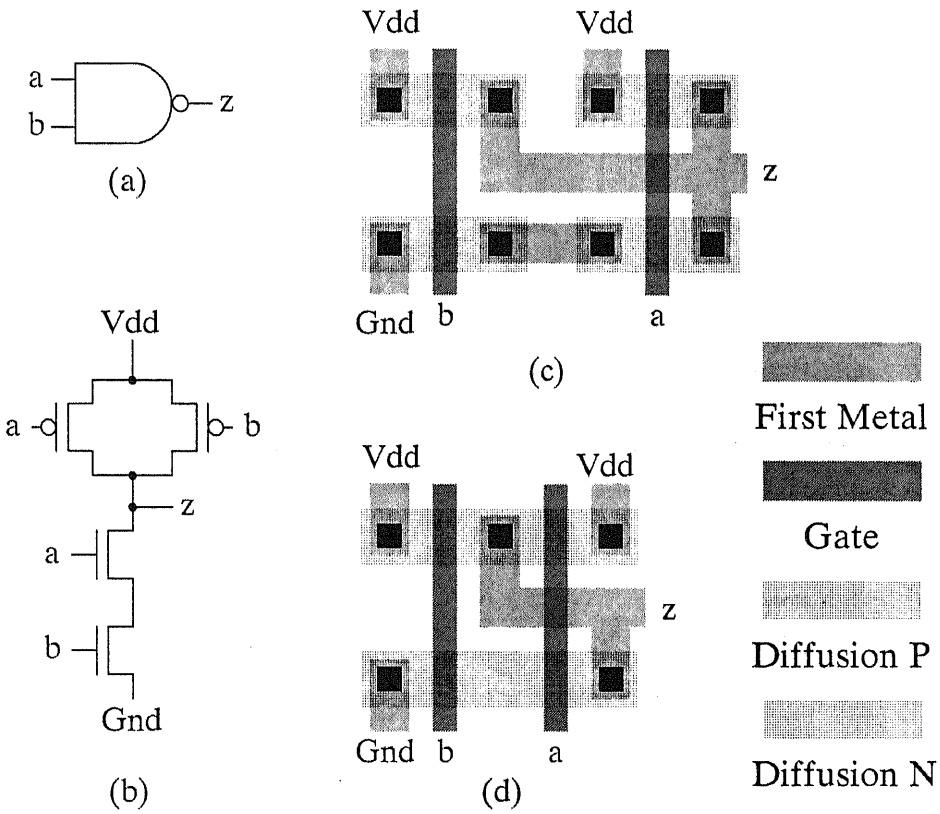


図 2.2: 拡散領域の共有によるコンパクトなレイアウト

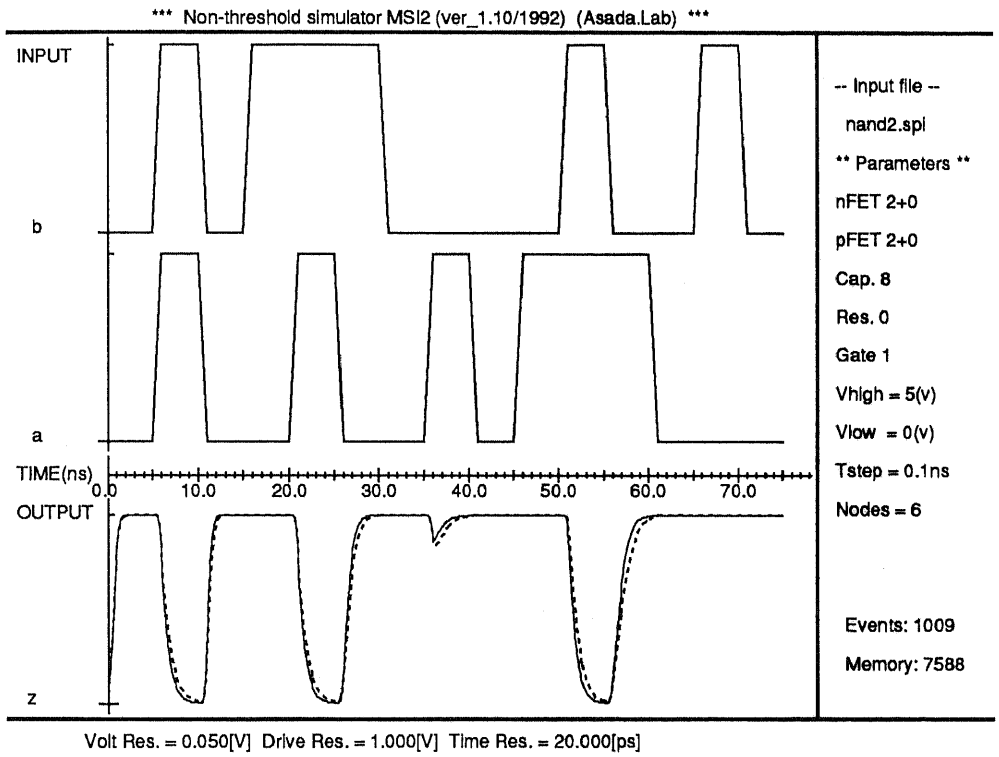


図 2.3: 拡散領域の共有による遅延時間の改善

点線：図 2.2-(c) のレイアウトに対応

実線：図 2.2-(d) のレイアウトに対応

2.3 ソース - ドレーン共有型レイアウトの 2 次元拡張: 2 次元 FET セル

図 2.2の例のように、1 つソース - ドレーンの拡散領域が最大 2 つトランジスタに共有されるレイアウトのことをトランジスタ ネットワークの **1 次元展開**あるいは **1 次元レイアウト**ということにする。これに対して、1 つソース - ドレーンの拡散領域が 2 つ以上のトランジスタに共有されるレイアウトのことをトランジスタ ネットワークの **2 次元展開**あるいは **2 次元レイアウト**ということにする。1 次元レイアウトと比べて、2 次元レイアウトの配線数およびコンタクト数はさらに減少することと期待できるので、この節では、トランジスタのネットワークを 2 次元に拡張するための基本セルの設計を示す。

まず、3 つトランジスタのソース - ドレーンがつながっている例 2.4-(a) について考える。この 3 つトランジスタからなるネットワークを 1 次元展開すると、そのレイアウトは図 2.4-(b) に示すようになる。ここで、図 2.4-(a) はネットワークの 1 部分を表しており、図 2.4-(b) は未配線であり、等価レイアウトではないが、面積は図 2.4-(a) と等価であると考えられる。図 2.4-(a) の 3 つトランジスタが 1 つ共通なソース - ドレーンの拡散領域を共有するとき、考えられるレイアウト パターンを図 2.4-(c) と (d) に示す。図 2.4-(c) と (d) のレイアウトパターンを **2 次元 3FET セル**と呼ぶことにする。

これらのセルのレイアウト面積については、図 2.4-(b) の面積を 1 としたとき、1 つの標準的な設計ルールに基づくと、図 2.4-(c) と (d) の面積はそれぞれ 93%,99% となる。

つぎに、4 つトランジスタのソース - ドレーンがつながっている例図 2.5-(a) について考える。この 4 つトランジスタからなるネットワークを 1 次元展開すると、そのレイアウトは図 2.5-(b) に示すようになる。ここで、図 2.4と同じように、図 2.5-(a) はネットワークの 1 部分を表しており、図 2.5-(b) は未配線であり、等価レイアウトではないが、面積は図 2.5-(a) と等価であると考えられる。図 2.5-(a) の 4 つトランジスタが 1 つ共通なソース - ドレーンの拡散領域

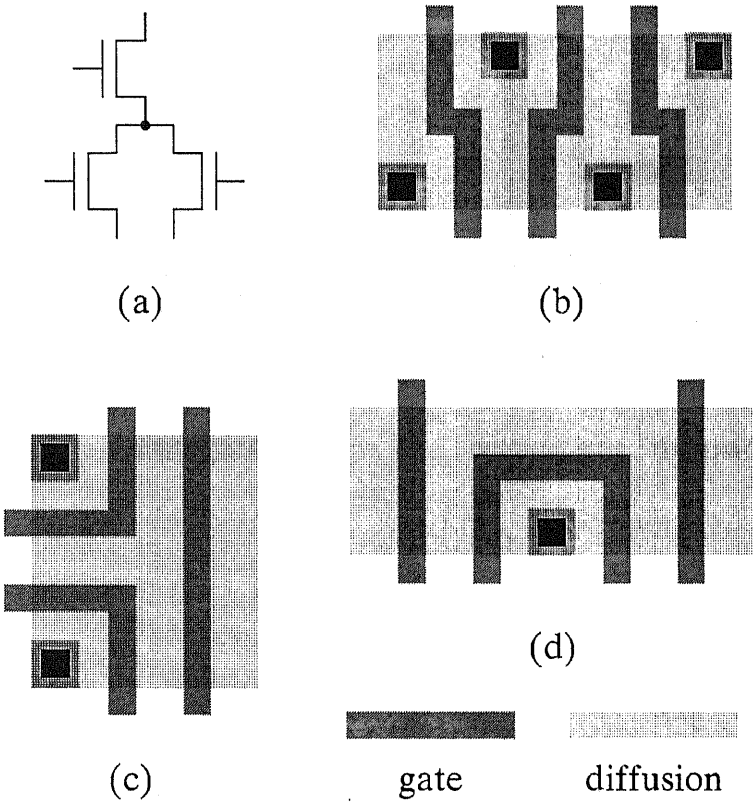


図 2.4: 2 次元の 3FET セル
(b) の面積を 1 としたとき、(c), (d) の面積は 93%, 99% となる。

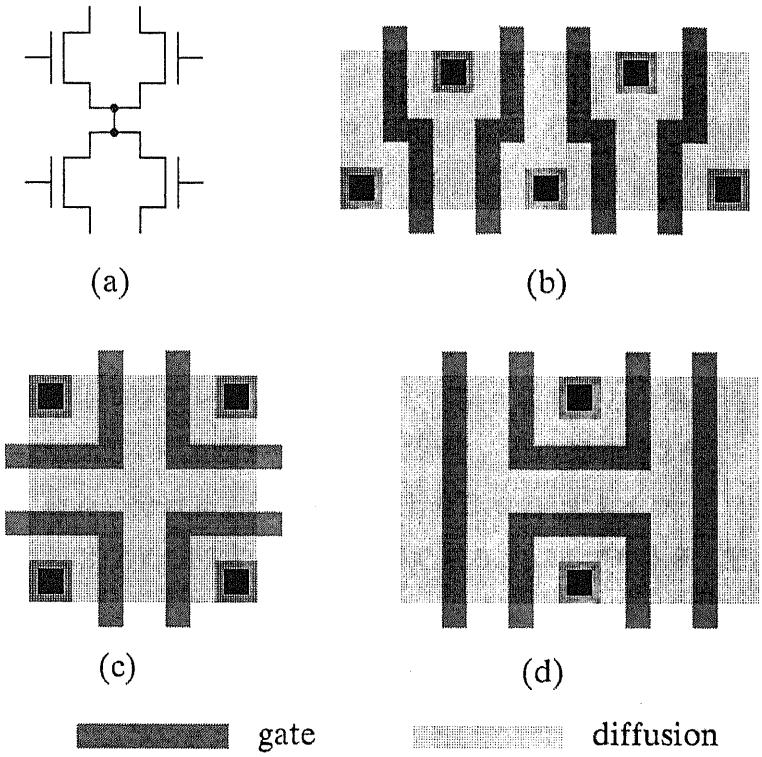


図 2.5: 2次元 4FET セル
(b) の面積を 1 としたとき、(c), (d) の面積は 88%, 115% となる。

を共有するとき、考えられるレイアウト パターンを図 2.5-(c) と (d) に示す。図 2.5-(c) と (d) のレイアウトパターンを **2次元 4FET セルと呼ぶことにする。**

これらのセルのレイアウト面積については、図 2.5-(b) の面積を 1 としたとき、1つの標準的な設計ルールに基づくと、図 2.5-(c) と (d) の面積はそれぞれ 88%,105% となる。

2.4 2次元 FET セルを用いたレイアウト設計

前節に示した2次元 3FET, 4FET セルを用いて実際回路の設計を行ない、そのレイアウト面積を比較する。

まず、図 2.6(a) に示すブリッジ回路のレイアウト設計を行う。ノード1、ノー

ド2は3つのトランジスタの接続点であるので、図2.4に示す2次元 3FET セルを2つ使って、レイアウトすることができる。2つ 3FET セルには6つトランジスタがあるが、その中の1つトランジスタをマージすることができる。よって、図2.6(a) の2次元ソース-ドレーン共有型のレイアウトは図2.6(c) に示すようになる。図2.6(b) は1次元レイアウトの結果である。1次元レイアウトの面積を1としたとき、1つの標準的な設計ルールに基づくと、2次元レイアウトの面積は約90% となる。

つぎに、4入力排他論理のレイアウト設計を行なう。4入力排他論理の回路図の N タイプ トランジスタ部分を図2.7(a) に示す。この図の12個 N タイプ トランジスタを4組みに分けると例えば、左上の3つの N タイプ トランジスタ A, B, $\bar{\text{B}}$ は1つの2次元 3FET セルで実現できる。レイアウト例 (配線は省略されている) を図2.7(c) に示す。この図の真中にある2次元 4FET セルは4つインバータの N タイプ トランジスタである。図2.7(b) は1次元レイアウトの結果である。1次元レイアウトの面積を1としたとき、1つの標準的な設計ルールに基づくと、2次元レイアウトの面積は82% となる。

その他、1ビット加算器、ビット直列乗算器に対しレイアウトした結果では2次元配置の面積は1次元配置の面積の91%, 96% となる。

2.5 2次元 FET セルを用いたレイアウトの面積の評価

以上のレイアウトの結果から2次元レイアウトの面積は1次元の面積より小さいことがわかったが、一般的な回路に対し、面積を見積るための簡単な式を示す。

1次元、2次元レイアウト面積の合計をそれぞれ S_1, S_2 とすると、S_1, S_2 の関係は次の式で書ける。

$$S_1 = S_2 + \sum D_3 + \sum D_4 \tag{2.1}$$

ただし、D_3, D_4 はそれぞれ 3FET, 4FET セルの1次元と2次元との面積の差である。つまり、図2.4, 2.5の (b) と (c), (d) の面積の差である。S_1 がトラン

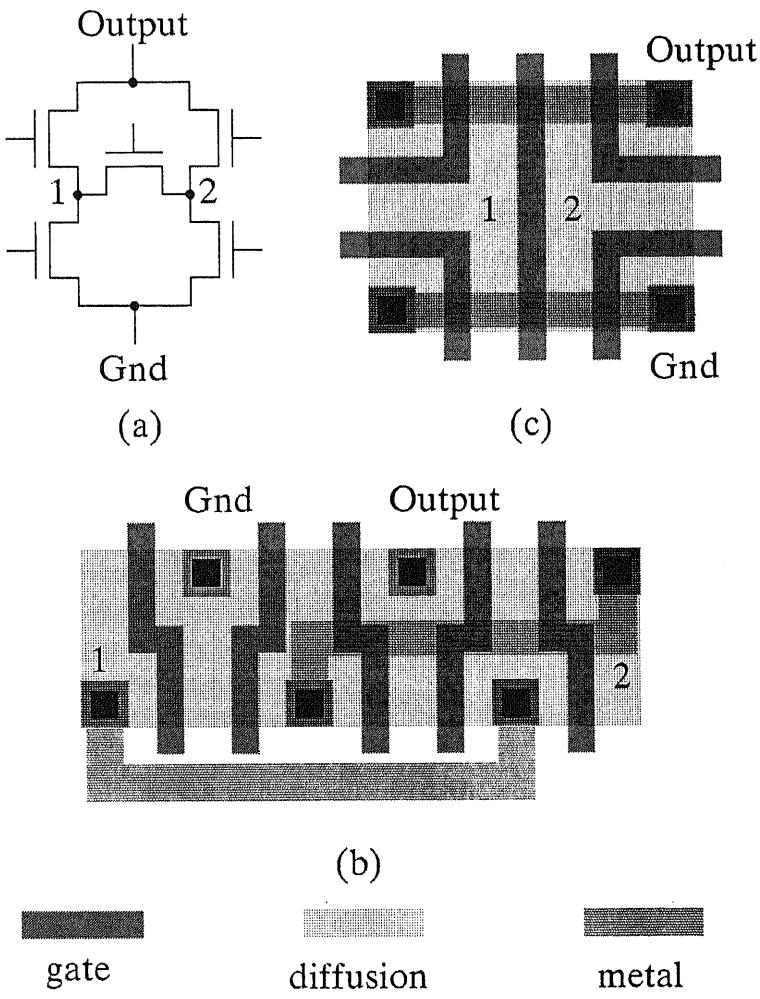


図 2.6: ブリッジ回路の 2 次元レイアウト
(b) の面積を 1 とした時、(c) の面積は 90% となる。

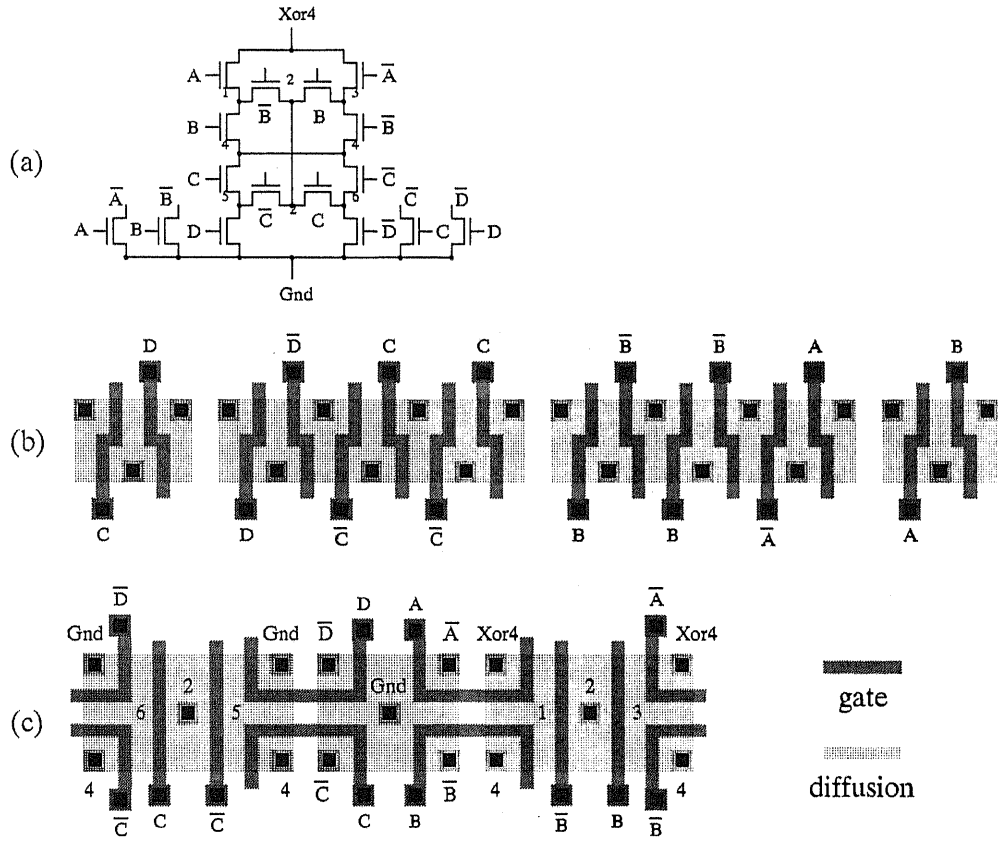


図 2.7: 4 入力排他論理回路の 2 次元レイアウト
(b) の面積を 1 とした時、(c) の面積は 82% となる。

ジスタの n に比例すると仮定すると、S_2 と S_1 の比率はつぎのようになる。

$$R = \frac{S_2}{S_1} = 1 - r_3 \times \frac{3n_3}{n} - r_4 \times \frac{4n_4}{n} \tag{2.2}$$

ただし、n_3, n_4 はそれぞれ 3, 4 トランジスタの接続ノードの数であり、r_3, r_4 はそれぞれ 3FET, 4FET セルの 1 次元レイアウトの面積に対する 1 次元と 2 次元との面積の差の比率である。図 2.4, 2.5により、r_3, r_4 の値は 0 と 0.12 の間である。$2n > 3n_3 + 4n_4$ であるので、R の値は 0.8 と 1 の間となる。したがって、2 次元配置は 1 次元配置より約 10% 程度の面積が減少することが期待される。また、ソース、ドレインを共有することによりメタル配線の数は $n_3 + n_4$ に比例して減少する。

2.6 2次元 FET セルを用いたレイアウトの問題点

トランジスタのネットワークを 1 次元的に展開するより、2 次元的に展開した方がより小さい面積のレイアウトが得られることを示したが、2 次元 FET セルを用いたレイアウトの設計には次の困難と問題が存在する。

まず、ネットワークのどの部分を 3FET あるいは 4FET セルにマッピングするかを決定するのが困難である。例えば、図 2.7-(a) のノード 5、6 を 3FET セルにマッピングするとノード 2、4 を 4FET セルにマッピングすることができなくなる。

また、ゲート幅は図 2.4, 2.5に示しているパターンより小さくならない、あるいは、その中の 1 つトランジスタのゲート幅を大きくすると、他のトランジスタのゲート幅も変わってしまうという問題がある。トランジスタのゲート幅は大きすぎると、それを駆動するトランジスタの負担が大きくなるが、小さいすぎると、駆動能力が足りない。つまり、回路全体の遅延時間を最小化するという観点から、最適なゲート幅が存在する。よって、2 次元 FET セルを用いたレイアウト設計では、ゲート幅の最適化問題は一層難しくなる。

2 次元 FET セルを用いたレイアウトのもう 1 つ問題点として、トランジスタのゲートは曲がることがある。トランジスタのゲートが曲がると、曲がった

ところにおいて、電界集中によって、ブレークダウンやホットギャリヤなどの現象が起こってしまうので、トランジスタのゲートをまっすぐにした方が望ましい。

2.7 ソース-ドレーン共有型レイアウトについて検討した結論

ソース-ドレーン共有型レイアウトは接続関係のあるトランジスタのソースあるいはドレーンをつなぎ合わせることによって、拡散領域の面積を最小化するという特徴を持つレイアウトのことである。1つ拡散領域が共有できるトランジスタの数は、2までのレイアウトを1次元、2以上のレイアウトを2次元ソース-ドレーン共有型レイアウトという。

論理回路の最も基本な素子 NAND ゲートのレイアウト パターンから回路のパラメータを抽出して、シミュレーションした結果では、ソース-ドレーン共有型レイアウトの方がより速いスピードで動作することを確認した。

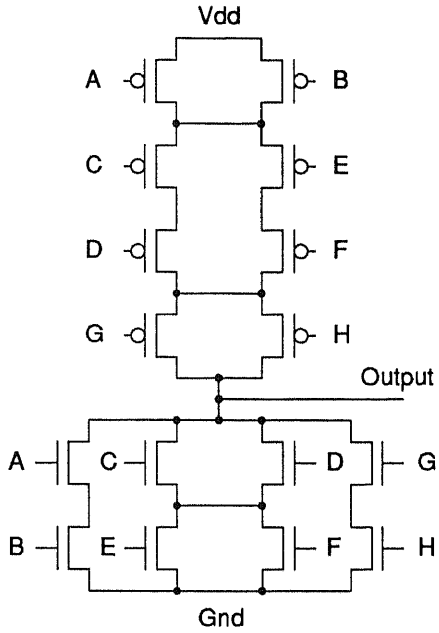
ソース-ドレーン共有型レイアウトが2次元レイアウトに拡張すると、レイアウト面積は1次元レイアウト面積と比べて、最大 20% 平均約 10% 程度面積が減少することが可能である。しかし、2次元レイアウトには、トランジスタのゲートは曲がることがあるといったような問題点が多く、高性能集積回路の自動設計に適用することが難しい。したがって、本研究では、1次元レイアウトのスタイルを採用する。

この章の最後に、本研究で採用する1次元レイアウトの明確な定義を示す。

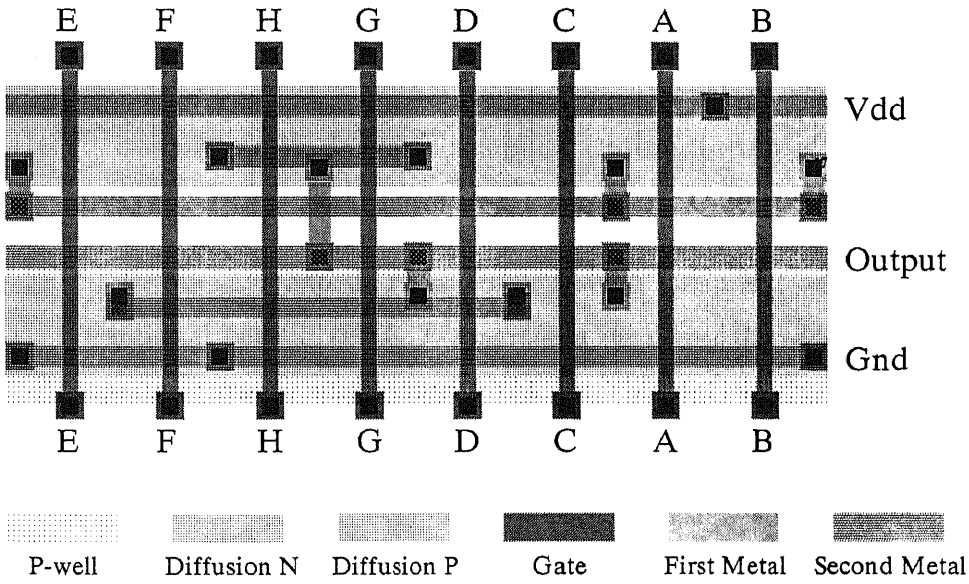
1. 1つ拡散領域を共有するトランジスタの数は最大2である。
2. トランジスタのゲートは曲がらないものとする。
3. ポリシリコン（トランジスタのゲート）と拡散領域（トランジスタのソースあるいはドレーン）は交互の順序に並べる。
4. P タイプ トランジスタを上の行に、N タイプ トランジスタを下の行に置く。

5. 縦の配線は第１レイヤ メタル線、横の配線は第２レイヤ メタル線を使う。

このような１次元レイアウトの１例を図 2.8に示す。



(a) 回路図



(b) レイアウト

図 2.8: 本研究で採用するレイアウ トスタイル

参考文献

[1] O.Wing, S.Huang, and R.Wang. "Gate Matrix Layout". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-4(3):220–231, July 1985.

[2] 浅田 邦博 他. *"MOSES: MOS*集積回路モジュール設計システム *"*. 東京大学出版会, 1991.

[3] Y. Sone, S. Suzuki and K. Asada. "A Gate Matrix Deformation and 3-Dimensional maze routing for dense MOS module generation". *Proceedings of The IEEE 1989 Custom Integrated Circuits Conference (CICC)*, pages 3.5.1–3.5.4, May 1989.

[4] T. Uehara and W. M. vanCleemput. "Optimal layout of CMOS functional arrays.". *IEEE Transactions on Computers*, C-30:305–312, May 1981.

[5] R. L. Maziasz and J. P. Hayes. "Layout optimization of static CMOS functional cells". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-9(7):708–719, July 1990.

第 3 章　一括処理型トランジスタのペアリング手法

CMOS トランジスタ回路を 1 次元にレイアウトするとき、縦に配置するＰタイプ トランジスタとＮタイプ トランジスタをトランジスタ ペアという。同じゲート信号を持つトランジスタが 2 つ以上存在するとき、トランジスタ ペアの作成処理が必要となる。この章では、すべての可能なトランジスタ ペアにスコアを計算して、このスコアを評価関数として、トランジスタのペアを同時に決定するという一括処理型トランジスタのペアリング手法を提案する。

3.1　従来のトランジスタ ペアリング手法とその問題点

CMOS トランジスタ回路のＰ側とＮ側は互いにデュアルな関係を持つ場合、トランジスタ ペアはデュアルなトランジスタからなる。これは最も一般的なトランジスタ ペアリング手法である [1, 2, 3, 4]。しかし、実際に使われる回路の中では、非デュアルな関係を持つものが多く存在するので、非デュアルな CMOS 回路にも適用できるものが必要となる。

AT&T の GENAC セル ジェネレータ [5] では、次のルールの順序でトランジスタのペアを決める。

1. デュアルな関係のあるもの;

2. P タイプと N タイプのトランジスタともソースおよびドレーンに接続関係のあるもの;

3. P タイプと N タイプのトランジスタのゲートおよびソースあるいはドレーンに接続関係のあるもの;

4. P タイプと N タイプのトランジスタのゲートに接続関係のあるもの;

5. P タイプと N タイプのトランジスタのソースあるいはドレーンに接続関係のあるもの;

6. それ以外のもの。

日立の MAGICAL セル ジェネレータ [6] では、トランジスタの直列と並列の関係によってペアを決める手法をとる。これはデュアルな関係でペアを決める手法と基本的には同じものである。

Wimer ら [7] のトランジスタ ペアリングは、トランジスタの 3 つノードの中で、共通ノードの数によってペアを決める手法である。

これらのトランジスタ ペアリング手法は、非デュアルな CMOS 回路について、トランジスタの 3 端子（ソース、ゲート、ドレーン）だけについて調べ、ルール ベースでトランジスタのペアを選択するという逐次処理式アプローチである。明らかなように、このような手法には 2 つの問題点がある。

1. トランジスタの接続関係を取り入れていない。

2. ペアリングの結果は処理の順序に依存する。

トランジスタ ペアの左右に配置されるものは同じタイプのトランジスタであるため、それぞれのトランジスタにはそれぞれのネットワークにおける接続関係を考えなければならない。

本研究では、すべての可能なトランジスタ ペアについて調べ、それぞれのペアのまわりの接続関係を評価して、評価した結果に基づき、すべてのペアを一括に決定するという一括処理型トランジスタ ペアリング手法を提案する。本手法は非デュアルな CMOS 回路を含めて任意な CMOS 回路に適用することができる。本手法で見つかったトランジスタのペアはかならずしもデュアルな関係を持つわけではないが、デュアルな関係で作ったペアと比べて、より少ないパスでトランジスタのネットワークをカバーすることができる。

3.2 コンパチブル ペア パス カバーの定義

本論文では、CMOS 回路を他の文献 [2, 4] と同じように、トランジスタのゲートを枝、ソース - ドレーンの接続を節点としたグラフで表す。

定義 1: グラフ M_p と M_n をそれぞれ PMOS と NMOS 回路を表すとし、縦に配置する P タイプ トランジスタと N タイプ トランジスタをペアといい、

$$P_i = \begin{cases} L_{pi} & G_{pi} & R_{pi} \\ L_{ni} & G_{ni} & R_{ni} \end{cases} \quad \text{あるいは} \quad (m_p, m_n) \tag{3.1}$$

と書く。ただし、G_{pi} と G_{ni} はそれぞれ M_p と M_n の枝で、(L_{pi}, R_{pi}) と (L_{ni}, R_{ni}) はそれぞれ G_{pi} と G_{ni} に連結されている節点である。(L_{pi}, L_{ni}) と (R_{pi}, R_{ni}) は左 EP(End Point) と右 EP と呼ぶことにする。

定義 2: 1 つトランジスタにおいて、EP の組合せは次の 4 タイプがある。

$$type1 \begin{cases} ps_i & pg_i & pd_i \\ ns_i & ng_i & nd_i \end{cases} \quad type2 \begin{cases} ps_i & pg_i & pd_i \\ nd_i & ng_i & ns_i \end{cases}$$

$$type3 \begin{cases} pd_i & pg_i & ps_i \\ ns_i & ng_i & nd_i \end{cases} \quad type4 \begin{cases} pd_i & pg_i & ps_i \\ nd_i & ng_i & ns_i \end{cases} \tag{3.2}$$

ただし、ps_i, pg_i, pd_i と ns_i, ng_i, nd_i はそれぞれ P タイプ トランジスタと N タイプ トランジスタのソース、ゲート、ドレーンのラベル（ノード番号）で

ある。

定義 3: ペア P_i と P_j において、$R_{pi} = L_{pj}$ かつ $R_{ni} = Lnj$ ならば、P_i と P_j はコンパチブルであるという。つまり、P_i と P_j にある 2 つの P タイプ トランジスタと 2 つの N タイプ トランジスタともつなぎ合わせることができる。

定義 4: ペアの列 P_i において、P_i と P_{i+1} がコンパチブルであれば、その列をパスと呼ぶ。1 本パスにあるすべてのトランジスタペアはそのパスに現れる順序で配置し、ペアとペアの間に拡散領域のセパレーションは使わない。

定義 5: グラフ M_n と M_p のすべての枝を 1 度だけ含むパスの集合はそのグラフの 1 つのカバーと呼ばれる。パスとパスの間に拡散領域のセパレーションが必要なので、レイアウト面積の最小化問題は最少パス数のカバーを求める問題と等価である。

定義 6: CMOS トランジスタ回路をグラフで表現するとき、グランド ノードとパワー ノードを切り離すと、元の連結グラフはいくつかの部分グラフに分かれる。その部分グラフのことを**論理ブロック**と呼ぶことにする。

3.3 スコア — トランジスタ ペアの接続関係の評価関数

トランジスタのペアが決まったら、そのペアにある P タイプと N タイプ トランジスタは縦に配置され、そのペアの左右には、それぞれのコンパチブルなトランジスタ ペアが配置される。もし、そのペアとコンパチブルのトランジスタ ペアが存在しなければ、そのペアの隣に配置されるトランジスタ ペアとの間にセパレーションを挿入しなければならない。つまり、トランジスタ ペアリングの結果はレイアウト面積に直接影響する。そのため、トランジスタのペアを選択するときから、そのペアはコンパチブルなペアが存在するかどうかをチェックする必要がある。言い換えると、トランジスタのペアの選択基準として、そのペアとコンパチブルなペアの数は 1 つ考えられるものである。以下では、ペア P_i とコンパチブルなペアの数をペア P_i のスコアと呼ぶことにす

る。この節では、スコアの計算式について説明する。

ゲート ラベルが k である P タイプと N タイプ トランジスタの集合をそれぞれ MG_k^p と MG_k^n とし、つぎのように定義する。

$$MG_k^n = \{m_i | g(m_i) = k \text{ and } t(m_i) = N\},$$

$$MG_k^p = \{m_i | g(m_i) = k \text{ and } t(m_i) = P\}, \tag{3.3}$$

ただし、$g(m_i)$ は トランジスタ m_i のゲート ラベルを求める関数であり、$t(m_i)$ はトランジスタ m_i のタイプ（N あるいは P）を求める関数である。

集合 MG_k^n にあるトランジスタ m_ξ のソースとドレーンに接続されているトランジスタの集合をそれぞれ CS_ξ^n 集合と CD_ξ^n 集合とし、つぎのように定義する。

$$CS_\xi^n = \{m_i | s(m_\xi) = s(m_i) \text{ or } d(m_i), \text{ for } m_\xi \in MG_k^n\},$$

$$CD_\xi^n = \{m_i | d(m_\xi) = s(m_i) \text{ or } d(m_i), \text{ for } m_\xi \in MG_k^n\}, \tag{3.4}$$

ただし、$s(m_i)$ と $d(m_i)$ はそれぞれトランジスタ m_i のソースとドレーンのラベルを求める関数である。

同じように、集合 MG_k^p にあるトランジスタ m_ξ のソースとドレーンに接続されているトランジスタの集合をそれぞれ CS_ξ^p 集合と CD_ξ^p 集合とし、つぎのように定義する。

$$CS_\xi^p = \{m_i | s(m_\xi) = s(m_i) \text{ or } d(m_i), \text{ for } m_\xi \in MG_k^p\},$$

$$CD_\xi^p = \{m_i | d(m_\xi) = s(m_i) \text{ or } d(m_i), \text{ for } m_\xi \in MG_k^p\}, \tag{3.5}$$

集合 MG_k^p にあるトランジスタ m_i と集合 MG_k^n にあるトランジスタ m_j をペア (m_i, m_j) にするとき、図 3.1に示すような 2 通りの配置がある。図 3.1(a) の場合、トランジスタ m_i のソースとトランジスタ m_j のソースは縦に並べられる。この場合、ペア (m_i, m_j) の左側と右側とコンパチブルなペア数はそれぞれ

$$ss_{ij} = \sum_l \psi(CS_i^p | l) \times \psi(CS_j^n | l)$$

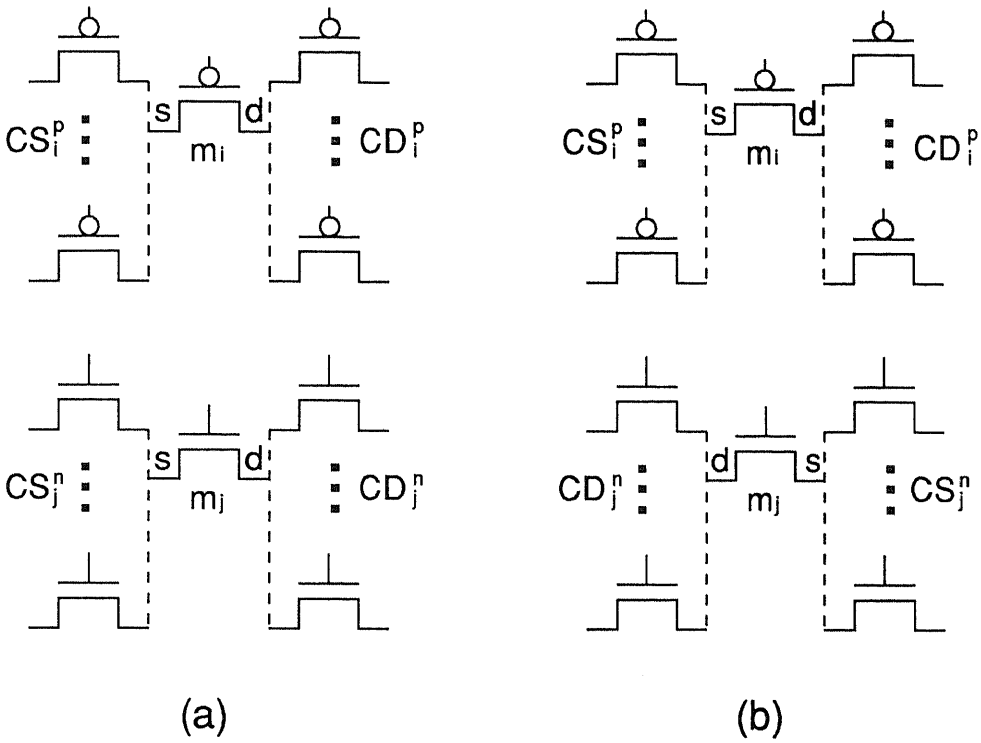


図 3.1: ペア (m_i, m_j) の 2 通りの可能な配置

$$dd_{ij} = \sum_l \psi(CD_i^p|l) \times \psi(CD_j^n|l) \tag{3.6}$$

となる。ただし、$\psi(C_{set}|l)$ は集合 C_{set} にあるゲート ラベル l を持つトランジスタ数を計算する関数である。同じように、図 3.1(b) の場合、トランジスタ m_i のソースとトランジスタ m_j のドレーンは縦に並べる。この場合、ペア (m_i, m_j) の左側と右側とコンパチブルなペア数はそれぞれ

$$\begin{aligned} sd_{ij} &= \sum_l \psi(CS_i^p|l) \times \psi(CD_j^n|l) \\ ds_{ij} &= \sum_l \psi(CD_i^p|l) \times \psi(CS_j^n|l) \end{aligned} \tag{3.7}$$

となる。

ペア (m_i, m_j) とコンパチブルな可能なペア数の合計

$$s_{ij} = ss_{ij} \times dd_{ij} + sd_{ij} \times ds_{ij} \tag{3.8}$$

はそのペアのスコアとして定義される。

3.4 一括処理型ペアリング アルゴリズム

従来のトランジスタ ペアリング手法は、非デュアルな CMOS 回路について、ルール ベースでトランジスタのペアを選択するという逐次処理式アプローチである。それらのルールは、Pタイプと Nタイプ トランジスタの3端子（ソース、ゲート、ドレーン）の中で、接続している端子の数によってペアを決めるものである。これに対して、本研究では、すべて可能なトランジスタ ペアについて調べ、それぞれのペアのまわりの接続関係を評価して、評価した結果 — スコアに基づいて、すべてのペアを一括に決定するという一括処理型トランジスタ ペアリング手法を提案する。この節では、この一括処理型トランジスタ ペアリング アルゴリズムについて詳しく説明する。

コンパチブル ペアはトランジスタの接続関係を表すものである。トランジスタの接続関係はコンパチブル ペアの数によって評価することができる。この評価関数はスコアと呼ばれる。よって、すべての可能なトランジスタ ペアはスコアを持っている。すべての可能なトランジスタ ペアについて、同じ条件でスコアを計算したあと、スコアに基づいて、すべてのトランジスタ ペアを一括に選択するのは提案するトランジスタ ペアリング アルゴリズムの基本的な考え方である。

当然なことであるが、1つトランジスタ ペアとコンパチブルなペアの数は多ければ多いほどパスの長さが長くなるので、スコアの高いトランジスタ ペアを選ぶべきである。長いパスを求める理由については次の章に説明する。しかし、1つのスコアの高いトランジスタ ペアを選ぶと、スコアがゼロであるトランジスタ ペアが残される場合がある。スコアがゼロとは、そのペアの左にも、右にもコンパチブル ペアが存在しないことである。このため、本アルゴリズムでは、スコアがゼロではないトランジスタ ペアの組合せの中から、スコアの合計がもっとも高いものを選択する。

ゲート ラベルが k である P タイプと N タイプ トランジスタの集合にあるトランジスタ数がそれぞれ n_p と n_n である場合、$n_p \times n_n$ のスコア マトリクス $\mathbf{S_k} = [s_{ij}]$ が得られる。つまり、すべて可能なトランジスタ ペア数は $n_p \times n_n$ である。トランジスタ ペアリングは、$n_p \times n_n$ トランジスタ ペアから $n = min(n_p, n_n)$ ペアを選択することである。

トランジスタのペアを一括に決めるため、スコア マトリクス $\mathbf{S_k} = [s_{ij}]$ から、次の条件を満たす最高スコア集合 HS を求める。

1. HS にある要素の数は $n = min(n_p, n_n)$ であること。

2. HS にある各要素はスコアマトリクスの異なる行と列に属すること。

3. HS にあるすべての要素は 0 ではないこと。

4. HS にある要素の合計は最大であること。

集合 HS が決まることにより、n ペアのトランジスタが同時に確定される。P タイプ トランジスタの数と N タイプ トランジスタの数が一致しない CMOS 回路の場合、つまり、$n_p \neq n_n$ のとき、n ペアのトランジスタが決まったら、残りのトランジスタは単独に" ペア" となる。したがって、すべてのトランジスタは同時に確定される。このように選んだトランジスタ ペアは処理の順序に依存しないという特徴がある。

最高スコア集合 HS が存在しない場合、つまり、トランジスタ ペアの組合せの中で、スコアの合計が最大となるものが 2 つ以上存在する場合、どちらの組合せを選択するかをすぐに決めることはできない。このとき、決めることのできるものを先に確定して、確定した結果を利用して、スコアを再計算する。ここで、トランジスタ ペアの選択は一括で確定できなくて、ペアリングの結果は処理順序に依存するように見えるが、実は、すべてトランジスタ ペアのスコアの計算は同じステージ、つまり、同じ条件で行なわれ、どちらかが有利になるわけではない。同じステージで決着をつけないトランジスタのペアは、

次のステージでもう一度選択処理を行なわれることは本アルゴリズムのもう1つ特徴である。

以上をまとめて、提案する一括処理型トランジスタ ペアリング アルゴリズムはつぎの処理からなる。

1. ゲート ラベルの同じPタイプとNタイプ トランジスタの集合 MG_k^p と MG_k^n を作成する。
2. 式(3.8)を用いて、スコア マトリクスを計算する。
3. 最高スコア集合 HS を求める。
4. 最高スコア集合 HS にあるすべてのトランジスタ ペアを同時に確定する。確定できないトランジスタ ペアが存在すれば、ステップ1へ。

計算の効率を向上させるため、一括処理型トランジスタ ペアリング アルゴリズムの処理は論理ブロックごとに行われる。

典型的な CMOS 回路について実験した結果では、この一括処理型ペアリング アルゴリズムは網羅的な方法と同じような結果が得られる。

3.5 一括処理型ペアリング アルゴリズムの応用

この節では、実際の回路を用いて、一括処理型ペアリング アルゴリズムの処理を示す。

図3.2の回路は参考書でよく使われる1ビット加算器回路[8]である。この回路は4つ論理ブロックからなる。4つ論理ブロックの中で、2つ論理ブロック (m_{12}, m_{11}) と (m_{28}, m_{27}) はインバータである。その他の2つ論理ブロックを block-1 と block-2 と呼ぶことにする。

まず、block-1 について、トランジスタのペアの選択処理を説明する。入力信号が A であるPタイプ トランジスタとNタイプ トランジスタの集合はそれぞれ

$$MG_A^p = \{m_3, m_{10}\},$$

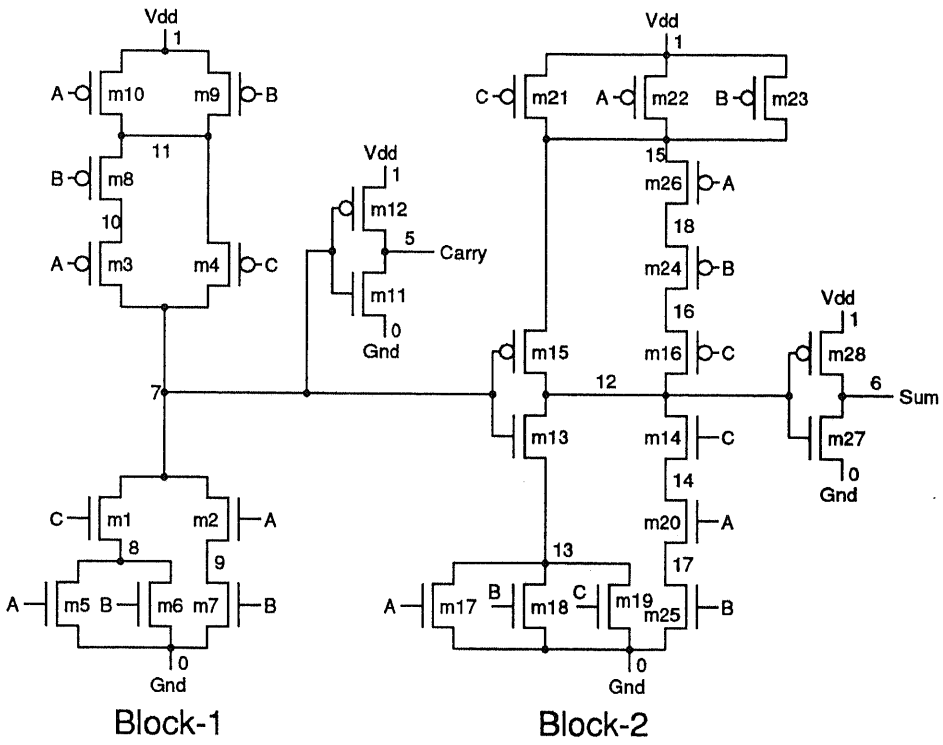


図 3.2: 1 ビット加算器回路

(参考書 [8] より)

Table 3.1: MG_A^p と MG_A^n にあるトランジスタの接続関係

transistor	connections in source	connections in drain
m_3 in MG_A^p	$CS_1^p = \{m8\}$	$CD_1^p = \{m4\}$
m_{10} in MG_A^p	$CS_2^p = \{m9\}$	$CD_2^p = \{m4, m8, m9\}$
m_2 in MG_A^n	$CS_1^n = \{m7\}$	$CD_1^n = \{m1\}$
m_5 in MG_A^n	$CS_2^n = \{m6, m7\}$	$CD_2^n = \{m1, m6\}$

と

$$MG_A^n = \{m_2, m_5\}, \tag{3.9}$$

となる。この2つのPタイプ トランジスタと2つのNタイプ トランジスタからのすべて可能なトランジスタ ペアは $m_3, m_2, m_3, m_5, m_{10}, m_2$ と m_{10}, m_5 と作ることができる。トランジスタ ペアリングはこの4ペアから2ペアを選択する問題である。

MG_A^p と MG_A^n にあるトランジスタのソースとドレーンに接続しているトランジスタの集合は表3.1にまとめられる。

表3.1のトランジスタの集合は式(3.4) と式(3.5) からも求められる。それらのトランジスタの集合から、式(3.8) を使って、スコア マトリクスがつぎのようになる。

$$\mathbf{S_A} = \begin{pmatrix} 1\times1+0\times0 & 1\times1+2\times0 \\ 2\times1+0\times1 & 2\times3+4\times1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 2 & 10 \end{pmatrix}, \tag{3.10}$$

このスコア マトリクスから最高スコア集合 $HS = \{s_{11}, s_{22}\}$ が得られる。すなわち、もっとも多いコンパチブル ペアを持っている2つトランジスタペア $(m3, m2)$ と $(m10, m5)$ は同時に確定される。

block-1 にある入力信号 B のトランジスタについて、同じ計算にしたがって、スコア マトリクス $\mathbf{S_B}$ はつぎのようになる。

$$\mathbf{S_B} = \begin{pmatrix} 4 & 4 \\ 3 & 3 \end{pmatrix}. \tag{3.11}$$

このスコアマトリクスにおいて、$s_{11}+s_{22}=s_{12}+s_{21}$ となるため、トランジスタのペアを確定することができない。この場合、これらのトランジスタのペアは次のステージで再計算することにする。

block-1 にある入力信号 C のトランジスタについては、P タイプと N タイプのトランジスタは 1 つだけであるから、ペア (m_4, m_1) は簡単に確定される。

以上のペアリング結果をまとめると、block-1 において、3 つペア (m_3, m_2), (m_{10}, m_5) と (m_4, m_1) が確定された。この論理ブロックにあるトランジスタがすべて確定されれば、この論理ブロックの処理は終了する。そうではなければ、確定されたトランジスタ ペアを利用して、未確定のトランジスタ ペアについて、スコアの再計算を繰り返して行なう。

例えば、block-1 にある入力信号 B のトランジスタについて、次のステージで、スコア マトリクス $\mathbf{S_B}$ を再計算した結果はつぎのようになる。

$$\mathbf{S_B} = \begin{pmatrix} 0 & 4 \\ 2 & 0 \end{pmatrix}. \tag{3.12}$$

これで、ペア (m_9, m_6) と (m_8, m_7) は確定される。

block-1 にあるトランジスタ ペアは第 2 ステージですべて確定される。その結果から見ると、ペア (m_3, m_2), (m_{10}, m_5), (m_9, m_6) と (m_8, m_7) は非デュアルなトランジスタからなるが、このペアリングの結果によって、図 3.2の回路をカバーするパスの数はこの回路の下限（ロワー バウンド） — 2 になった。CMOS 回路をカバーするパスの下限について 4.7 節に述べる。もし、この回路のトランジスタ ペアをデュアルな関係で作るならば、この回路をカバーするパスの数が 2 となることは不可能である。

3.6 一括処理型トランジスタ ペアリング手法のまとめ

同じゲート信号を持つトランジスタが 2 つ以上存在するとき、トランジスタペアの作成処理が必要となる。従来のトランジスタ ペアリング手法は、非デュアルな CMOS 回路について、ルール ベースでトランジスタのペアを選択する

という逐次処理式アプローチである。これに対して、すべて可能なトランジスタ ペアについて調べ、それぞれのペアのまわりの接続関係を評価して、評価した結果 — スコアに基づいて、すべてのペアを一括に決定するという一括処理型トランジスタ ペアリング手法を提案した。

本手法は次の特徴がある。

1. 非デュアルな CMOS 回路を含めて、任意な CMOS 回路に対して、同じアルゴリズムでトランジスタのペアを作成する。

2. トランジスタのペアはＰタイプ トランジスタとＮタイプ トランジスタの共通ノードの数によって決まるではなく、それぞれのタイプのトランジスタの、それぞれのネットワークにおける接続関係によって決まる。

3. すべて可能なトランジスタ ペアについて調べ、同じ条件で接続関係を評価して、評価した結果に基づいて、トランジスタ ペアを一括に確定する。

本手法の研究を通じて、デュアルな関係でトランジスタ ペアを作るより、非デュアルな関係でトランジスタ ペアを作る方が、より小さい面積なレイアウトが得られる場合があることとわかった。

参考文献

[1] R.L.Maziasz and J.P.Hayes. "Layout optimization of CMOS functional cells". *Proceedings of the 24th ACM/IEEE Design Automation Conference (DAC)*, pages 77–84, July 1987.

[2] R. L. Maziasz and J. P. Hayes. "Layout optimization of static CMOS functional cells". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-9(7):708–719, July 1990.

[3] T.Lengauer and R.Müller. "Linear algorithms for optimizing the layout of dynamic CMOS cells". *IEEE Transactions on Circuits and Systems*, 35:279–285, March 1988.

[4] T. Uehara and W. M. vanCleemput. "Optimal layout of CMOS functional arrays.". *IEEE Transactions on Computers*, C-30:305–312, May 1981.

[5] C.Ong, J.Li, and C.Lo. "GENAC: An Automatic Cell Synthesis Tool". *Proceedings of the 26th ACM/IEEE Design Automation Conference (DAC)*, pages 239–244, June 1989.

[6] Y. Shiraishi, J. Sakemi, M. Kutsuwada, A. Tsukizoe, and T. Satoh. "A Higt Packing Density Module Generator for CMOS Logic Cells". *Proceedings of the 25th ACM/IEEE Design Automation Conference (DAC)*, pages 439–444, June 1988.

[7] S.Wimer, R.Y.Pinter, and J.A.Feldman. "Optimal Chaining of CMOS Transistors in a Functional Cell". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-6(5):795–801, September 1987.

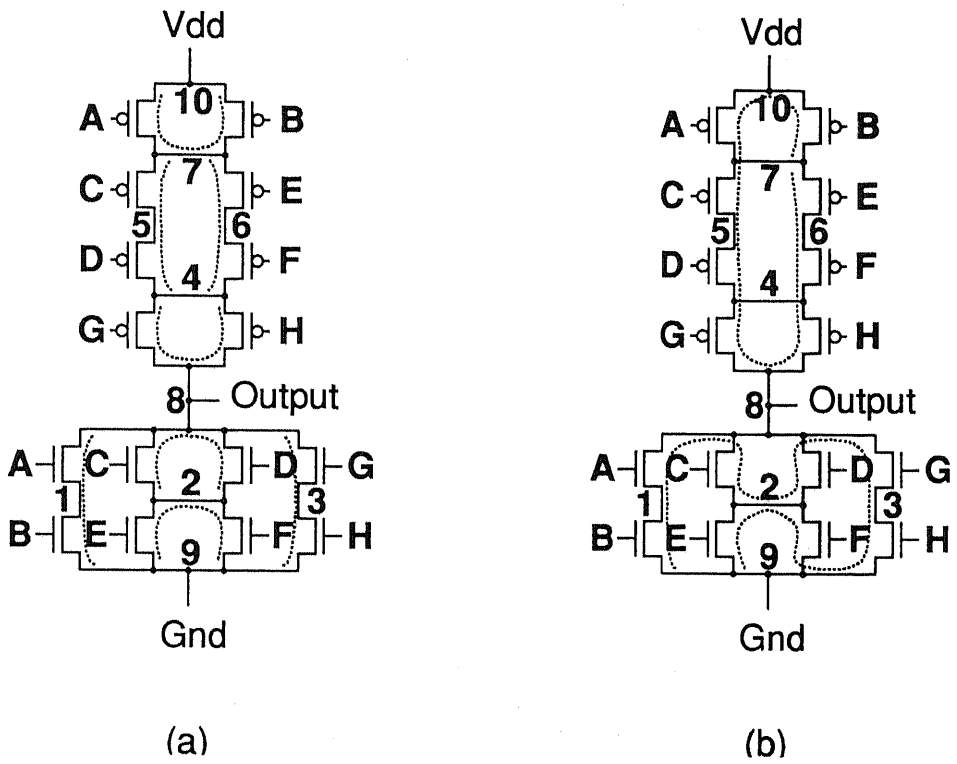
[8] N.Weste and K.Eshraghian. *"Principles of CMOS VLSI Design"*. MA: Addison-Wesley, 1985.

第 4 章　コンパチブル マトリクスを用いたパス探索手法

CMOS 回路を 1 次元レイアウトに展開する手法は、トランジスタのゲートを枝、ソース - ドレーンの接続を節点とした 2 つのグラフ（P グラフと N グラフ）から、枝の順序が同じとなる 2 本のパスを見つけることによって実現する。この章では、コンパチブル ペアの概念を導入して、P グラフと N グラフを 1 つコンパチブル マトリクスで表現して、このコンパチブル マトリクスからパスを探索するというコンパチブル ペア探索アルゴリズムを提案する。

4.1　パス探索によるレイアウト面積の最小化問題

与えられた CMOS トランジスタのネットワークを 1 次元レイアウトに展開するもっとも単純な方法はトランジスタ ペアをランダムに並べるものである。例えば、図 4.1(I)-(a) の回路を図 4.1(II)-(a) に示すように { A, B, C, D, E, F, G, H } という順に並べて、隣にあるトランジスタのソースあるいはドレーンは電気的に接続しているノードであれば、それらのトランジスタ ペアの拡散層をつなぎ合わせて、そうではないノードであれば、トランジスタ ペアの拡散層の間にセパレーションを挿入する。これで、図 4.1(II)-(a) のレイアウトで



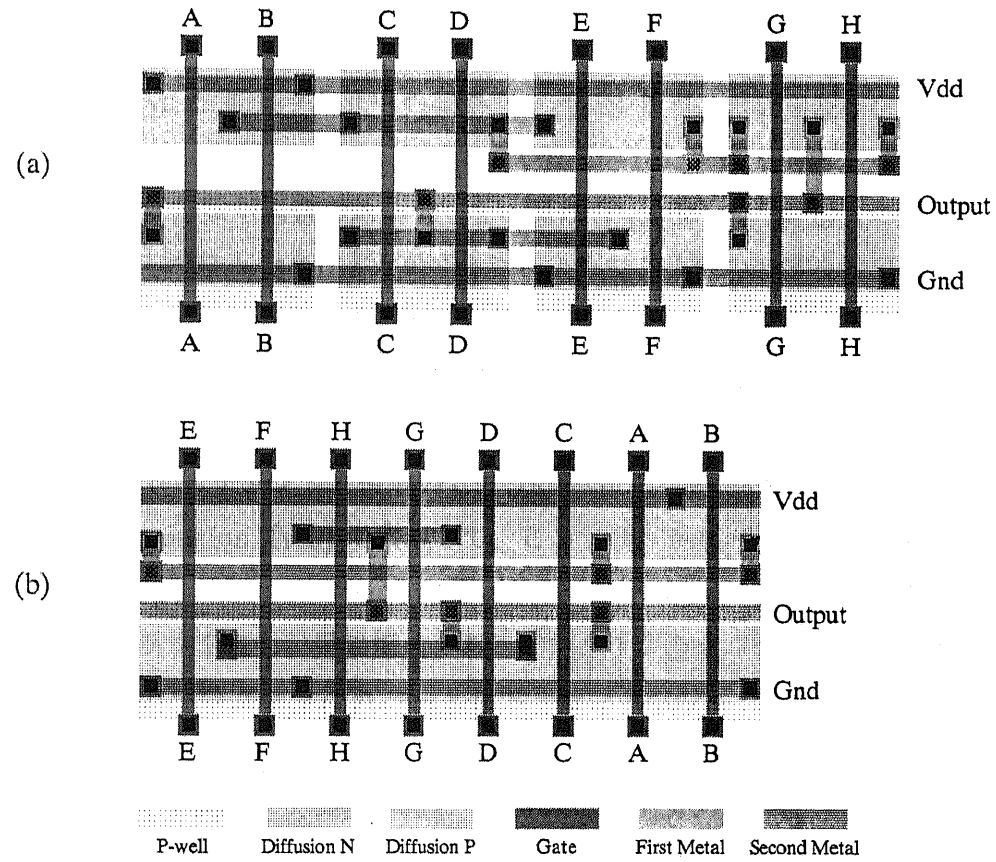
(I) 回路

図4.1: CMOS ネットワークをカバーするパスの数とレイアウト面積の 関係

は、3つのセパレーションが存在する。このレイアウト結果を回路図（図4.1(I)-(a)）から見ると、そのトランジスタのネットワークは4本のパス（点線）でカバーされるということになる。

トランジスタ ペアの並べ順序を変えることにより、セパレーションを減らすことができる。図4.1(I)-(b) はセパレーションのないレイアウトである。このレイアウト結果を回路図（図4.1(I)-(a)）から見ると、そのトランジスタのネットワークは1本のパス（点線）でカバーされるということになる。このときのパスはデュアル オイラー パスと呼ばれる。このときのレイアウトは、セパレーションがないため、最小面積になる。

この例からわかるように、CMOS トランジスタのネットワークをカバーす



(II) レイアウト

図 4.1: CMOS ネットワークをカバーするパスの数とレイアウト面積の 関係

るパスを見つけることによって、その回路はレイアウトに展開することができる。最小面積のレイアウトを探索する問題は最少のパス数を求める問題に帰着される。

4.2 従来のパス探索アルゴリズムとその問題点

与えられた CMOS 回路から、その回路をカバーするパスの数を最小化する最初のアルゴリズムは Uehara ら [1] によって提案されたものである。そのアルゴリズムは、入力数が偶数であるゲートに仮の (Pseudo) 入力を加えて、実の入力と仮の入力が絡み合うことが最小となるように、入力の順序を変更するというヒューリスティックなものである。このようなヒューリスティックな手法およびその後で報告されたいくつかの手法 [2, 3, 4] では、特定の例しか最適解が得られなかった。

Maziasz ら [5, 6] はデュアルな直並列 CMOS 回路の接続関係をコンポジション ツリー (Composition Tree) で表現して、このツリーからパスを探索するアルゴリズムを開発した。このアルゴリズムでは、コンポジション ツリーの節点は1つ部分グラフを表し、兄弟節点にある部分グラフのカバーのすべて可能な連結をその親の部分グラフのカバーとして、コンポジション ツリーのリーフ（1ペア トランジスタ）からボトム アップ的に処理をおこなう。この処理がルートに到達する時、その回路をカバーするすべてのカバーが得られる。パス数が最小となるカバーは解となる。このアルゴリズムは節点の数だけの手間がかかるように見えるが、ルートに近づくにつれて、可能なカバーの数は急激に増えるので、すべての節点についての手間は、同じ仕事量ではない。

その他に、直並列型ダイナミック CMOS 回路のパス探索アルゴリズムは Lengauer らによって提案されている [7]。

以上のアルゴリズムには直並列回路に限定されるという1つの共通な問題点がある。

非直並列 CMOS 回路に関する研究には Ong ら [8] と Wimer ら [9] のアプ

ローチがある。これらのアルゴリズムでは、トランジスタのペアを１つずつ取り、取ってきたトランジスタのペアからなる部分グラフのすべて可能なカバーを作り、すべてのトランジスタ ペアをとり終ったら、パス数の少ないカバーを求めた解とする。しかし、１つトランジスタ ペアを加えるだけで、可能なカバーの数は数倍から数十倍増えることがあるので、彼らのアルゴリズムでは、新しいカバーができたら、あるコスト関数によって、よい結果が得られないと思われるカバーを削除する。よって、このようなアルゴリズムには処理の結果はそのコスト関数および処理の順序に依存するという問題がある。言い換えると、途中で捨てられたカバーが最適解であるかも知れない。

Hwang ら [10] は非直並列 CMOS 回路についても研究し、深さ優先探索アルゴリズムを提案している。このアルゴリズムでは、トランジスタの接続関係をバイパータイト (Bipartite) グラフで表現する。すなわち、グラフの節点はソース - ドレーンの接続ノード（片側の節点はそれぞれ P 回路のノードと N 回路のノード）とし、グラフの枝はそれぞれの節点を共有できるトランジスタのペアとする。彼らのパス探索の処理では、枝を１本ずつ取り、取ってきた枝を並べて、１番長いものを探索する。このアルゴリズムは計算量を減らすため、枝を取る順序はあるコスト関数によって決める。

上述をまとめると、従来のアルゴリズムには次の問題がある。

1. 直並列の CMOS 回路に限定されるという制約がある。

2. 探索の空間はコスト関数によって制限される。

この他に、第３章にすでに述べたトランジスタ ペアリングの問題もある。

これらの問題を克服するため、本研究では、まず、コンパチブル ペアの概念を導入して、トランジスタの接続関係をコンパチブル マトリクスで表現する。これで、非直並列回路を含めて、任意な CMOS 回路を表すことができるようになる。このコンパチブル マトリクスには、探索する必要のないトランジスタ ペアに関する情報も含まれているので、無駄な探索を避けることができる。すべての探索はコンパチブル マトリクスにおいて行なわれるため、高

速な探索が可能となり、探索空間を広げることができる。

4.3 コンパチブル マトリクス

コンパチブル ペアは2つトランジスタ ペアの関係を表すものである。その定義は第3章に与えられた。パス探索の立場から見ると、コンパチブル ペアは次の探索の対象となる。よって、パスの探索はコンパチブル ペアの探索と見ることができる。パスの探索があるトランジスタ ペアに到達するとき、そのペアのすべてのコンパチブルなペアについて調べなければならない。このため、そのペアと他のすべてのトランジスタ ペアの関係を1つテーブルにまとめれば、便利である。このテーブルのことを**コンパチブル マトリクス**と呼ぶことにする。この節では、コンパチブル マトリクスの定義について説明する。

m トランジスタ ペアがある場合、すべてのペアとペア間の関係は $m \times m$ のマトリクスで表すことができる。次のように定義されたマトリクスをコンパチブル マトリクス $\mathbf{C} = [c_{ij}]$ と呼ぶ。

$$c_{ij} = \begin{cases} u/v & i \neq j \text{ かつ } P_i \text{ と } P_j \text{ は} \\ & \text{コンパチブルである} \\ 0 & \text{その他} \end{cases} \tag{4.1}$$

ただし、u と v はそれぞれトランジスタ ペア P_i と P_j のコンパチブル タイプの番号 (1-4) である。トランジスタ ペアのタイプは 3.2 節に定義されたようにPトランジスタと Nトランジスタの配置によって4通りがある。それぞれのタイプに番号を与えている。上式のように定義されたマトリクスからペア P_i のすべてのコンパチブルなペアは i 行目から見つかることができる。

図4.2のブリッジ回路は1つもっとも簡単な非直並列回路である。この回路のすべてのトランジスタペアを表4.1に示す。この回路から作ったコンパチブル マトリクスを表4.2に示す。

Table 4.1: ブリッジ回路のトランジスタペアおよびその4タイプ

pair	type 1	type 2	type 3	type 4
A	$\begin{cases} 4A1 \\ 2A1 \end{cases}$	$\begin{cases} 4A1 \\ 1A2 \end{cases}$	$\begin{cases} 1A4 \\ 2A1 \end{cases}$	$\begin{cases} 1A4 \\ 1A2 \end{cases}$
B	$\begin{cases} 5B1 \\ 6B2 \end{cases}$	$\begin{cases} 5B1 \\ 2B6 \end{cases}$	$\begin{cases} 1B5 \\ 6B2 \end{cases}$	$\begin{cases} 1B5 \\ 2B6 \end{cases}$
C	$\begin{cases} 7C5 \\ 6C3 \end{cases}$	$\begin{cases} 7C5 \\ 3C6 \end{cases}$	$\begin{cases} 5C7 \\ 6C3 \end{cases}$	$\begin{cases} 5C7 \\ 3C6 \end{cases}$
D	$\begin{cases} 7D4 \\ 3D1 \end{cases}$	$\begin{cases} 7D4 \\ 1D3 \end{cases}$	$\begin{cases} 4D7 \\ 3D1 \end{cases}$	$\begin{cases} 4D7 \\ 1D3 \end{cases}$
E	$\begin{cases} 4E5 \\ 3E2 \end{cases}$	$\begin{cases} 4E5 \\ 2E3 \end{cases}$	$\begin{cases} 5E4 \\ 3E2 \end{cases}$	$\begin{cases} 5E4 \\ 2E3 \end{cases}$

図4.2のブリッジ回路は5つのトランジスタ ペアからなる。それぞれのゲートのラベルを {A,B,C,D,E} とする。ゲート両側の数字はソースあるいはドレーンのノード番号である。1つのペアにおいて、Pトランジスタと Nトランジスタのソース、ドレーンの配置によって4タイプがある。

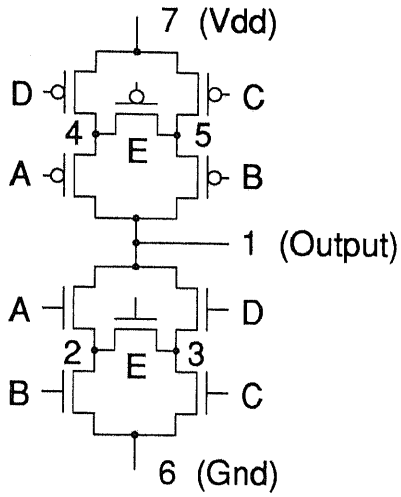


図 4.2: もっとも簡単な非直並列回路 — ブリッジ回路

Table 4.2: ブリッジ回路のコンパチブルマトリクス

pair	A	B	C	D	E
A	0	2/4	0	3/4	4/2
B	1/3	0	4/3	0	3/4
C	0	2/1	0	3/1	1/3
D	1/2	0	4/2	0	2/1
E	3/1	1/2	2/4	4/3	0

このコンパチブル マトリクスは表4.1にしたがって作ったものである。例えば、表4.1において、ペア A のタイプ2の右のノード番号はペア B のタイプ4の左のノード番号と一致するので、要素 $(1,2) = 2/4$ となる。

4.4 コンパチブル ペア探索アルゴリズム

コンパチブルマトリクスの定義から、ペア P_i のすべてのコンパチブルなペアは i 行目から見つかることができる。これによって、コンパチブルペアの探索が可能となる。コンパチブルペア探索アルゴリズムはつぎの処理からなる。

1. コンパチブルマトリクスの i 行目から 0 ではないすべての要素を探し、$c_{ij} = u/v$ が見つかったら、タイプ u のペア P_i はカレントパスの最初のノードとなり、タイプ v のペア P_j は次のノードとなる。

2. マトリクスの j 番目の行から 0 ではないすべての要素を探し、$c_{jk} = v/w$ が見つかったら、タイプ w のペア P_k はカレントパスの次のノードとなる。

3. $j = k, v = w$ にしてステップ 2 へ戻り、コンパチブル ペアが見つからなくなるまで探索を繰り返す。

このようにペア $P_i(i = 1, \ldots, m)$ で始まるすべてのパスを見つけることができる。

一般には、グランドあるいは電源ノードの拡散領域の共有より、信号ノードの共有の方が望ましいので、コンパチブルペアの探索はグランドあるいは電源ノードを持つトランジスタのペアから始まる。

4.5 最長パス優先探索アプローチ

与えられた CMOS 回路をカバーする最小数のパスの集合を求める問題 — パス カバーリング問題は集合分割 (Set Partition) 問題と同じような問題である。この種類の問題は NP- 完全問題として知られている。このため、次の最長パス優先探索アプローチが開発された。

1. コンパチブルペア探索アルゴリズムにより、見つかったすべてのパスの中から 1 番長いパスを選択する。

2. 1番長いパスにあるペアをコンパチブルマトリクスから外して、コンパチブルペア探索アルゴリズムを繰り返す。

これはパスカバーリング問題を解決する効率のよいアプローチである。デュアルなグラフについて、このアプローチは最小数のパスが得られるかどうかはまた証明されていないが、単一のグラフについては、このアプローチは最小数のパスが得られることを簡単に証明することができる（付録）。

表4.2から、コンパチブル ペア探索アルゴリズムによって、最初に得られた最長パスは $\{C(3), D(1), A(2), B(4)\}$ である。括弧にある番号はそのペアのタイプ番号である。このパスにある4トランジスタ ペアをコンパチブル マトリクスから削除すると、ペア E だけが残り、2本目のパスはペア E だけからなる。

4.6 コンパチブル ペア探索アルゴリズムの効率

本研究で開発したコンパチブル ペア探索アルゴリズムには、次の特徴があるため、効率のよいアルゴリズムであると言える。

1. コンパチブル ペアのみの探索；

2. 最長パス優先の探索；

3. 論理ブロックごとの探索。

一般の CMOS 回路については、グランドとパワー ノードを除いて、ほとんどのノードは1から4まで個トランジスタのソースあるいはドレーンと接続している。つまり、CMOS 回路を表すグラフのほとんどの節点の次数は4以下である。したがって、コンパチブル ペアの探索は論理ブロックごとに行なえば、コンパチブル マトリクスはスパースなマトリクスと言える。

1つトランジスタ ペアとコンパチブルなペアの数は複数（例えば、4）であっても、コンパチブル ペアを探索するとき、1つコンパチブル タイプだけ

について調べるので、ほとんどの場合、0から2までのトランジスタ ペアしか存在しない。

さまざまの代表的な CMOS 回路に対して実験した結果を見ると、このコンパチブル ペア探索アルゴリズムの計算複雑度はおよそトランジスタ数の2乗に比例することがわかった。

4.7 CMOS 回路をカバーするパス数の最小値

この章の最初の節にも説明したように、与えられた CMOS 回路をカバーするパス数が少なければ少ないほど小さい面積のレイアウトが得られる。したがって、パス探索アルゴリズムによって見つかったパスの数はそのアルゴリズムを評価する基準となる。この節では、本研究で提案するコンパチブル ペア探索アルゴリズムを評価するため、与えられた CMOS 回路をカバーするパス数の最小値を導く。

文献 [11] から次の定理がある。

定理: グラフ M にオイラー パスが存在するための必要十分条件は、M が連結なグラフかつ次数が奇数である節点がただ2つある（またはない）グラフであることである。

この定理から、簡単に次の推論が得られる。

推論: M は連結なグラフかつ次数が奇数である節点が $2g$ 個あるグラフであれば、M はかならず、g 個より少なくないオイラー パスが存在する部分グラフに分割される。

以下では、1つグラフにおいて、次数が奇数である節点の数の半分をそのグラフをカバーするパス数の最小値あるいはロワー バウンドといい、P_{min} と表す。CMOS 回路の場合、P側のグラフのロワー バウンドとN側のグラフのロワー バウンドの中で、大きい方を CMOS 回路のロワー バウンドとする。

4.8 コンパチブル マトリクスを用いたパス探索手法のまとめ

CMOS 回路のレイアウト面積の最小化問題はその回路をカバーするパスの数を最小化する問題に帰着することができる。

この問題を解決するため、提案されている従来のアプローチには、直並列のCMOS 回路に限定され、探索空間はコスト関数によって限定されるという問題が存在する。これらの問題を克服するため、本研究では、コンパチブル ペア探索アルゴリズムを開発した。

本手法は、トランジスタの接続関係をコンパチブル マトリクスで表現することにより、非直並列、非デュアルな回路を含めて、任意な回路に対しても適用できる。導入したコンパチブル マトリクスには、探索する必要のないトランジスタ ペアに関する情報が含まれているので、無駄な探索を避けることができる。すべての探索はコンパチブル マトリクスにおいて行なわれるため、探索空間を制約しなくても、高速な探索が可能となる。

参考文献

[1] T. Uehara and W. M. vanCleemput. "Optimal layout of CMOS functional arrays.". *IEEE Transactions on Computers*, C-30:305–312, May 1981.

[2] R. Nair, A. Bruss, and J. Reif. "Linear-time algorithms for optimal CMOS layout". *VLSI: Algorithms and Architectures, P. Bertolazzi and F. Luccio, eds.*, Amsterdam, The Netherlands: North-Hooland:627–338, 1985.

[3] D. Hill. "Sc2: A hybird automatic layout system". *IEEE International Conference on Computer-Aided Design, ICCAD-85, Digets of Technical Papers*, pages 172–174, November 1985.

[4] Y.Z.Liao and C.K.Wong. "An algorithm to compact a VLSI symbolic layout with mixed constraints". *Proceedings of the 20th ACM/IEEE Design Automation Conference (DAC)*, pages 107–112, June 1983.

[5] R.L.Maziasz and J.P.Hayes. "Layout optimization of CMOS functional cells". *Proceedings of the 24th ACM/IEEE Design Automation Conference (DAC)*, pages 77–84, July 1987.

[6] R. L. Maziasz and J. P. Hayes. "Layout optimization of static CMOS functional cells". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-9(7):708–719, July 1990.

[7] T.Lengauer and R.Müller. "Linear algorithms for optimizing the layout of dynamic CMOS cells". *IEEE Transactions on Circuits and Systems*, 35:279–285, March 1988.

[8] C.Ong, J.Li, and C.Lo. "GENAC: An Automatic Cell Synthesis Tool". *Proceedings of the 26th ACM/IEEE Design Automation Conference (DAC)*, pages 239–244, June 1989.

[9] S.Wimer, R.Y.Pinter, and J.A.Feldman. "Optimal Chaining of CMOS Transistors in a Functional Cell". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-6(5):795–801, September 1987.

[10] C.Hwang, Y.Lin Y.Hsieh, and Y.Hsu. "A Fast Transistor-Chaining Algorithm for CMOS Cell Layout". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-9(7):781–786, July 1990.

[11] F. Harary. *"Graph Theory"*. MA: Addison-Wesley, 1969.

付録 A　最長パス優先探索アプローチによるグラフ分割の証明

定義: 最も長いパス優先探索によってグラフを分割する処理は**最長パス優先探索アプローチ**と呼ばれる。

定理: $2k$ 個の節点の次数が奇数である連結グラフ G は最長パス優先探索アプローチによって、k 個のオイラー パスが存在するグラフに分割される。

証明:

仮定: G をおおう最も長いパスからなる部分グラフを SG_1 とする。

SG_1 に次数が奇数である節点は必ず2個を含まれる。よって、G から SG_1 を除いた部分グラフ $G' = G - SG_1$ には、次数が奇数である節点は $2k-2$ 個含まれる。部分グラフ G' について次の2つケースに分けて考える。

1. G' は連結グラフである。あるいは G' は非連結グラフであるが、G' にある非連結な部分グラフの中で、次数が奇数である節点を含まないグラフが存在しない。

2. G' は非連結グラフであり、G' にある非連結な部分グラフの中で、次数が奇数である節点を含まないグラフが存在する。

ケース1の場合、帰納法によって G' は $k-1$ 個部分グラフに分割される。つまり、G は k 個オイラー パスが存在するグラフに分割される。ケース2は起こり得ないことである。仮に、ケース2が起こったとして、G' にある非連結な部分グラフの中で、次数は奇数である節点を含まないグラフをそれぞれ $EG_1, EG_2, \cdots, EG_h$ とし、G は連結グラフであるため、SG_1 は必ず $EG_i, (i = 1, 2, \cdots, h)$ とどこかにつながっている。言い換えると、SG_1 は G をおおう最も長いパスからなる部分グラフであるという仮定に矛盾する。（証明終）

第 5 章　CMOS モジュール ジェネレータ GmC の製作と評価

モジュール ジェネレータは、セル ライブラリを持たない大規模集積回路の自動設計システムの 1 部分として、あるいは半導体デバイス テクノロジーの進歩によってセル ライブラリを更新するためのツールとして、非常に重要なレイアウト自動設計技術である。本研究では、CMOS トランジスタ回路のモジュール ジェネレータ GmC を製作した。このジェネレータの中心となるものはこれまで説明してきた一括処理型トランジスタ ペアリング アルゴリズムとコンパチブル ペア探索アルゴリズムの 2 つ独特なアルゴリズムである。この章では、このシステムの概要と特徴について説明したあと、実際回路の設計を通じて、このモジュール ジェネレータ、そして、提案した 2 つアルゴリズムの評価を行なう。

5.1　GmC の概要と特徴

GmC は CMOS トランジスタ回路を 1 次元ソース - ドレーン共有型レイアウトに展開するモジュール ジェネレータである。GmC は MOSES — MOS 集積回路モジュール設計システム [1] における位置付けを図 5.1に示す。GmC

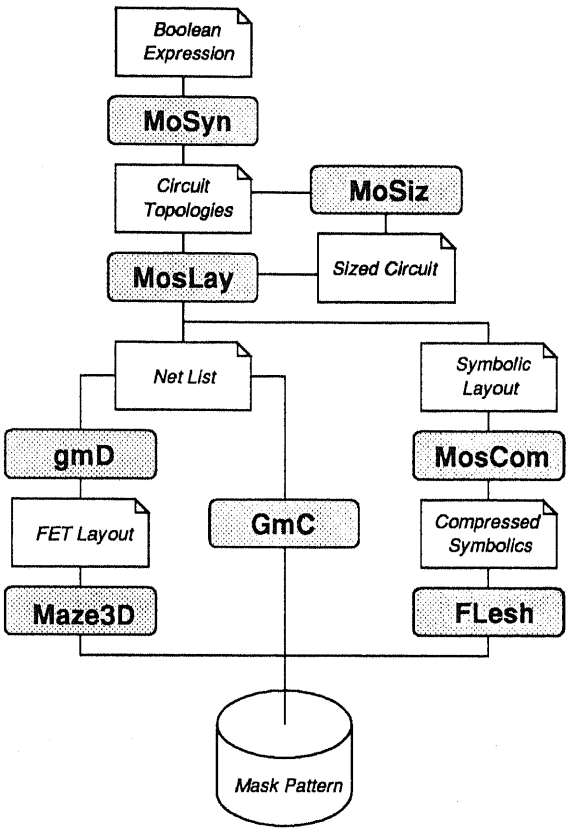


図 5.1: MOSES システムの中での GmC の位置付け

の入力は SPICE の入力フォーマットと同じようなトランジスタ回路記述ファイルである。**GmC** の出力は TIG フォーマット [1] のマスク パターンであり、グラフィクス エディタ **GeX** で見ることができる。**GmC** は自動的に設計ルール記述ファイルを読み込み、その設計ルールにしたがってレイアウトを行なう。また、トランジスタの寸法は記述していない場合、**GmC** は設計ルールの最小寸法でトランジスタを生成する。なお、論理式を入力とする場合、まず、MoSyn[1] の入力フォーマットで論理式を記述して、MoSyn によって論理合成を行ったあと、MosLay によって SPICE の入力フォーマットに変換すればよい。

GmC の構成および処理の流れは次 6 部分がある。

1. トランジスタ回路の記述ファイルからトランジスタの接続情報を抽出して、トランジスタのネット リストを作る。

2. 一括処理型トランジスタ ペアリング アルゴリズムを用いて、最適なトランジスタ ペアを選択する。

3. コンパチブル ペア探索アルゴリズムを用いて、トランジスタのネットワークをカバーするパスの数を最小化する。

4. 配線のチャネル幅が最小となることを第 1 の評価基準、配線の長さの合計が最小となることを第 2 の評価基準として、トランジスタの最適な配置を求める。

5. チャネル配線法を用いて、2 層メタルで配線を行なう。

6. 設計ルールにしたがってマスク パターンを生成する。

この中で、GmC の中心となるものは第 2 部分と第 3 部分である。この 2 つ新しいアルゴリズムを使うことにより、非直並列、非デュアルな CMOS 回路を含めて、任意な CMOS 回路に対しても、同じ処理でコンパクトなレイアウトを設計することができる。同じ処理とは、プログラムにオプション スイッチを指定しなくてもよいことである。

5.2 GmC のトランジスタ配置手法

5.2.1 論理ブロック限定による 1 次元配置手法

コンパチブル ペア探索アルゴリズムによって、CMOS トランジスタのネットワークをカバーする最少のパスが求められる。つまり、求められたパスを 1 次元並べる時、横幅が最小となる。しかし、これらのパスの並べ順序によって、横切る配線の混雑度が変わる。混雑度が大きいほど、レイアウトの高さが増える。チャネル配線では、この混雑度はトラック数あるいはチャネル幅と呼ばれる。そこで、GmC では、トランジスタ配置は次の 2 つ基準で評価する。

第1評価基準: 配線のチャネル幅が最小である;

第2評価基準: 配線の長さの合計が最小である。

チャネル幅は、トランジスタの配置が決まる前にはわからないものである。よって、最適な配置を求めるため、あらゆる配置に対して、配線を行なわなければならない。しかし、トランジスタの配置は組合せ問題であり、探索空間は非常に広い問題である。例えば、パス数が4である場合、１９２通りの配置があるが、8である場合、5百万通り以上の配置もある。このため、**GmC** では、クラスタリング手法を用いて、コンパチブル ペア探索アルゴリズムによって求められたパスをグループに分けている。それぞれのグループに対して、トランジスタ配置の最適化を行なう。

具体的に言うと、まず、同じ論理ブロックに属するパスを1つのグループにする。論理ブロックとは、CMOS 回路のグランド ノードとパワー ノードを切り離した時、ソース - ドレーンによってつながっている部分回路のことである。次に、グループとグループの間の接続数によって、さらに大きなグループを作り上げる。大グループに含まれるグループの数は4とする。4グループにする理由は **GmC** の配線能力に適当であるという単純なものである。**GmC** の配線能力については、１００トランジスタ規模の回路は SUN ワークステーションで 0.05 秒から 0.1 秒くらいで、4グループの１９２通りを調べる時間は数秒で完了するが、5グループにすると、１９２０通りの配置があるため、十倍の時間が必要となる。

5.2.2 パスの再構成による2次元配置手法

コンパチブル ペア探索アルゴリズムによって求められたパスを1次元に展開すると、横長くなるの場合がある。このため、**GmC** では、与えられたモジュールのレシオに合わせて、パスを再構成して、トランジスタの2次元配置を行なう。

まず、図 5.2(a) の1ビット全加算器回路について考える。その回路をカバー

するすべてのパスを図5.2(b) に示す。例えば、これらのパスを2行に配置すると、2行とも同じくらいの長さになるため、一番長いパス Path1 をどこかで切って、折り返さなければならない。この場合に、パスの数が1本増えるかもしれない。しかし、Path1 の 2, 8 番目のノードと Path2 の2番目のノードに注目すれば、それらノードはグラフ上の同じノードであるとわかる。このようなノードを CP 点 (Cross Point) ということにする。Path1 と Path3 にも他の CP 点が存在する。2本のパスは CP 点で交差すると、様々の長さのパスを再構成することができる。例えば、図5.2(b) の2本のパス Path1 (長さ =11) と Path2 (長さ =2) は図5.2(b) に示すように、長さが8と5の2本のパスに変更することが可能である。

パスを再構成するため、すべてのパスを CP 点で切る。切られた部分パスをセグメントという。1つ回路をカバーするすべてのパスの列をその回路のカバーと呼ぶ。パスを再構成するということはさまざまなカバーを求めることを意味するので、セグメントを交換することにより、2次元配置に合うカバーが求められる。効率のよい交換をするため、セグメント間の関係をペアの関係と同じようにコンパチブル マトリクスで表し、コンパチブルではないセグメントの交換をなくす。

2次元の配置を、m ペアのトランジスタを $P \times F$ のアレーに並べると考えるとき、その処理は次のようになる。

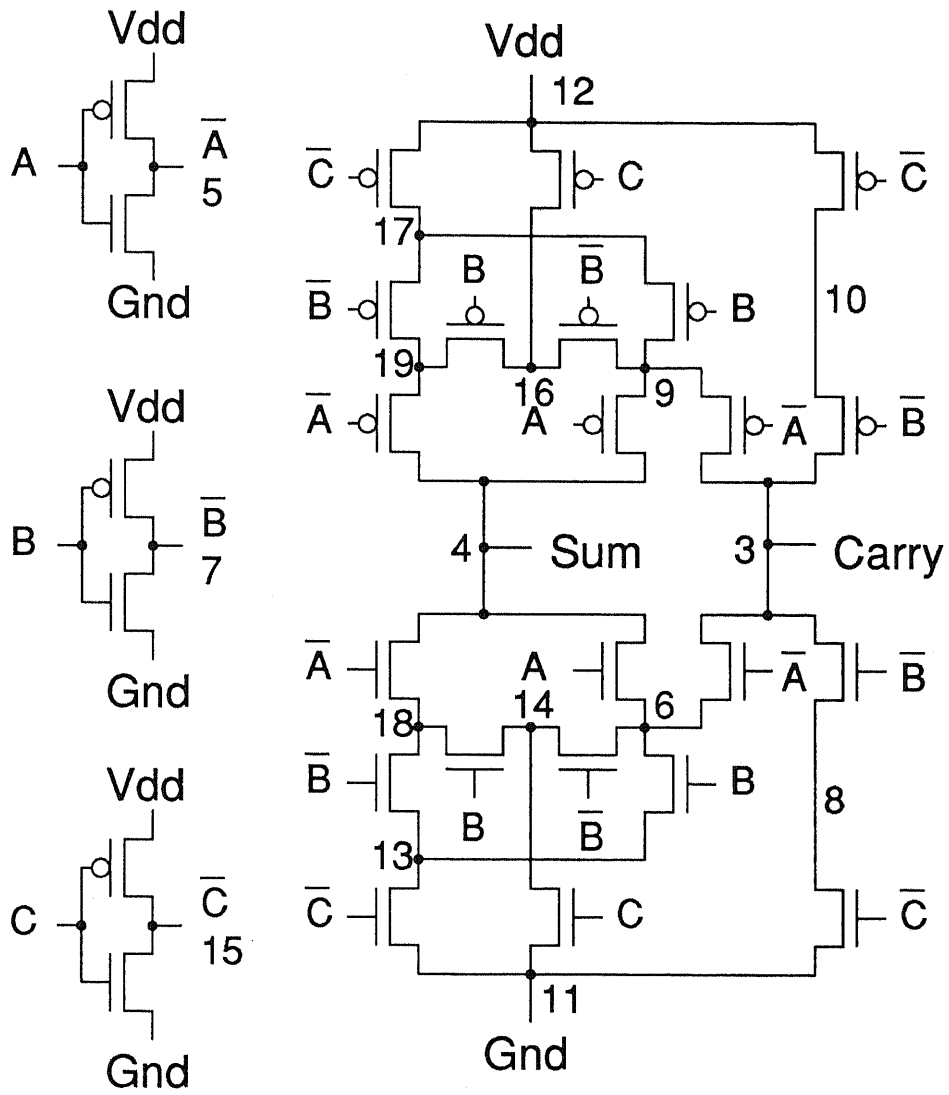
1. アスペクトから行数 P と列数 F を求め、各行のトランジスタペア数 F_i を計算する。

2. CP 点を探して、セグメントを定義する。

3. コンパチブルセグメント (CS) マトリクスを生成する。

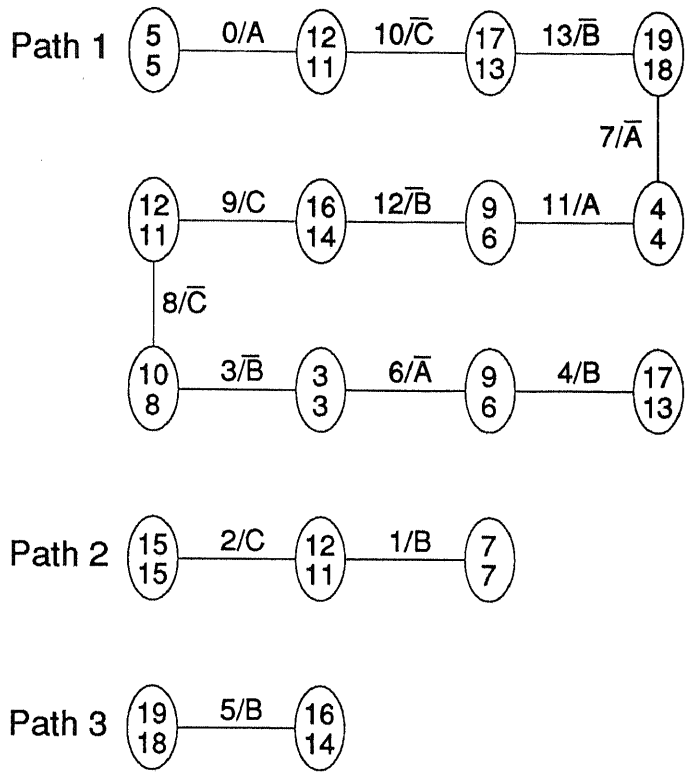
4. CS マトリクスの制約に基づく、セグメントの交換を行なう。

5. 得られたカバーについて調べ、すべてのパスを切らなくでも、P パティションに分割し、各パティションのパスの長さの合計が F_i となれば、終



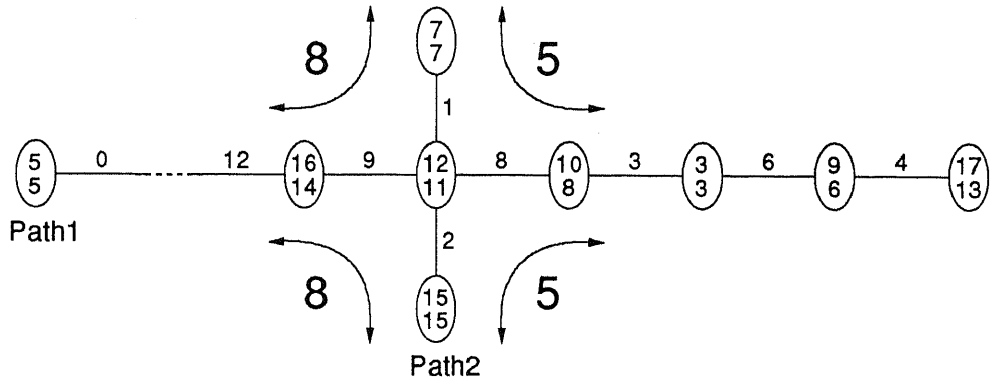
(a) 回路図

図 5.2: 2 次元レイアウトのためのパス再構成手法



(b) 回路をカバーするすべてのパス

この図 (b) は P- グラフにあるパスと N- グラフにあるパスをまとめて 1 本で表しているものである。ノードにある番号はトランジスタのソースあるいはドレーンの番号であり、上の方は P トランジスタ、下の方は N トランジスタの番号である。



(c) パスの再構成例

図 5.2: 2 次元レイアウトのためのパス再構成手法

了する。そうではなければ、ステップ4へ戻る。

4ビット全加算器の2次元レイアウト結果を図5.3に示す。

5.3 直並列回路のレイアウト設計による GmC の評価

この節では、CMOS トランジスタ回路のモジュール ジェネレータ **GmC** を使って、直並列回路に対してレイアウト設計を行なう。設計の結果から、**GmC** の2つの核心のアルゴリズム — 一括処理型トランジスタ ペアリング アルゴリズムおよびコンパチブル ペア探索アルゴリズムを評価する。

実験は MOSES — MOS 集積回路モジュール設計システムにあるすべての例題および教科書と文献にある典型的な例題について行なわれた。実験結果の一部を表5.1 に示す。この表にある回路は次の通りである。

Cell 1: 文献 [2] からの例であり、その回路とレイアウトをそれぞれ図5.4-(a) と図5.4-(b) に示す。

Cell 2: 参考書 [3] の３１２ページにある例題であり、その回路図とレイアウトをそれぞれ図5.5-(a) と図5.5-(b) に示す。

Cell 3: 参考書 [3] の１６１ページにある例題である。

この表にある $\mathbf{P_{min}}$ はその回路をカバーするパスの数の下限であり、**Paths** はそれぞれのアルゴリズムによって見つかったパス数である。
$\mathbf{P_{min}} = \mathbf{Paths}$ となれば、そのアルゴリズムは最適解が求められることを意味する。CPU 時間にはトランジスタの配置と配線を行なう時間も含まれている。

Maziasz らのアルゴリズム [2] は、直並列回路に対して最適解が求められると知られているアルゴリズムであるので、今回の実験では、それと比較する。

実験の結果から見ると、**GmC** によって見つかったパスの数はすべて下限に達したことがわかった。

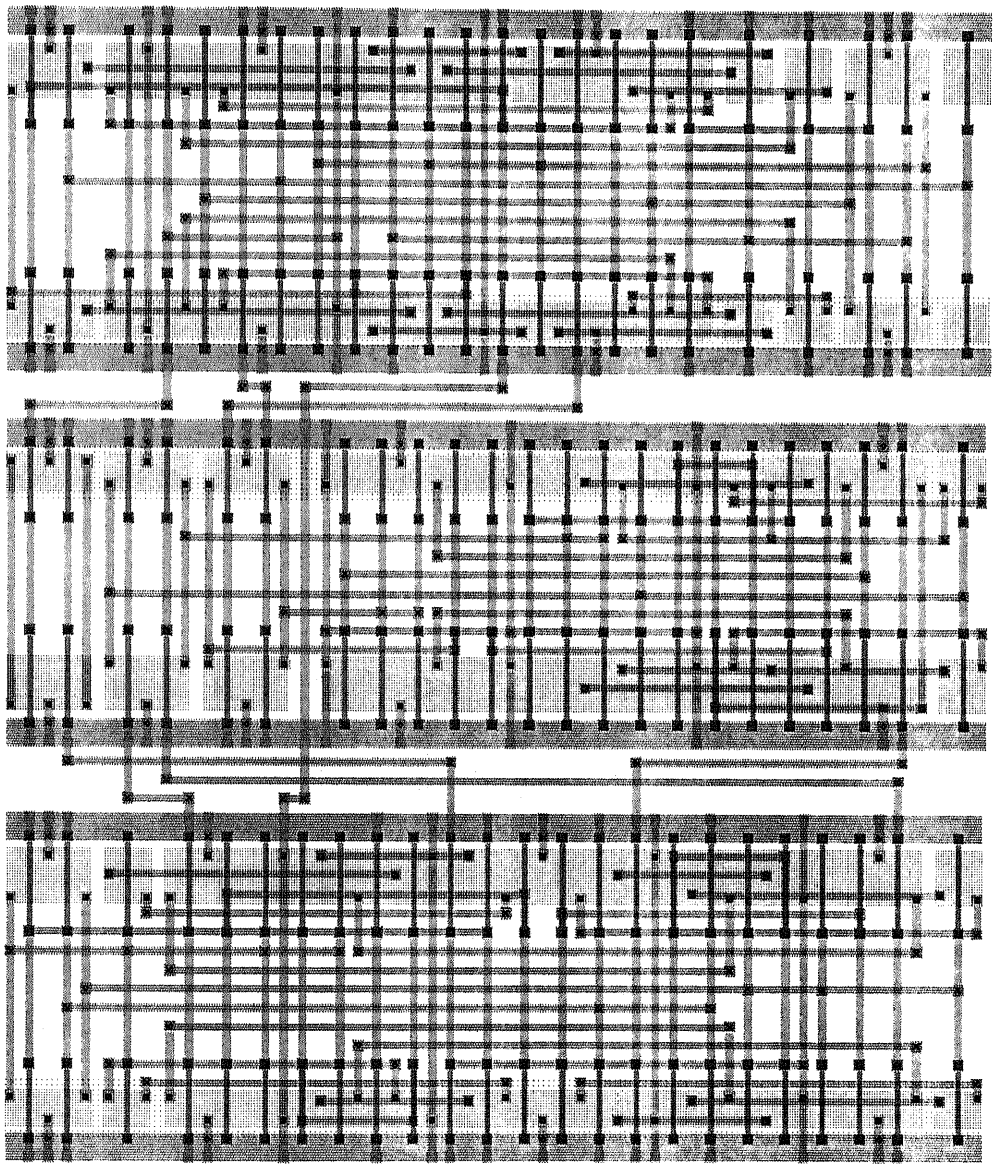
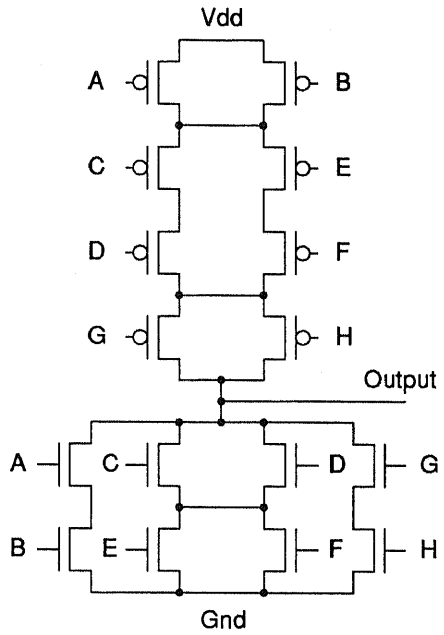
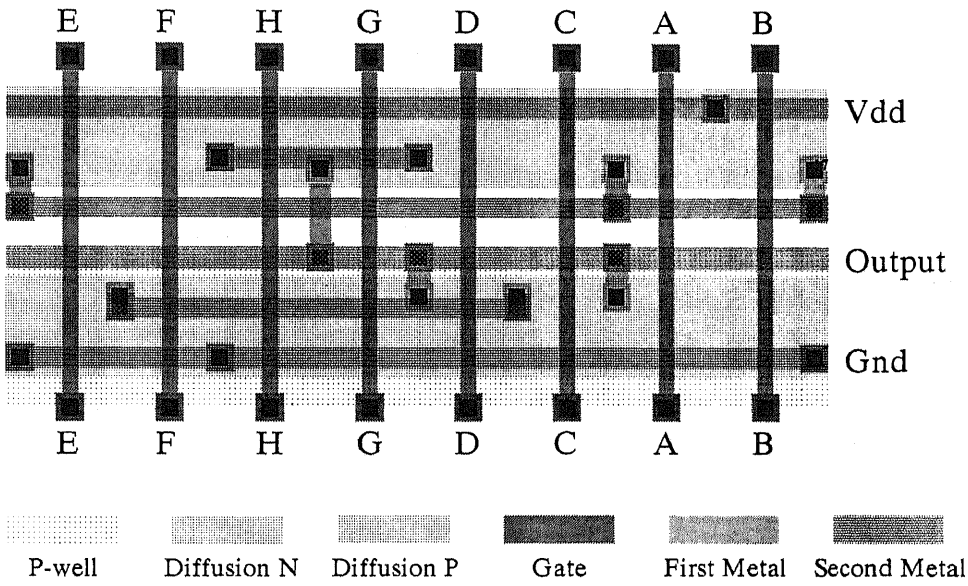


図 5.3: 4 ビット全加算器の 2 次元レイアウト結果 (aspect=1:1)

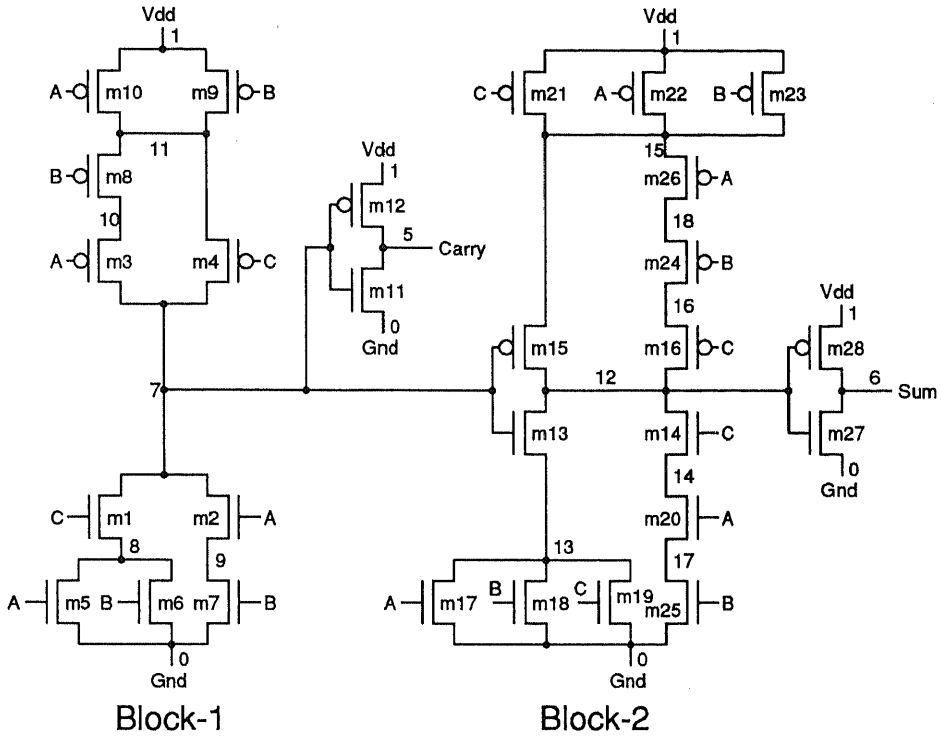


(a) 回路図

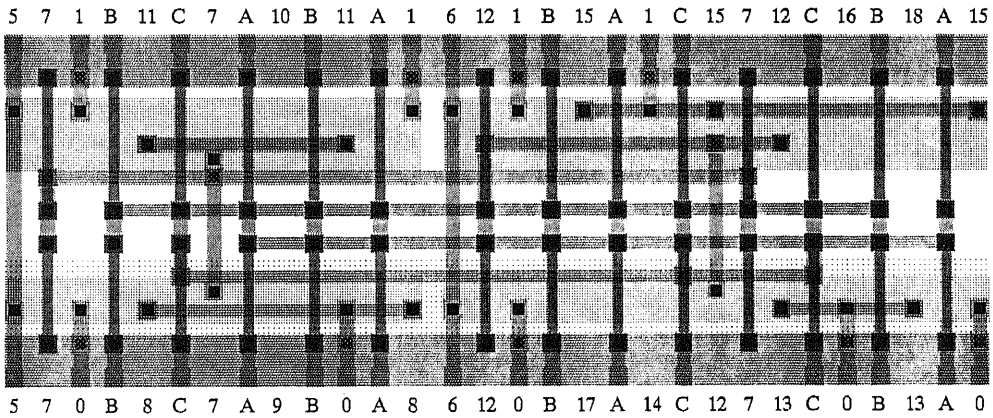


(b) レイアウト結果

図 5.4: Cell 1 の回路図とレイアウト



(a) 回路図 (参考書 [3] より)



(b) レイアウト結果

図 5.5: Cell 2 の回路図とレイアウト

Table 5.1: 直並列回路に対するレイアウト設計の結果

	Cell 1		Cell 2		Cell 3	
	[2]	GmC	[2]	GmC	[2]	GmC
FETs	16		28		10	
P_{min}	1		2		1	
Paths	1	1	3	2	1	1
CPU time†	36.30	0.03	2.00	0.04	0.1	0.01

†) seconds on R-6000/MIPS.

Cell 2 の例題 (図 5.5-a) については、Maziasz らのアルゴリズムにはデュアルなトランジスタからペアを作るという制約があるため、網羅的な探索にもかかわらず、その回路をカバーするパス数は 2 となる解を見つけることができない。**GmC** では、一括処理型トランジスタ ペアリング アルゴリズムを用いて、すべてのトランジスタ ペアが平均的にもっとも多いコンパチブル ペアを持つように選択する。これによって、Cell 2 の回路 (図 5.5-(a)) に対するペアリングの結果では、(m_3, m_2), (m_{10}, m_5), (m_9, m_6) と (m_8, m_7) の 4 つのペアはデュアルなトランジスタ ペアではないが、この 4 つのペアが選択されたため、**GmC** は下限と一致した数のパスが得られた。そのレイアウトの結果を図 5.5-(b) に示す。

GmC も Maziasz らのアルゴリズムも、同じくらいの数のパスについて調べたが、Maziasz らのアルゴリズムより、**GmC** の方がはるか早い時間でパスの探索を完了する。**GmC** が高速探索を実現しているもっとも重要なポイントはコンパチブル マトリクスの導入と思われる。

Maziasz らのアルゴリズムでは、パスの探索はコンポジション ツリーで行なわれる。コンポジション ツリーの 1 つの節点は 1 つの部分グラフであり、その節点にその部分グラフのすべてのカバーを持っている。パス探索は 2 つの兄弟節点にあるカバーのすべての可能な連結を作り、親節点に渡すというような

処理である。2つの兄弟節点がそれぞれ m と n 通りのカバーを持っているとすれば、この2つの節点のカバーを連結すると、$m \times n$ 通り以上の数の可能なカバーが現れる。ここでいう連結とは、コンパチブル関係を調べることと同じ意味である。コンパチブル ペア探索アルゴリズムでは、コンパチブル関係はコンパチブル マトリクスに持っているので、手間が少なくなる。また、Maziasz らのアルゴリズムでは、新しいカバーが現れると、それを保留するため、カバーのコピーが必要となる。このような操作も手間のかかるものである。

この実験から、コンパチブル マトリクスはパス探索において、非常に優れたものであるとわかった。

5.4 非直並列回路のレイアウト設計による GmC の評価

この節では、CMOS トランジスタ回路のモジュール ジェネレータ **GmC** を使って、非直並列回路に対してレイアウト設計を行なう。設計の結果から、**GmC** の2つの核心のアルゴリズム — 一括処理型トランジスタ ペアリング アルゴリズムおよびコンパチブル ペア探索アルゴリズムを評価する。

実験は MOSES — MOS 集積回路モジュール設計システムにあるすべての例題および教科書と文献にある典型的な例題について行なわれた。実験結果の一部を表 5.2 に示す。この表にある回路は次の通りである。

Brg: ブリッジ回路であり、その回路とレイアウト パターンをそれぞれ図 5.6-(a) と図 5.6-(b) に示す。

Add1: MOSES [4] によって合成された1ビット加算器である。その回路とレイアウト パターンをそれぞれ図 5.7-(a) と図 5.7-(b) に示す。

TG26: 参考書 [3] の３１８ページにある例題である。

Xor8: MOSES [4] によって合成された8入力排他論理回路である。

Add4: MOSES [4] によって合成された4ビット桁上げ先見加算器である。

Table 5.2: 非直列回路に対するレイアウト設計の結果

Circuit	Brg	Add1	TG26	Xor8	Add4
FETs	10	28	26	68	140
P_{min}	1	3	3	6	15
Paths	2	3	3	6	15
CPU time†	0.02	0.19	0.04	1.5	18.6

†) seconds on R-6000/MIPS.

この表にある $\mathbf{P_{min}}$ はその回路をカバーするパスの数の下限であり、**Paths** は **GmC** によって見つかったパス数である。CPU 時間にはトランジスタの配置と配線を行なう時間も含まれている。

実験の結果を見ると、非直並列の CMOS 回路についても、**GmC** によって見つかったパスの数はすべて下限に達したことがわかった。

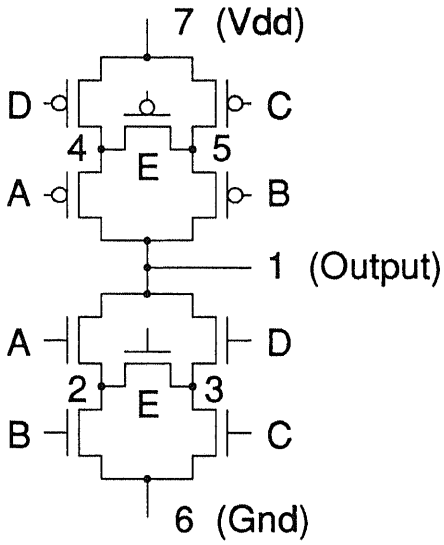
ブリッジ回路については、網羅的な探索を行ったが、それをカバーするパスの数が1である解が存在しないことがわかった。

TG26 回路については、Hwang ら [5] によって、導かれた下限は4であった（TABLE I の case 10 参照）。第4章での定理と推論によれば、この回路の下限は3であって、**GmC** によって、3本パスを見つけられた。

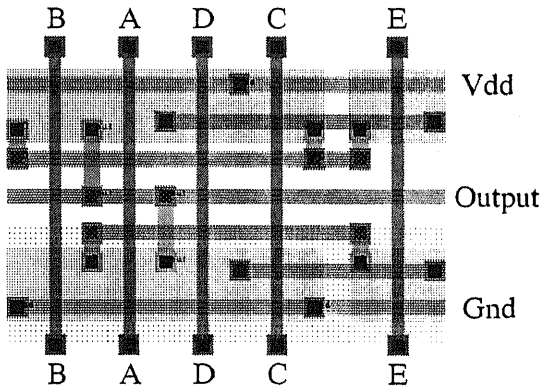
5.5 GmC の評価のまとめ

コンパクトな CMOS 集積回路の設計を支援するため、高速で、高品質なレイアウトが自動設計できるモジュール ジェネレータ **GmC** を開発した。このモジュール ジェネレータに一括処理型トランジスタ ペアリング アルゴリズムとコンパチブル ペア探索アルゴリズムの2つの独特なアルゴリズムを取り入れたことによって、今まで発表されているアルゴリズムの中で、最も優秀なものより、更に短い時間で最適なレイアウトを設計することができる。

数十種類の典型的な CMOS トランジスタ回路に対して、設計と実験をした

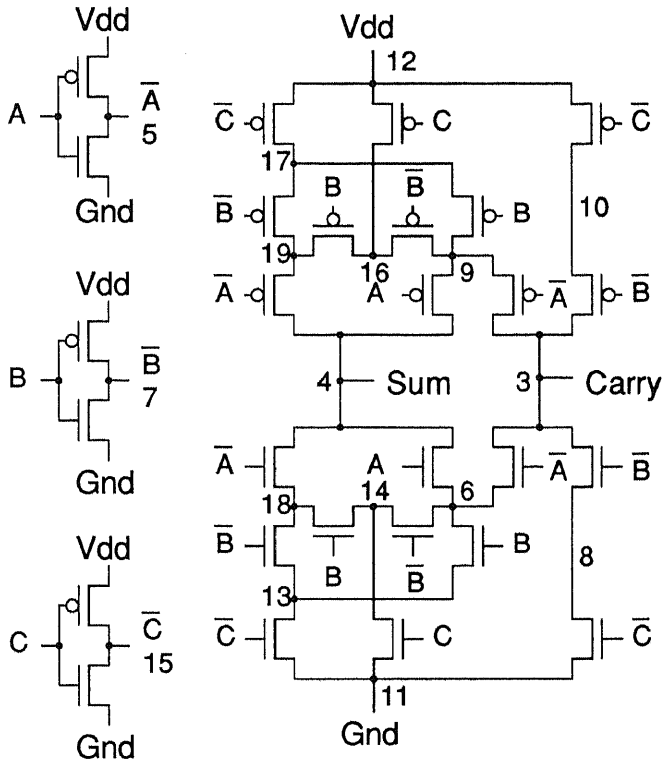


(a) 回路図

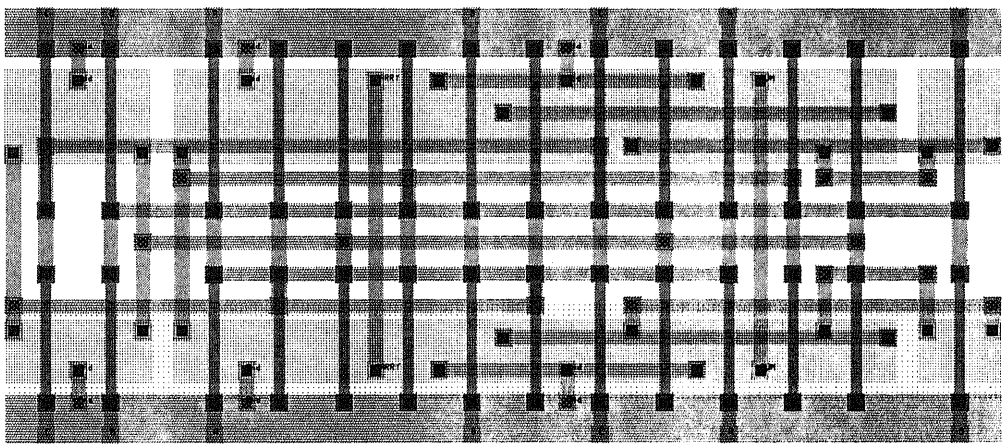


(b) レイアウト結果

図 5.6: Brg 回路とそのレイアウト



(a) 回路図



(b) レイアウト 結果

図 5.7: Add1 回路とそのレイアウト

結果からみると、このモジュール ジェネレータによって求められた解は下限に達したことがわかった。

与えられた CMOS トランジスタ回路をカバーするパス数の最小化問題については求められる解はパス探索アルゴリズムだけではなく、トランジスタのペアリング アルゴリズムにも依存する。つまり、最適ではないトランジスタ ペアリング結果からは、網羅的なパス探索アルゴリズムがあっても、最適な解を見つかることができない。

参考文献

[1] 浅田 邦博 他. *"MOSES: MOS集積回路モジュール設計システム "*. 東京大学出版会, 1991.

[2] R. L. Maziasz and J. P. Hayes. "Layout optimization of static CMOS functional cells". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-9(7):708–719, July 1990.

[3] N.Weste and K.Eshraghian. *"Principles of CMOS VLSI Design"*. MA: Addison-Wesley, 1985.

[4] K. Asada and J. Mavor. "MOSYN: a MOS circuit synthesis program employing 3-way decomposition and reduction based on seven-valued logic". *IEE Proceedings*, 137 Pt E(6):451–461, November 1990.

[5] C.Hwang, Y.Lin Y.Hsieh, and Y.Hsu. "A Fast Transistor-Chaining Algorithm for CMOS Cell Layout". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-9(7):781–786, July 1990.

第 6 章　レイアウト自動設計における信号遅延の削減手法

大規模集積回路の高速化を支援するレイアウト自動設計あるいはパフォーマンス ドリブン レイアウト (Performance Driven Layout) は最近に注目され始めている研究である。代表的な手法として、与えられたタイミング制約によるセル配置 [1, 2, 3] が知られている。これらの手法は、タイミング制約を守ることができるが、信号遅延が最小となるレイアウトを発見することができない。タイミング要求は本来、与えられるものではなく、回路に固有なものである。よって、本研究では、クリティカル パスの探索と遅延時間の解析によってタイミング要求を求め、求めたタイミング要求にしてがって、信号遅延が最小となるようにレイアウトする方法について研究した。また、見つかったクリティカル パスをさらに短縮すれば、より高速な集積回路を設計することができるので、本研究では、モジュール ジェネレータ **GmC** を用いた論理回路の再合成手法と論理回路の物理的な構造を考慮し、論理的に等価である接続を入れ換える手法によるクリティカル パスの短縮について研究した。これらの手法の研究にあたって、集積回路高速化を支援するレイアウト自動設計ツール **RACE** を開発した。

6.1 モジュール ジェネレータ GmC を用いた論理回路の再合成

論理回路の等価変換によって、論理回路の遅延時間を短縮することが可能である。これはよく知られていることであるが、これを取り入れることのできる設計システムは限られている。特に、現在主流となっているセル ライブラリ設計方式の設計システムでは、論理回路の等価変換を行おうとしても、セル ライブラリに登録したものに限ってのみ変換が可能となる。つまり、その探索空間はライブラリに限られている。

ライブラリの制約をなくし、探索空間を広げるため、任意な論理回路に対しても、高速でコンパクトなレイアウトを自動設計することのできるモジュール ジェネレータは必要不可欠なものとなる。第 5 章で説明したモジュール ジェネレータ GmC はこの目的で作られたものである。この節では、このモジュール ジェネレータ GmC を用いた論理回路の等価変換あるいは再合成について、簡単な例で説明する。

まず、次の例について考える。例えば、論理

$$z = \overline{a \cdot x}$$

$$x = \overline{b \cdot c}$$

を書き換えると、

$$z = \overline{\overline{a \cdot b \cdot c}} = \overline{a \cdot (\overline{b} + \overline{c})}$$

となる。これで、2 つ 2 入力 NAND ゲートは 1 つの OAI 複合ゲートに変換されたわけである。ただし、OAI 複合ゲートの $\overline{b}$ と $\overline{c}$ の入力にインバータが付いている。以上の変換を回路図で示すと、図 6.1となる。

図 6.1の右側の OAI 複合ゲートについては、出力ノード z からグランド Gnd まであるいは電源 Vdd から出力ノード z までのパスの中で、最も長いパスは 2 入力 NAND ゲートと同じように、2 つのトランジスタであるため、この OAI 複合ゲートの遅延時間は 2 入力 NAND ゲートとほぼ同じと考えると、

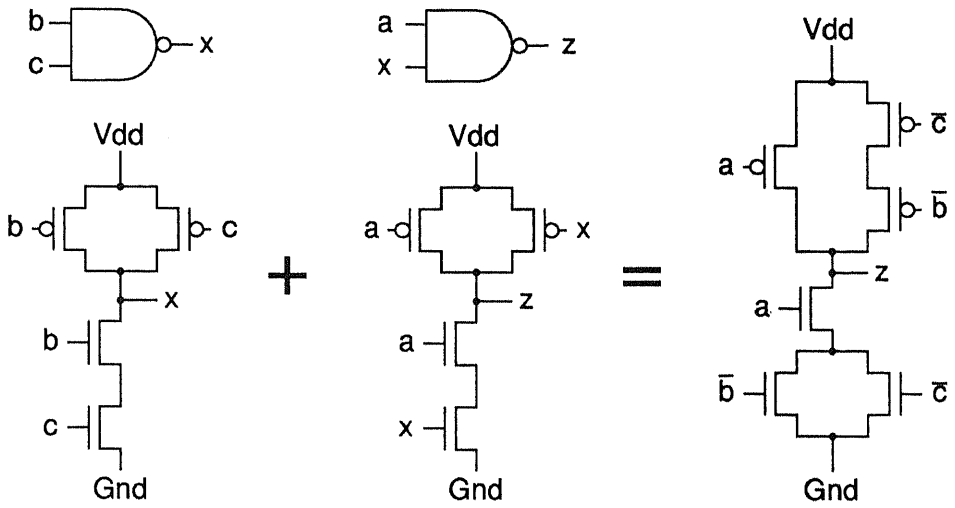


図 6.1: NAND ゲートと NAND ゲートの再合成例

変換前後の回路の遅延時間の差はおよそ 1 つ 2 入力 NAND ゲートと 1 つインバータの遅延時間の差となる。しかし、プロセス テクノロジーによって、OAI 複合ゲートと 2 入力 NAND ゲートの遅延時間の差は 2 入力 NAND ゲートとインバータの遅延時間の差と同じもしくはより大きくなる場合がある。このとき、それぞれの回路をモジュール ジェネレータ **GmC** によってレイアウトに展開した後、シミュレーションによる遅延時間の解析を行なえば、最適な選択が得られる。

論理合成の立場から見ると、リテラル数あるいはトランジスタ数を最小化することは論理合成の 1 つ目標である。よって、図 6.1の OAI 複合ゲートは選択されない可能性が大きい。それは OAI 複合ゲートの入力 $\bar{b}$ と $\bar{c}$ に付いているインバータを含めて、トランジスタの数は 2 つ増えたからである。しかし、トランジスタ数の増加は必ずしも遅延時間の増加になるとは限らない。したがって、モジュール ジェネレータを取り入れることのある論理合成ツールは、より高速な集積回路を設計することができる。

また、図 6.1の OAI 複合ゲートの $\bar{b}$ と $\bar{c}$ の入力に付いているインバータについては、例えば、入力端子 b あるいは c はインバータによって駆動される場

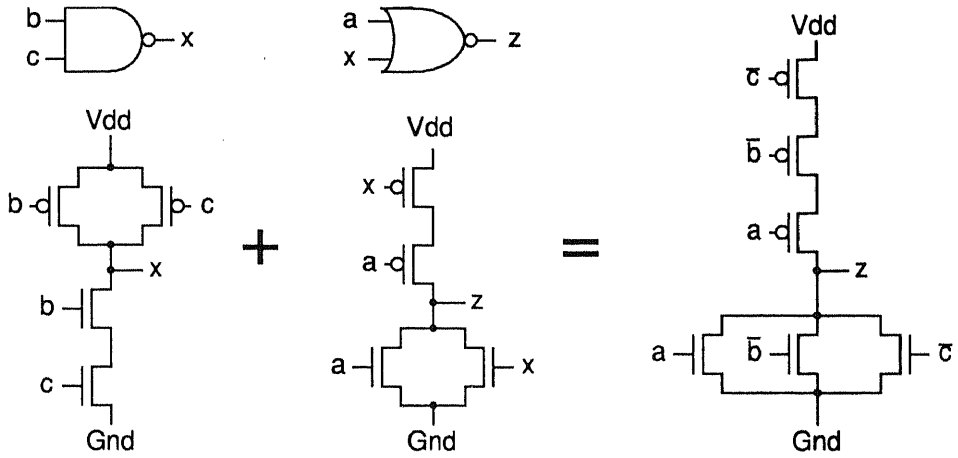


図 6.2: NAND ゲートと NOR ゲートの再合成例

合、OAI 複合ゲートに変換された時、連続するインバータが現れるので、2 つ連続するインバータをキャンセルすることができる。よって、変換後の回路は 1 つインバータ分の遅延時間が増えなければ、採用される場合がある。このような条件つき採用は、モジュール ジェネレータがあるため、可能となる。

NAND ゲートと NAND ゲートの他に、NAND ゲートと NOR ゲート、NOR ゲートと NOR ゲート、NOR ゲートと NAND ゲートの 2 段論理回路を 1 段論理回路に変換する例をそれぞれ図 6.2, 6.3, 6.4 に示す。

6.2 論理回路の再合成によるクリティカル パスの最小化

回路のスピードはクリティカル パスによって決まる。**クリティカル パス**とは信号が通らねばならない論理的道筋のうち、最も時間のかかるものである。よって、クリティカル パスを短縮することは回路の信号遅延の改善において、重要なポイントとなる。この節では、実際の論理回路に対して、前節に述べた論理回路の再合成手法によるクリティカル パスの最小化処理を示す。

図 6.5は Parthenon （パルテノン） [4] で合成した Counter という論理回路から、**RACE** によって求めたクリティカル パスを示すものである。クリ

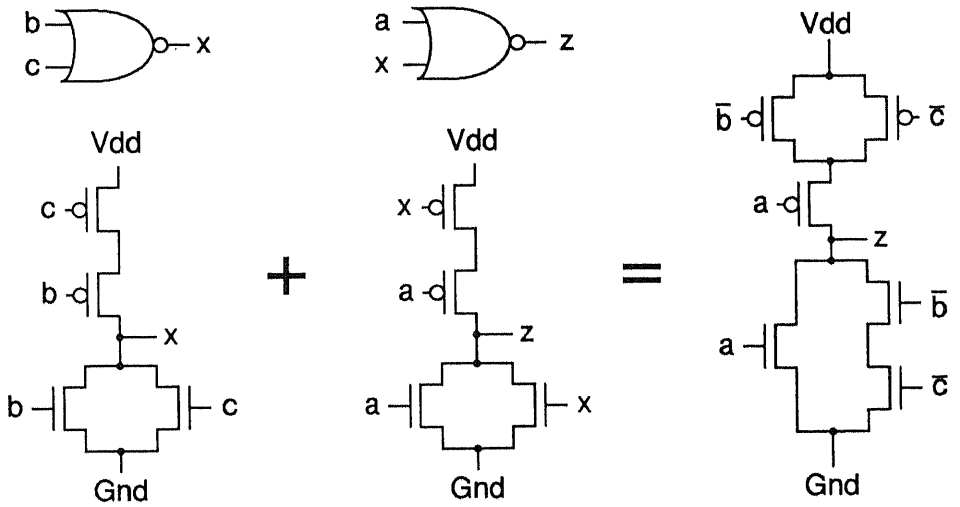


図 6.3: NOR ゲートと NOR ゲートの再合成例

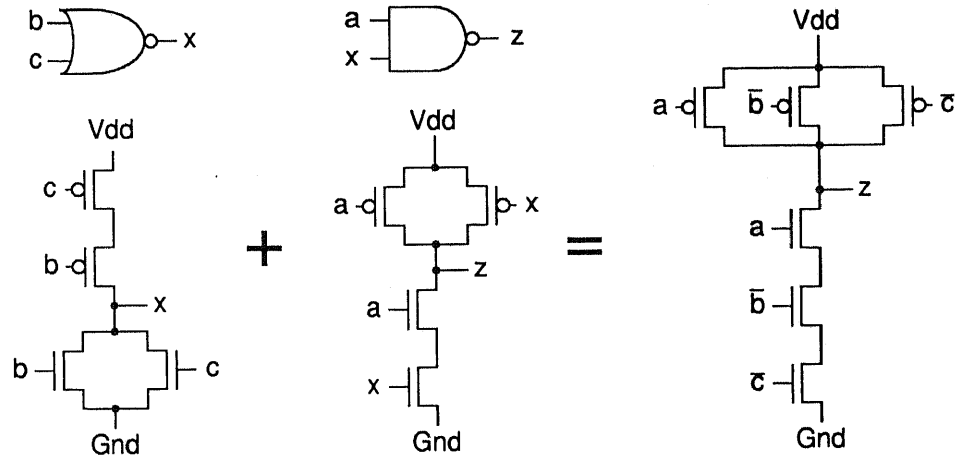


図 6.4: NOR ゲートと NAND ゲートの再合成例

ティカル パスは網かけで示されている。論理回路の再合成によるクリティカル パスの短縮は次の処理からなる。

1. 論理ゲート i の前後にあるゲートを 1 つの**論理ゲートのペア**にする。例えば、図 6.5のゲート g_{24} ゲートに対して、$\{g_{28}, g_{24}\}$ と $\{g_{24}, g_{27}\}$ の 2 組の論理ゲートのペアを作る。この 2 組の論理ゲートのペアとも g_{24} が含まれているので、**排他的論理ゲート ペア**と言うことにする。

2. それぞれの論理ゲート ペアに対して、前節に述べた再合成処理を行なう。

3. 遅延時間の改善幅が最も高い回路を優先的に採用する。元の 2 つの論理ゲートをネットリストから削除する。削除された論理ゲートの排他的論理ゲート ペアに対応する再合成回路も削除する。例えば、2 組の排他的論理ゲート ペア $\{g_{28}, g_{24}\}$ と $\{g_{24}, g_{27}\}$ に対して、それぞれ複合ゲート A と B が再合成されたとして、複合ゲート A を採用することによって、複合ゲート B は削除される。

以上の処理の結果、図 6.5の論理ゲート ペア $\{g_{71}, g_{70}\}$ と $\{g_{24}, g_{27}\}$ は変更されることになる。これで論理ゲートの再合成によるクリティカル パスの短縮の第 1 フェーズが終了する。第 1 フェーズの処理によって、クリティカル パスは 1.15 個のインバータ分の遅延時間が短縮された。

クリティカル パスが短縮されることによって、他のパスがクリティカル パスになる場合がある。例えば、図 6.5のクリティカル パスが短縮された後、クリティカル パスは図 6.6に示すようになった。このため、上述の論理回路の再合成によるクリティカル パスの短縮処理が再び実施される。その結果、論理ゲート ペア $\{g_{23}, g_{22}\}$ が変更されることになる。これで論理ゲートの再合成によるクリティカル パスの短縮の第 2 フェーズが終了する。第 2 フェーズの処理によって、クリティカル パスは 1.68 個のインバータ分の遅延時間が短縮された。

以上のように、論理回路の再合成によるクリティカル パスの短縮処理を繰

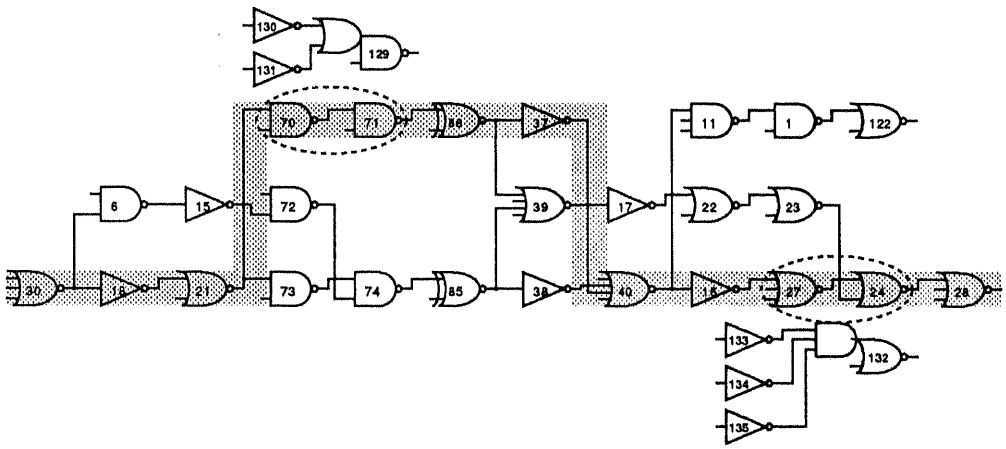


図 6.5: 論理回路の再合成によるクリティカル パスの短縮（フェーズ１）

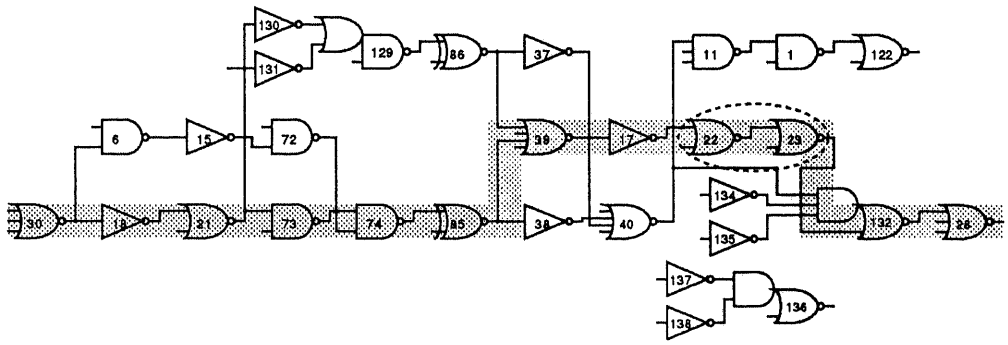


図 6.6: 論理回路の再合成によるクリティカル パスの短縮（フェーズ２）

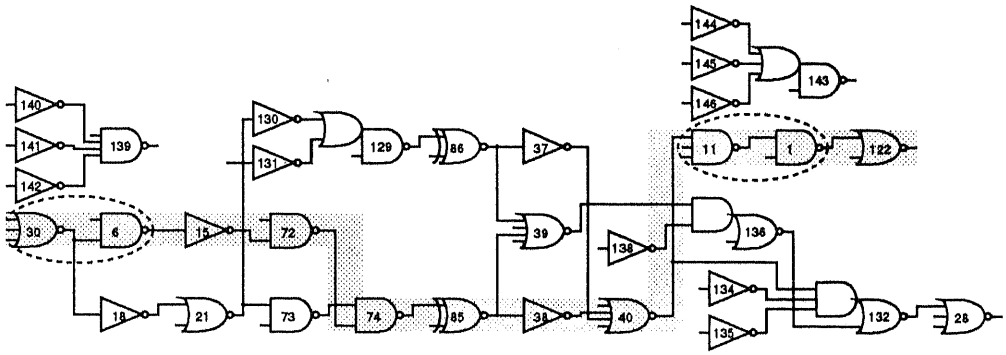


図 6.7: 論理回路の再合成によるクリティカル パスの短縮（フェーズ 3）

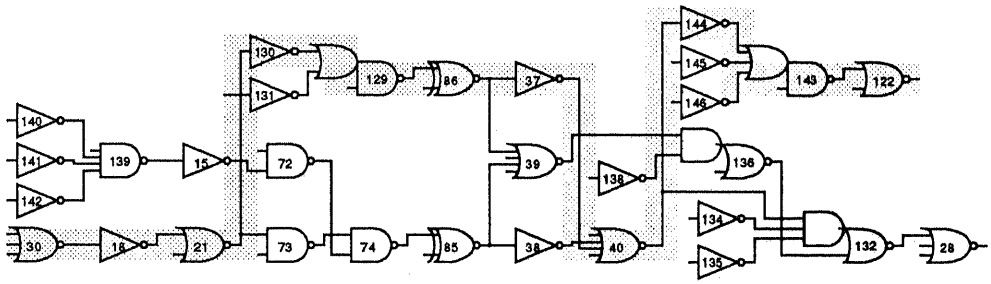


図 6.8: 論理回路の再合成によるクリティカル パスの短縮（結果）

り返して行なうと、クリティカル パスは最小値に達する。この Counter 回路の第 3 フェーズ処理および最後の結果をそれぞれ図 6.7と図 6.8に示す。

以上をまとめて、論理回路の再合成によるクリティカル パスの最小化は以下の処理からなる。

1. 論理ゲートのネットリストからクリティカル パスを探索する。

2. クリティカル パスに対して、論理回路の再合成によるクリティカル パスの短縮処理を行なう。

3. 次のクリティカル パスを探索する。見つかったクリティカル パスが前と同じパスであれば、終了する。それ以外ならば、2 へ戻る。

論理回路の再合成によって、トランジスタ数あるいは面積が増える場合があ

る。このため、**RACE**では、合成した回路の採用基準が設計者によって選択できるようにする。その基準は次の3つがある。

1. 遅延時間基準。
2. 面積 - 遅延時間積基準。
3. 面積基準。

6.3 論理ゲート入力端子の再割り当てによるクリティカル パスの最小化

論理上等価であるゲートの入力端子を入れ換えることによって、信号遅延を短縮することが可能である [5]。これはよく知られていることであるが、入力端子の割り当てまで考える論理合成は少ないのが実情のようである。その理由は論理設計の段階では論理ゲートが物理的な構造を持っていないからである。よって、レイアウト設計において、物理的な構造が回路の動作に与える影響について考えなければならない。

参考書 [5] によれば、入力端子の割り当ての一般的なガイドラインとしては、最も遅く到着する信号が入るトランジスタを、ゲートの最も出力に近いところに配置することである。

最も出力に近いトランジスタと最も出力に遠いトランジスタから出力までの遅延時間を回路シミュレータ SPICE で測った結果が図6.9と図6.10にまとめられている。すべての遅延時間はインバータの遅延時間で正規化されたものである。“nand-min” と “nor-min” はそれぞれ NAND ゲートと NOR ゲートの出力に最も近いトランジスタからの信号遅延であり、“nand-max” と “nor-max” はそれぞれ NAND ゲートと NOR ゲートの出力に最も遠いトランジスタからの信号遅延である。

本研究では、論理ゲート入力端子の再割当によるクリティカル パスの最小化手法として、次の2つ方法について研究した。

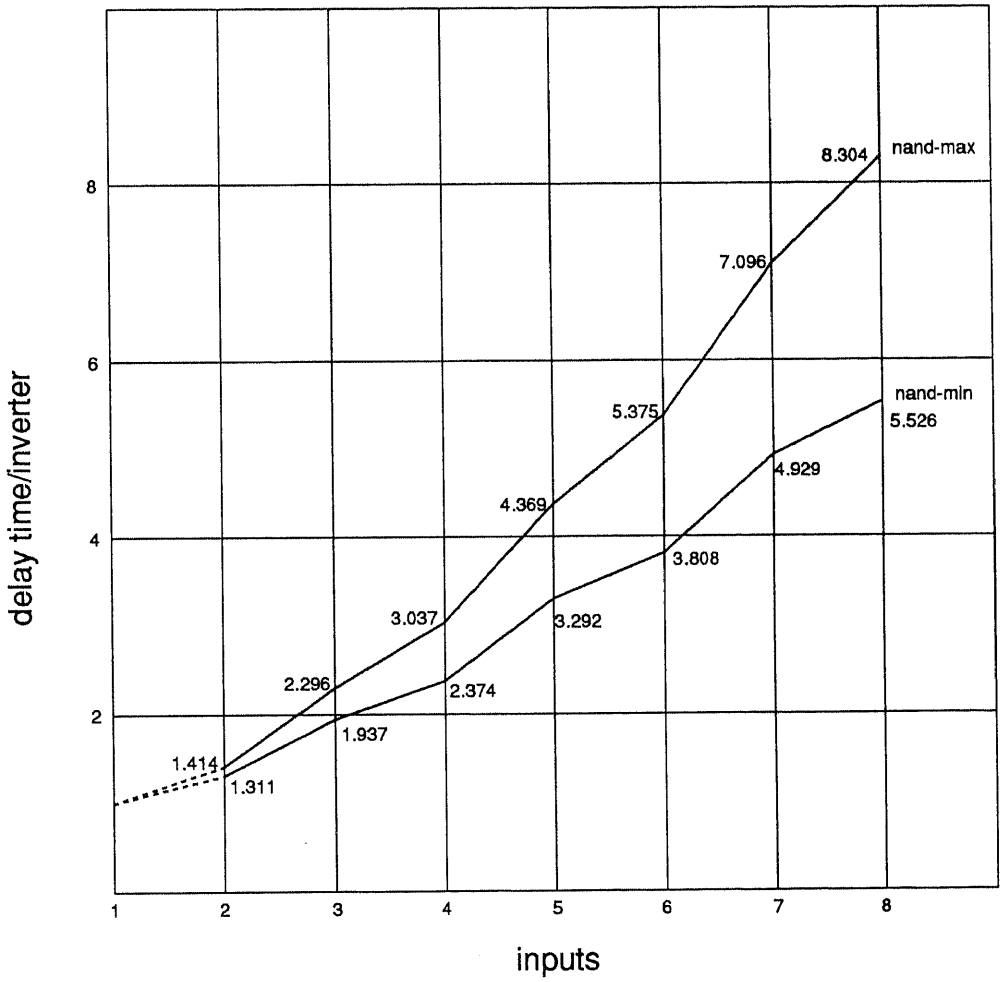


図 6.9: NAND ゲートの入力端子と遅延時間の関係

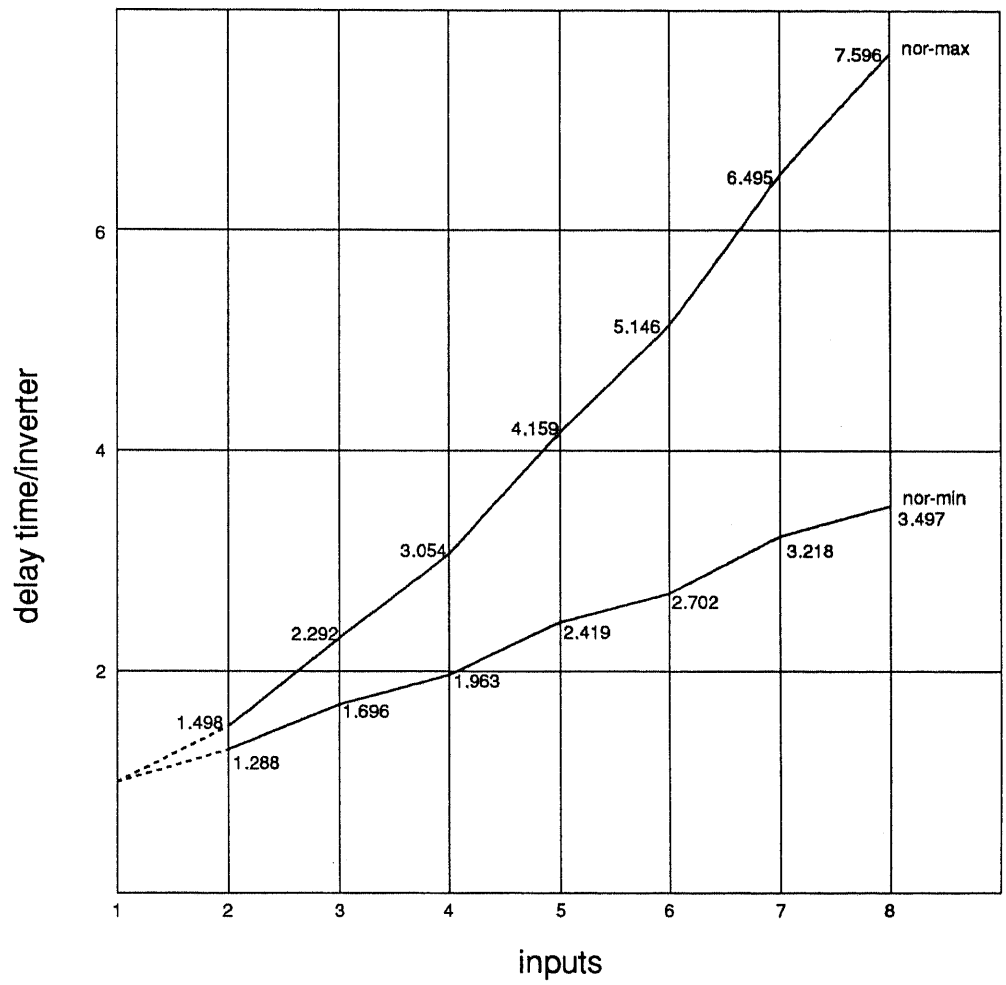


図 6.10: NOR ゲートの入力端子と遅延時間の関係

1. クリティカル パスにある論理ゲートの入力信号線を最も出力に近い入力端子に割り当てる手法。

2. 時間解析によって、論理ゲートに到達する時間順に基づく入力端子を割り当てる手法。

前者をクリティカル パス優先割り当て手法、後者を時間解析割り当て手法と呼ぶことにする。

6.3.1 クリティカル パス優先割り当て手法

クリティカル パス優先割り当て手法では、遅延時間のトップ m 本パスに対して処理するため、高速で解を求めることができる。m は、設計者によって与えられるものであるが、m が小さければ小さいほど計算量が少なくなり、m が大きければ大きいほど最適解が得られる。クリティカル パス優先割り当てによるクリティカル パスの最小化は次の処理からなる。

1. 論理ゲートのネットリストから、遅延時間が上位 m 本のパスを探す。トップのパスはクリティカル パスとなる。

2. クリティカル パスにある入力信号線を最も出力に近い入力端子に割り当てる。つまり、クリティカル パスにある入力信号線と最も出力に近い入力端子につながっている入力信号線を入れ換える。ただし、両端子は論理的に等価でなければならない。

3. 上位 m 本パスの遅延時間の更新を行ない、パスの並び順序を再ソートする。

4. クリティカル パスの位置が変らなければ、終了する。クリティカル パスの遅延時間が増加したら、元通りにして終了する。それ以外ならば、2 へ戻る。

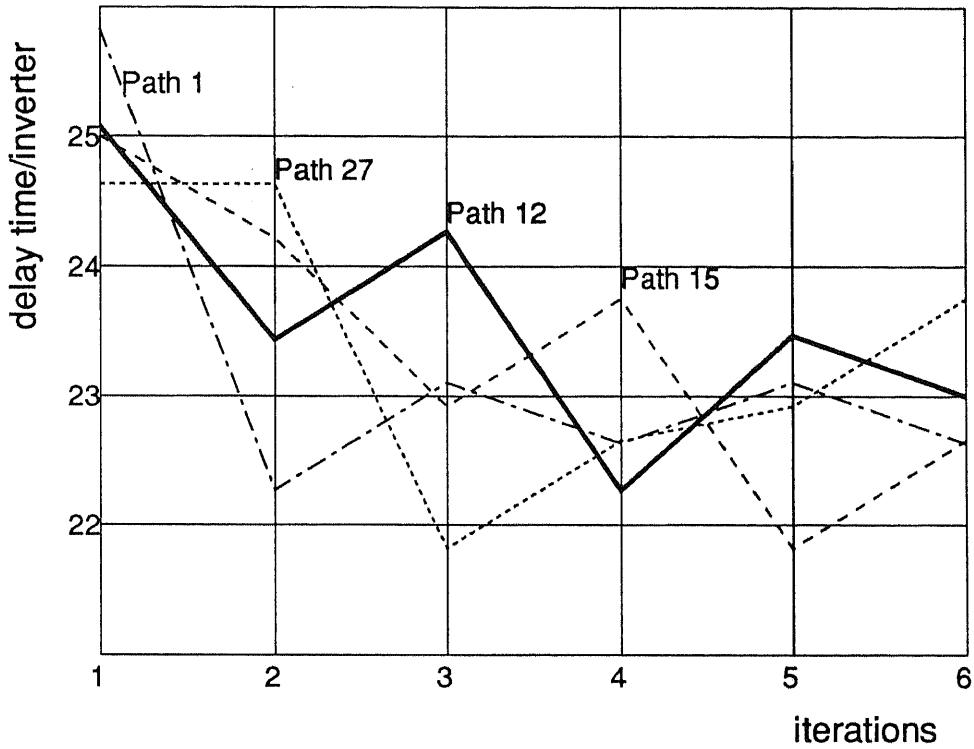
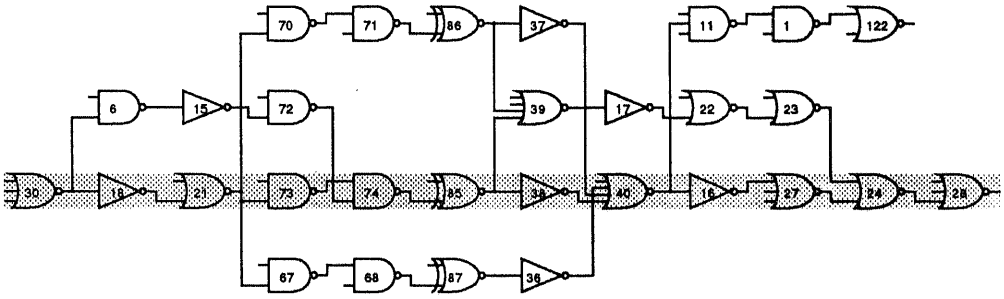


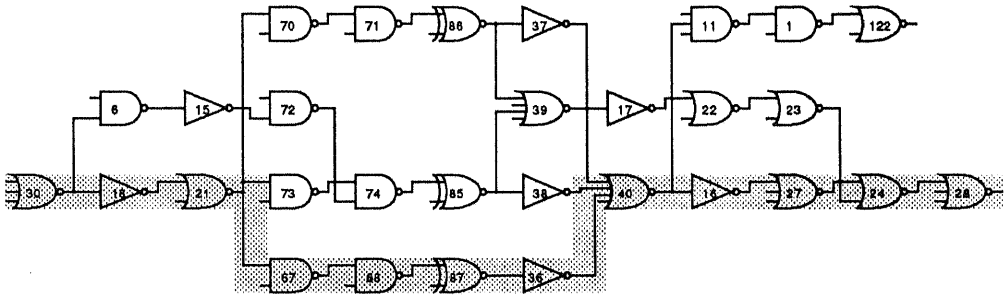
図 6.11: クリティカル パス優先割り当てによるクリティカル パスの推移

ここでは、パスの遅延時間はそのパスにある論理ゲートの遅延時間の合計とする。パスにおける信号の伝搬は、回路の入力あるいはフリップ フロップのクロック信号から、回路の出力あるいはフリップ フロップの入力までとする。

図 6.11は、 Parthenon （パルテノン） [4] で合成した Counter 回路に対して、論理ゲート入力端子の再割り当てによるクリティカル パスの最小化処理の各段階において、1度クリティカル パスとして現れたパスの遅延時間を示すものである。パスの番号は最初の遅延時間順である。この図からわかるように、このクリティカル パスの最小化処理によって、クリティカル パスは Path 1, Path 27, Path 12, Path 15, Path 12 というように下がっていったが、5回目の処理が終わったとき、クリティカル パスの遅延時間が増加したため、処理を終了する。これで4回目の処理が終わったときの結果が最終結果となる。



(a) 処理前



(b) 処理後

図 6.12: 論理ゲート入力端子の再割り当てによるクリティカル パスの最小化処理前後の論理回路図

図 6.12は、論理ゲート入力端子の再割り当てによるクリティカル パスの最小化処理前後の論理回路図を示すものである。ただし、論理ゲートの入力端子は出力に近い方が上にあるとする。クリティカル パスは網かけで示されている。

6.3.2 時間解析割り当て手法

時間解析割り当て手法では、遅延時間の解析によって、信号が論理ゲートに到達する時間を求め、求めた時間順によって入力端子を割り当てる。例えば、図 6.13の 4 入力 NAND ゲート入力端子 a_1, a_2, a_3, a_4 から出力 z までの遅延時間はそれぞれ 1.1, 1.2, 1.3, 1.4 であるとき、4 つ入力信号 n_1, n_2, n_3, n_4 が

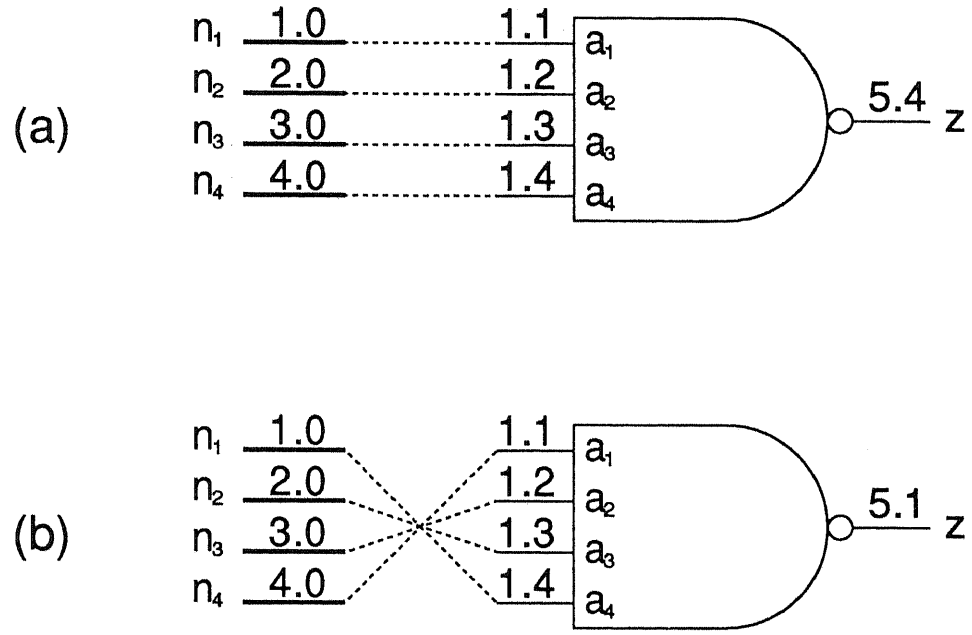


図 6.13: 4 入力 NAND ゲート入力端子の最適割り当て

このゲートに到達する時間はそれぞれ 1.0, 2.0, 3.0, 4.0 とわかったとき、図 6.13-(b) のように遅く到達する信号を遅延時間の短い入力端子に割り当てるならば、出力までの遅延時間は最小となる。よって、遅延時間解析による論理ゲート入力端子の割り当てが最適となる。

しかし、遅延時間解析による論理ゲート入力端子の割り当てを行なうため、各ノードに信号到達する時刻を記録する時間表が必要となる。また、1 つの論理ゲートについて入力端子の割り当てを行なったら、時間表を更新しなければならない。この更新は回路のすべての論理ゲートに対して行なわれるので、クリティカル パス優先割り当て手法より計算量が多くなる。

割り当て処理済みの論理ゲートに対しても、時間表の更新によって、再処理する必要がある場合がある。このような再処理をなくすため、論理ゲート入力端子の割り当ては入力段から処理していくことにした。つまり、信号伝搬に沿って処理していく。

図 6.14は Parthenon （パルテノン） [4] で合成した Counter 回路のすべ

ての論理ゲートに対して、時間解析による論理ゲート入力端子の最適割り当て処理の前後で、一部ノードにおいて、信号到達時間の変化を示すものである。この図から、各ノードにおいて信号遅延が改善されたことがわかった。また、この手法によるクリティカル パスの最小化の結果は、上述のクリティカル パス優先割り当て手法と一致したことが確認された。信号が図6.14の７５番目のノードに到達する時間はクリティカル パスに伝搬する時間である。

クリティカル パス以外のパスにおいて、信号遅延の改善は意味が少ないので、時間解析割り当て手法では、クリティカル パスにある論理ゲートに限定して処理を行なう。クリティカル パス以外の論理ゲートに対しては、消費電力を減らす時に考える。消費電力について次の章で議論する。時間解析割り当て手法は次の処理からなる。

1. 信号が各ノードに到達する時間を求め、時間表を作成する。

2. 信号が最も遅く到達するノードから、クリティカル パスを探索する。

3. クリティカル パスの入力段から割り当て処理を行なう。つまり、論理的等価である論理ゲート入力端子を出力端子までの遅延時間の小さい順で並べ、それらの入力端子に入る信号線を到達時間の大きい順で並べ、信号線の順序と入力端子の順序によってネット リストを変更する。

4. 時間表を更新する。クリティカル パスの最終段の論理ゲートの処理が終ったら、次のステップへ進む。それ以外ならば、３へ戻る。

5. 以上の処理によって、信号が最も遅く到達するノードと時間が変らなければ、終了する。それ以外ならば、ステップ２へ戻る。

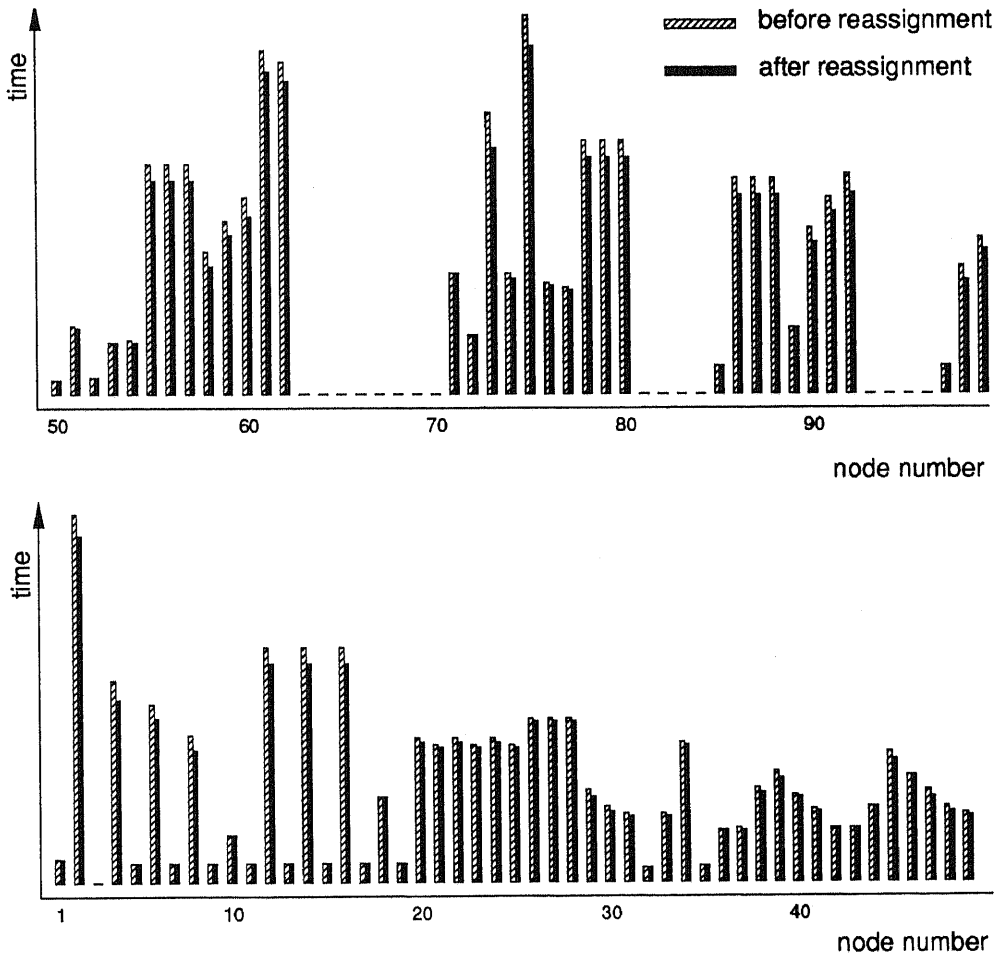


図 6.14: 時間解析による論理ゲート入力端子の最適割り当て処理前後で、信号が各ノードに到達する時間の改善

6.4 信号遅延を考慮した大規模集積回路のレイアウト自動設計手法

大規模集積回路のレイアウト自動設計に関する研究は次の 2 種類にわけることができる。

1. レイアウト面積と計算時間を減らすことに着目する研究。

2. 信号遅延に着目する研究。

前者では、セル間の接続関係、セルの寸法、配線の混雑度などによってセルを配置する [6, 7, 8, 9, 10, 11, 12]。後者では、信号遅延が最小となるようにセルを配置する。後者に関する従来の研究は、与えられたタイミング制約によってセルを配置する [1, 2, 3] というパフォーマンス ドリブン レイアウト (Performance Driven Layout) である。このタイミング制約手法は、タイミング制約を守ることができるが、信号遅延が最小となるレイアウトを発見することができない。タイミング要求は本来、与えられるものではなく、回路に固有なものである。よって、本研究では、クリティカル パスの探索と遅延時間の解析によってタイミング要求を求め、求めたタイミング要求にしてがって、信号遅延が最小となるようにレイアウトする方法について研究した。

6.4.1 レイアウト モデル

ここで考えるレイアウト モデルでは、すべてのセルは M 行に配置される。ただし、その真中の 1 行はクリティカル パスとする。M は次のように決められる。

1. 設計者による直接指定。設計者は $M-1$ 本の出力信号線を指定することができる。

2. 設計者による間接指定。設計者は レイアウトのアスペクト レシオを指定することができる。この場合、M はそれぞれのセルの面積によって決められる。

3. 設計者の指定がない場合、$M = m + 1$ とする。ただし、m は回路の出力信号線の数である。

6.4.2 信号遅延を考慮したレイアウト手法

信号遅延を考慮したレイアウト手法の基本的な考え方は、駆動するセルと駆動されるセルを近くに配置することによって、配線による遅延を最小にすることである。クリティカル パスは最初に配置されることによって、回路全体の遅延時間が最小となることを確保する。出力段にあるセルも最初に配置される。位置が決まったセルは既配置セルと呼ばれ、それ以外のセルは未配置セルと呼ばれる。本レイアウト手法では、既配置セルを駆動する未配置セルの中で、信号が最も遅く出すものは既配置セルの近くに配置される。この処理の詳細を次に示す。

1. クリティカル パスにあるすべてのセルを $k = [M/2]$ 番目の行 Q_k に入れる。

2. 列番号 $j = 1$ とし、出力段にある m 個セルを $Q_i(i = 1, \cdots, k-1, k+1, \cdots, M)$ の最初のセルとする。設計者による指定がある場合、指定された信号線を駆動するセルが最初のセルとなる。

3. 未配置セルがなくなったら、処理を終了する。

4. 列番号 $j = j + 1$ とし、$Q_i(i = 1, \cdots, M)$ にあるすべてのセルに入る信号線を S に入れる。その順序は、信号が到達する時間の遅い順とする。

5. S にある信号線を駆動する最初の M 個の未配置セルを探して、$Q_i(i = 1, \cdots, M)$ に j 番目のセルとして、割り当てる。ただし、j がクリティカル パスの長さより小さいとき、最後の M 番目の未配置セルを処理しない。これで、M 個あるいは $M - 1$ 個未配置セルは同時に確定される。割り当て処理を次に説明するが、割り当て処理が終わったら、ステップ3へ戻る。

このような配置では、ある時刻に動作するセルは一定の領域にあるという特徴があり、そして、この領域は時間とともに出力へ向かっていく。

次に、未配置セルの割り当て処理について説明する。M 個未配置セル $g_u(u = 1,\cdots,M)$ を $Q_i(i = 1,\cdots,M)$ に j 番目のセルとして、割り当てるとき、未配置セルに駆動される既配置セルがどの Q_i にあるかを調べればよいが、1つ未配置セルに駆動される既配置セルが複数である場合があれば、複数未配置セルに駆動される既配置セルが同じ Q_i に集中している場合もある。よって、次の割り当て処理が必要となる。

1. 駆動テーブルを作る。テーブルの縦に未配置セル g_u を置き、横に Q_i を置く。g_u に駆動されるセルが Q_i にあれば、テーブルの u 行、i 列に "D" マークを付ける。それ以外のところに "–" マークを付ける。クリティカルパス Q_k の j 番目のセルが存在する時、未配置セルの前の $M-1$ 個を取るとし、駆動テーブルは $M-1$ 行、$M-1$ 列となる。

2. 駆動テーブルの行あるいは列で "D" マークの数が1個だけの行と列を探す。見つかった場合、その行の未配置セルをその行に "D" マークが付いている列の Q_i に割り当てる。割り当て処理した行と列を駆動テーブルから除去する。これによって、駆動テーブルの行（列）にある "D" マークの数は0か2以上となる。

3. 駆動テーブルの最初の行から "D" マークを探す。見つかった場合、その行の未配置セルを、その行から最初に見つかった "D" マークが付いている列の Q_i に割り当てる。割り当て処理をした行と列を駆動テーブルから除去する。これによって、駆動テーブルの行（列）の "D" マークの数は0だけとなる。

4. 最後に、駆動テーブルの最初の行から割当を行なう。つまり、対角線の割り当てとなる。

6.5 レイアウト自動設計ツール RACE による設計と討論

RACE は大規模集積回路の高速化を支援するレイアウト自動設計ツールである。**RACE** の入力は EDIF フォーマット†で記述された論理回路のネットリストである。**RACE** の出力は TIG フォーマット [13] のセル配置であり、グラフィクス エディタ **GeX** で見ることができる。

この節では、Parthenon （パルテノン） [4] で合成した回路を例題として、**RACE** によるレイアウト設計の結果を示した。

図 6.15は、パルテノンで合成した Counter 回路に対して、信号遅延を考慮したセル配置の処理結果を示すものである。セルは矩形で表され、第 5 章に述べたモジュール ジェネレータ **GmC** で生成されたものである。セル内の数字はそのセルの番号である。セルの下にある数字はそのセルのタイプ‡を示すものである。これらの番号は **RACE** によって与えられるものである。この Counter 回路は 8 つの出力があり、クリティカル パスを含めて、9 行となる。真中の行に配置されるのはクリティカル パスである。

図 6.16は論理ゲート入力端子再割り当てによるクリティカル パスの最小化処理した後の、信号遅延を考慮したセル配置の結果を示すものである。この配置と図 6.15と比較すると、クリティカル パスが変わったことがわかった。また、一部のセルの位置の変化は、論理ゲート入力端子の再割り当てによって、各信号線に到達する時間が変わったことを示している。

図 6.17は論理回路の再合成によるクリティカル パスの最小化処理した後の、信号遅延を考慮したセル配置の結果を示すものである。この配置と図 6.15, 6.16と比較すると、セルの位置が変わっただけではなく、新しいセルも現れた。番号が１２９以上のセルは新しいセルである。セルのタイプは 9 種類から１７種類以上増えた。新しいタイプのセルがすべて採用されたわけではないので、数字

†EDIF は CAD ツール間、製造機器などの間での設計データの標準フォーマットである。EIA （米国工業会） が RS548 として登録している。

‡ここで言うセルは、 Parthenon では、インスタンス (instance) と呼ばれるものである。ここで言うセルのタイプは、 Parthenon では、ライブラリに登録しているセルのことである。

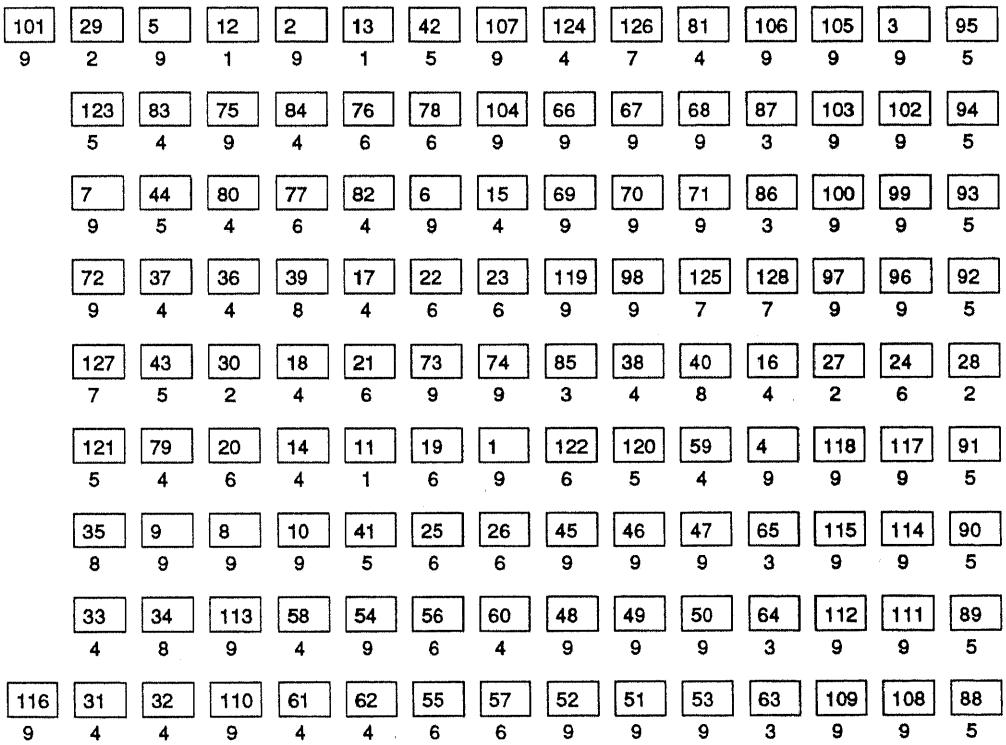


図 6.15: 信号遅延を考慮したセル配置（クリティカル パス最小化前）

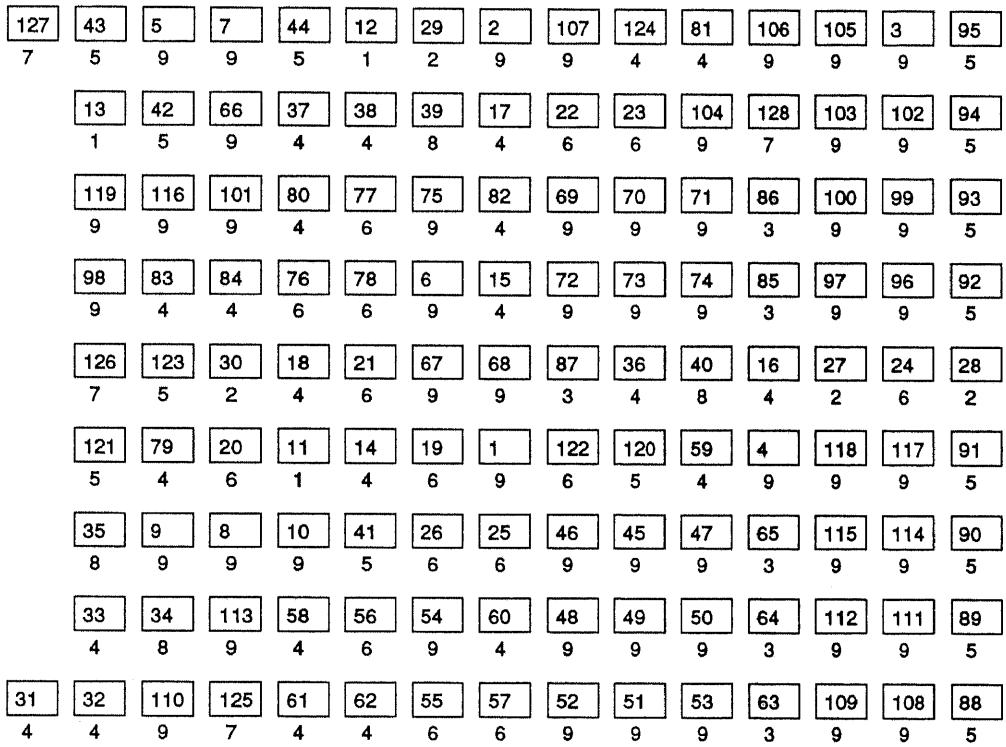


図 6.16: 信号遅延を考慮したセル配置（論理ゲート入力端子の最適割り当て処理後）

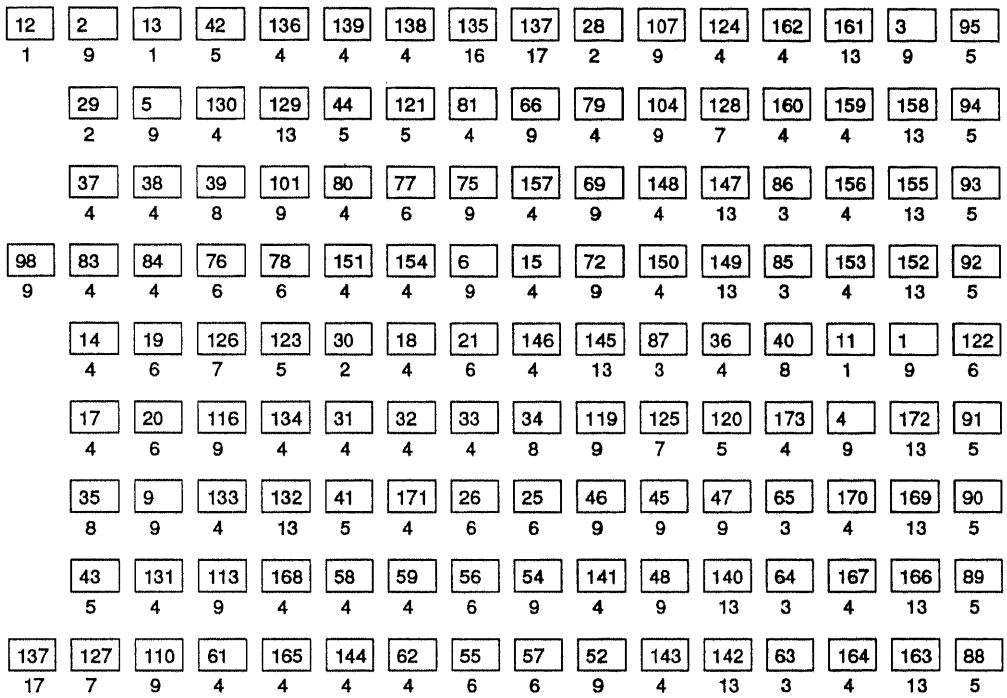


図 6.17: 信号遅延を考慮したセル配置（論理回路の再合成処理後）

は不連続である。

Parthenon にある他の例題について、**RACE** によって設計した結果を表 6.1に示す。この結果から見ると、すべての回路の信号遅延が改善されたことがわかる。改善幅は 10% 未満であるが、それには 2 つの理由がある。(a) Parthenon によって得られた論理回路はすでに Parthenon にあるシンセサイザ (sflexp) とオプティマイザ (opt_map) によって最適化されたものである。(b) **RACE** の論理回路の再合成処理は前後 2 段の論理回路に限定される。

6.6 レイアウト自動設計における信号遅延の削減手法のまとめ

レイアウト自動設計における信号遅延の削減手法について研究した結果では、モジュール ジェネレータ **GmC** を用いた論理回路の再合成手法と論理回路の物理的な構造を考慮した、論理的に等価である接続を入れ換える手法による信

Table 6.1: RACE の設計結果

Circuit	Gates	Nodes	Speed up
counter	128	146	8.1%
multi-16	2372	2404	5.5%
seg	44	48	5.3%
sft-32	646	685	1.0%
bool	21	32	0.7%

号遅延の削減は効果のある方法とわかった。この2つの手法を用いて、Parthenonで最適化された回路に対して、信号遅延は、さらに削減されることを示した。

また、信号遅延を考慮したレイアウト手法について研究し、信号遅延が最小となるようにレイアウトする手法を提案した。この新しい手法では、セルを配置するためのタイミング要求は、設計者によって与えられるものではなく、クリティカル パスの探索と遅延時間の解析によって回路から求められるものとして、求められたタイミング要求にしてがって、駆動すると駆動されるセルを近くに配置することにより、配線による遅延を削減する。

参考文献

[1] K. Nakao, O. Kitada, M. Hayashikoshi, K. Okazaki, and Y. Tsujihashi. "A High Density Datapath Layout Generation Method Under Path Delay Constraints". *Proceedings of The IEEE 1993 Custom Integrated Circuits Conference (CICC)*, pages 9.5.1–9.5.5, May 1993.

[2] T. Gao, P. M. Vaidya, and C. L. Liu. "A New Performance Driven Placement Algorithm". *IEEE International Conference on Computer-Aided Design, ICCAD-91, Digets of Technical Papers*, pages 44–47, November 1991.

[3] M. A. B. Jackson and E. S. Kuh. "Performance Driven Placement of Cell Based ICs". *Proceedings of the 26th ACM/IEEE Design Automation Conference (DAC)*, pages 370–375, June 1989.

[4] NTT Network Information System Laboratory. *"Parthenon User's Manual"*. 1989.

[5] N.Weste and K.Eshraghian. *"Principles of CMOS VLSI Design"*. MA: Addison-Wesley, 1985.

[6] C.Sechen and A.Sangiovnni-Vincentelli. "TimberWolf3.2: A new standard cell placement and global routing package". *Proceedings of the 23rd ACM/IEEE Design Automation Conference (DAC)*, pages 432–439, June 1986.

[7] T. K. Ng, J. Oldfield, and F. H. Kirsch. "Improvements of a mincut partition algorithm". *IEEE International Conference on Computer-Aided Design, ICCAD-87, Digets of Technical Papers*, pages 470–473, November 1987.

[8] E.S. Kuh and T. Ohtsuki. "Recent advances in VLSI layout". *Proceedings of the IEEE*, 78(2):237–259, February 1990.

[9] J. Frankle and R. M. Karp. "Circuit placements and cost bounds by eigenvector decomposition.". *IEEE International Conference on Computer-Aided Design, ICCAD-90, Digets of Technical Papers*, pages 414–417, November 1990.

[10] J. P. Blanks. "Near-optimal placement using a qaudratic objective function.". *Proceedings of the 22nd ACM/IEEE Design Automation Conference (DAC)*, pages 609–615, June 1985.

[11] 伊達 博 , 林 照峯. "ニューラルネットによるLSIモジュール配置手法". 電子情報通信学会論文誌, J73-A(10):1641–1647, 1 0月 1990.

[12] J. M. Kleinhans, G. Sigl, F. M. Johannes, and K. J. Antreich. "GORDIAN: VLSI placement by quadratic programming and slicing optimization". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-10(3):356–365, March 1991.

[13] 浅田 邦博 他. *"MOSES: MOS集積回路モジュール設計システム "*. 東京大学出版会, 1991.

第 7 章　レイアウト自動設計における消費電力の削減手法

集積回路の高速化、特に、 RISC アーキテクチャの導入による、回路の動作周波数が増加による、消費電力の増加は大きな問題となっている。低消費電力化に関する研究はすでに各設計、製造部門に繰り広げている。消費電力は回路の物理的な構造に関係する部分が多いので、消費電力の削減を考慮したレイアウト自動設計がこれから注目される新しい課題になるであろう。このような背景で本研究では、レイアウト自動設計における消費電力の削減手法について研究した。具体的には、次の 3 つ手法がある。

1. モジュール ジェネレータ GmC を用いた論理回路の再合成による消費電力の削減手法。

2. 論理回路の物理的な構造を考慮し、論理的等価である接続を入れ換えることによる消費電力の削減手法。

3. 論理シミュレーションによって各ノードの充放電を調べ、頻繁に充放電されるノードにつながっているセルを近くに配置し、配線容量、そして、消費電力が最小となるようにレイアウトする手法。

これらの手法の研究にあたって、低消費電力化を支援するレイアウト自動設計ツール **BASE** を製作した。

7.1 消費電力削減の一般的な手法

参考書 [1] より、CMOS 集積回路の消費電力は

$$P = fV_{DD}^2 \sum_n p_n C_n \tag{7.1}$$

と表される。ただし、f はクロック周波数、V_{DD}^2 は電源電圧、p_n は節点 n が 1 クロックで充放電する平均確率、C_n は節点 n の静電容量である。したがって、消費電力を削減するためには、動作周波数 f を下げる、電源電圧 V_{DD} を下げる [2, 3]、各節点における充放電する平均確率 p_n を下げる、および各節点における有効負荷容量を下げるなどの方法が考えられる。この中で、レイアウト設計の段階において、C_n と p_n の削減できる可能性がある。

C_n: C_n はゲートの入力容量、拡散容量、配線の負荷容量などの総和である。これらの容量は主にデバイス テクノロジーに依存するが、レイアウトに関係するものもある。例えば、第 2 章で述べたソース - ドレーン共有型レイアウトは拡散容量の削減に効果のある手法である。短い接続は配線容量を削減する主要な手段である。

p_n: p_n については、節点 n が論理ゲートの外部節点である場合、p_n は論理的な構造に関係するものであるが、節点 n が論理ゲートの内部節点である場合、p_n は論理ゲートの物理的な構造に関係するものである。例えば、論理的に等価である論理ゲートの入力端子を入れ換えることによって論理ゲートの内部節点の p_n が変わる [4]。

7.2 論理回路の再合成による消費電力の削減

消費電力として負荷容量の充放電電流による成分だけを考える時、論理ゲートの段数を減らせば、消費電力も減らすことができる。つまり、論理回路の再合成あるいは等価変換による消費電力の削減が可能である。論理回路の再合成による消費電力の削減処理は、第6章で説明した論理回路の再合成によるクリティカル パスの最小化と同じ処理であるので、ここでは省略する。ただし、消費電力が小さくなった回路の遅延時間は増える場合があるため、論理回路の再合成による消費電力の削減処理には、次の2つの制約を加える。

1. 処理の対象はクリティカル パス以外の論理ゲートとする。

2. クリティカル パスの遅延時間が増えないものとする。

7.3 論理ゲート入力端子の再割り当てによる消費電力の削減

式(7.1)からわかるように, 節点 n の充放電平均確率 p_n を小さくすることができれば、消費電力が小さくなる。節点 n が論理ゲートの内部節点である場合、p_n は論理ゲートの物理的な構造に関係するものである。つまり、論理上等価である入力端子を入れ換えることによって p_n が変わる。例えば、図7.1の2入力 NAND ゲートの入力 (a, b) に表7.1(a)の入力ベクタを与える時の、図7.1の C_z と C_i の充放電の変化を表7.1(a)に示す。ただし、"H" と "L" はそれぞれ充電された、放電された状態とする。a と b を交換した時の、同じ回路の C_z と C_i の充放電の変化を表7.1(b)に示す。表7.1からわかるように、入力端子の再割り当て処理によって C_i の充放電の回数は4から2に変わった。つまり、p_n は 4/11 から 2/11 に低減された。

3入力以上の論理ゲートの場合、C_i は "H" でもない、"L" でもない状態がある。例えば、図7.2において、$C_{ab} = H, C_{bc} = L$ のとき、(a, b, c) が $(0, 1, 0)$ になったら、C_{ab} と C_{bc} は "H" と "L" の間の中間状態になる。簡単

Table 7.1: 2 入力 NAND ゲートの入力ベクタと各節点における容量の充放電変化

(a) 入力端子の再割り当て処理前

No.	ab	C_z	C_i
1	10	H	H
2	11	L	L
3	10	H	H
4	00	H	H
5	10	H	H
6	11	L	L
7	10	H	H
8	00	H	H
9	01	H	L
10	00	H	L
11	10	H	L
Charges and Discharges		4	5

(b) 入力端子の再割り当て処理後

No.	ba	C_z	C_i
1	01	H	L
2	11	L	L
3	01	H	L
4	00	H	L
5	01	H	L
6	11	L	L
7	01	H	L
8	00	H	L
9	10	H	H
10	00	H	H
11	01	H	L
Charges and Discharges		4	2

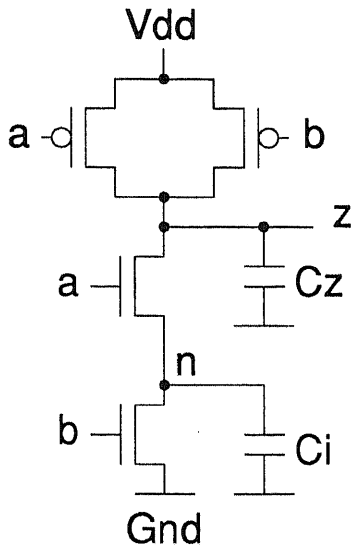


図 7.1: 2 入力 NAND ゲートの入力端子の再割り当てによる消費電力の削減

化のため、この中間状態は C_{ab} と C_{bc} のレベルの平均値とする。“H” を 1、“L” を 0 とする場合、中間状態は 0 と 1 の間となる。

節点 n の充放電平均確率 p_n は論理シミュレーションあるいは確率モデル [4] で求められる。後者を使う場合、それぞれ異なるタイプの論理ゲートについてモデル化する必要がある。本研究では、論理ゲートのタイプは無制限とするため、前者を採用した。また、回路の設計者は設計する回路の入力ベクタ、例えば、テスト パターン、命令コードなどをよく知っているあるいは持っているので、本研究では、論理シミュレーションによって、p_n を求める方法を採用した。入力ベクタが与えられていない時、入力ベクタはランダムに発生される。

論理回路の物理的な構造を考慮し、論理的等価である接続を入れ換えることによる消費電力の削減手法では、与えられた入力ベクタにしたがって論理シミュレーションを行ない、論理シミュレーションの結果から、論理ゲート入力端子のすべて可能な割り当てについて調べ、論理ゲートの内部節点の充放電回数が最小となる割り当てを求める。

論理ゲート入力端子の割り当てと論理シミュレーションとの 2 つの処理を繰

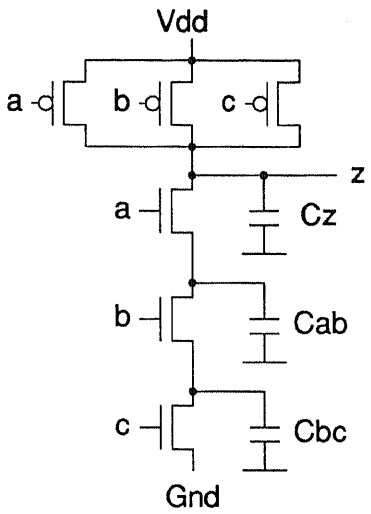


図 7.2: 3 入力 NAND ゲートの入力端子の再割り当てによる消費電力の削減

り返して行なえば、消費電力が最小となる割り当てを求めることができるが、このような方法では、シミュレーションの回数は可能な割り当て数になってしまう。本研究では、論理シミュレーションが 1 回だけ行なわれる手法を開発した。1 回の論理シミュレーションだけですべての可能な割り当てから、論理ゲートの内部節点の充放電回数が最小となる割り当てを求めることができる理由は、論理的に等価である論理ゲートの入力端子を入れ換えても、出力レベルに影響しないからである。本手法は次の処理からなる。

1. 与えられた入力ベクタに対して、論理シミュレーションを行なう。すべての信号線における電位レベルを保存する。

2. すべての論理ゲートの処理が終ったら、終了する。

3. 1 つ未処理の論理ゲートのすべて可能な割り当てについて、その内部ノードの充放電回数を表 7.1に示されたように計算する。充放電回数が最小となる割り当てを解とし、ステップ 2 へ。

この処理は論理ゲートの処理順序に依存しない。

図7.3は、Parthenon [5] で合成した Counter 回路に対して、実験した結果を示すものである。真中のカーブは割り当て処理を行なう前の、すべての論理ゲートの内部節点の充放電回数の合計を示すものである。一番下と一番上のカーブはすべての可能な割り当てについて調べた後の、論理ゲートの内部節点の充放電回数の合計の最小値と最大値を示すものである。この結果から計算すると、もし、$r = \frac{C_i}{C_z}$ が 0.1 から 0.9 の間となるとき、消費電力は 3% から 19% の削減が可能である。ただし、論理ゲートの内部節点に付いている容量をすべて C_i とし、外部節点に付いている容量をすべて C_z とする。

7.4 消費電力を考慮したレイアウト設計手法

消費電力の式 (7.1) からわかるように、消費電力の削減手法の1つは C_n を減らすことである。C_n の1つ重要な成分は配線容量である。配線の幅が一定の場合、配線容量は配線の長さに比例する。しかし、配線の長さの合計が最小となるようにレイアウトすることは消費電力が最小となるとは限らない。それは C_n に重み p_n が付いているからである。p_n は、前節に述べたように、論理シミュレーションで求められる。この節では、$p_n C_n$ の合計が最小、つまり、消費電力が最小となるようにレイアウトする手法について述べる。

消費電力を考慮したレイアウト手法では、第6章に述べた信号遅延を考慮したレイアウト手法によって求められたセルの配置結果を初期配置とする。ネット n の配線長は次の式で定義されるものとする。

$$L_n = W_n + \sum_{i \in n} |Y_n - y_i| \tag{7.2}$$

ただし、W_n はネット n の全端子を含む最小矩形の幅であり、Y_n はネット n の中心点の y 座標であり、y_i はネット n にある端子の y 座標である。式7.2は、ネット n の中心に横配線（トラック）が1本だけで、ネット n にあるすべての端子がそのトラックに縦配線でつながるという配線モデルと考えられる。

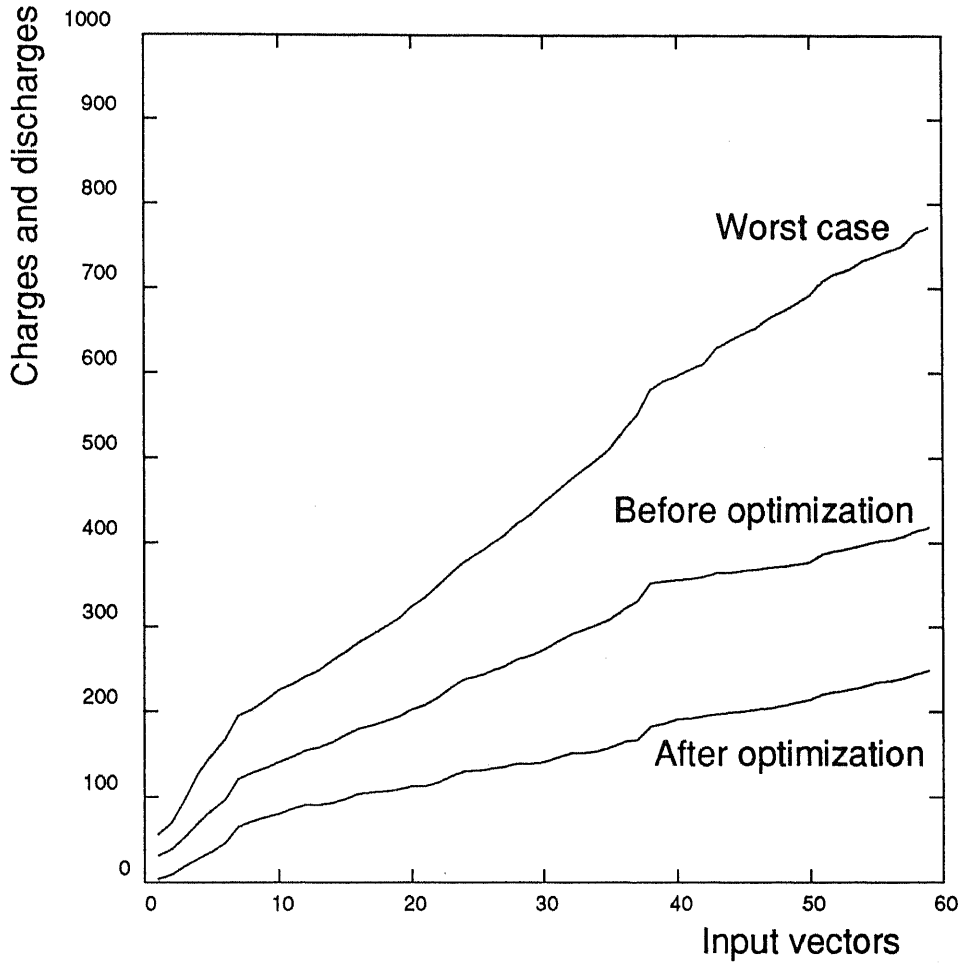


図 7.3: 入力端子の割り当て処理前後 Counter 回路の内部節点の充放電回数の合計

消費電力を考慮したレイアウト手法の基本的な考え方は、ネット n につながっているセルをそのネットの中心点に移動することにより配線長を短くするというものである。ここで言う移動とは、中心点へ向かって隣にあるセルとの位置交換のことである。処理の順序は $p_n\ C_n$ 積、つまり、$p_n\ L_n$ 積が最大となるネットからとする。それは $p_n\ L_n$ 積の大きいものを減らすことによる、消費電力の削減効果が大きいからである。セルの移動は、$p_n\ L_n$ 積の合計が最小となるまで繰り返される。この処理の詳細を次に示す。

1. 式 (7.2) で配線長 L_n を計算する。

2. $p_n\ L_n$ 積を計算し、$p_n\ L_n$ 積の昇順でネットの処理順序 Q_m を決める。最初に処理するネット $Q_m(m=1)$ とする。

3. ネット Q_m につながっているセルの処理順序を決める。ネット Q_m 中心点よりマンハッタン距離の遠い端子を持つセルが最初に処理されるものとする。この処理順序の狙いは、同じネットにある端子のそのネットの中心点までの距離を均等にすることである。

4. 決められたセル処理順序からセルを選び、選んだセルをネット Q_m の中心へ移動させる。セルの移動は y 方向優先とする。これは、セルが y 方向への集中によって、横配線数を減らすことができるからである。

5. セル移動によって $p_n\ L_n$ 積の合計が増えたら、その移動を行なう前の配置に戻す。

6. ネット Q_m につながっているどのセルを移動しても、$p_n\ L_n$ 積の合計が増えるとき、ネット Q_m の処理が完了したとする。ネット Q_m の処理が完了していなければ、ステップ 4 へ進む。ネット Q_m の処理が完了したとき、ネット Q_m の処理によって $p_n\ L_n$ 積の合計が減ったら、ステップ 1 へ進む。変らなければ、$m=m+1$ ステップ 3 へ進む。

以上の処理では、どのセルを移動しても $p_n\ L_n$ 積の合計が減少できないときだけが終了となる。また、$p_n\ L_n$ 積の合計が減ったら、配線長 L_n が再計算さ

れ、ネットの処理順序は再ソートされることにより、常に $p_n\ L_n$ 積が最大となるネットは優先して処理される。

7.5 消費電力を考慮したレイアウト手法の評価

この節では、前節に述べた消費電力を考慮したレイアウト手法による設計を示し、本手法の評価を行なう。

本手法では、第 6 章に述べた信号遅延を考慮したレイアウト手法によって求められた配置を初期配置とするため、クリティカル パスにあるセルは移動しないという制約が加えられている。また、回路の出力信号線が設計者によって決められている時、それらのセル、つまり、第 1 列にあるセルは移動しないという制約もある。

図 7.4の初期配置から、前節に述べた消費電力を考慮したレイアウト手法によって、求められた最終配置を図 7.5に示す。図 7.4の $p_n\ L_n$ 積の合計を 1 としたとき、図 7.5の $p_n\ L_n$ 積の合計は 77.9% となる。つまり、配線容量による消費電力は 22.1% 削減されたこととなる。この時の配線長の合計は 8.9% 削減された。比較のため、配線長の合計が最小となる配置も求めた。この時の配線長の合計は 14.2% 削減されたが、$p_n\ L_n$ 積の合計は 15.2% しか削減できない。よって、本手法は消費電力の削減に効果のある方法と考えられる。

本手法の 1 つ特徴は、常に $p_n\ L_n$ 積が最大となるネットから処理することである。それ以外の処理順序にすると、例えば、入力ファイルに記述したネットの順序で処理すると、$p_n\ L_n$ 積の合計は 13.5% しか削減できない。

最後に、回路の遅延を考えずに、つまり、クリティカル パスにあるセルの移動に制約をなくして、$p_n\ L_n$ 積の合計が最小となる配置を求めたが、この時、配線容量による消費電力は 46.8% 削減された。

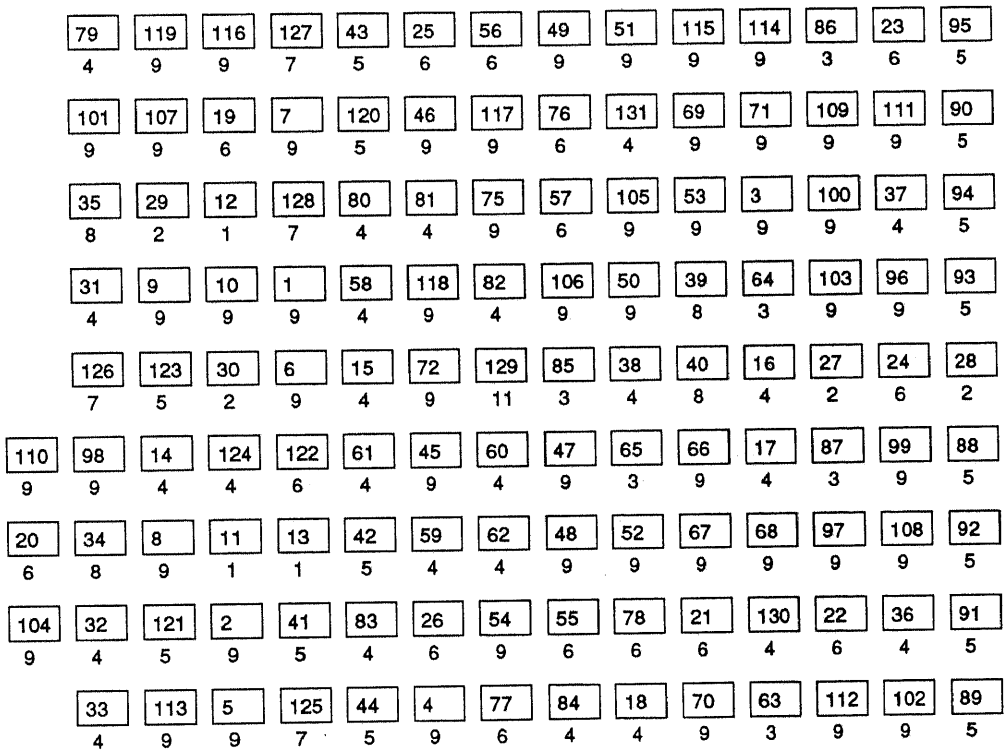


図 7.4: Counter 回路の初期配置（配線容量の削減前）

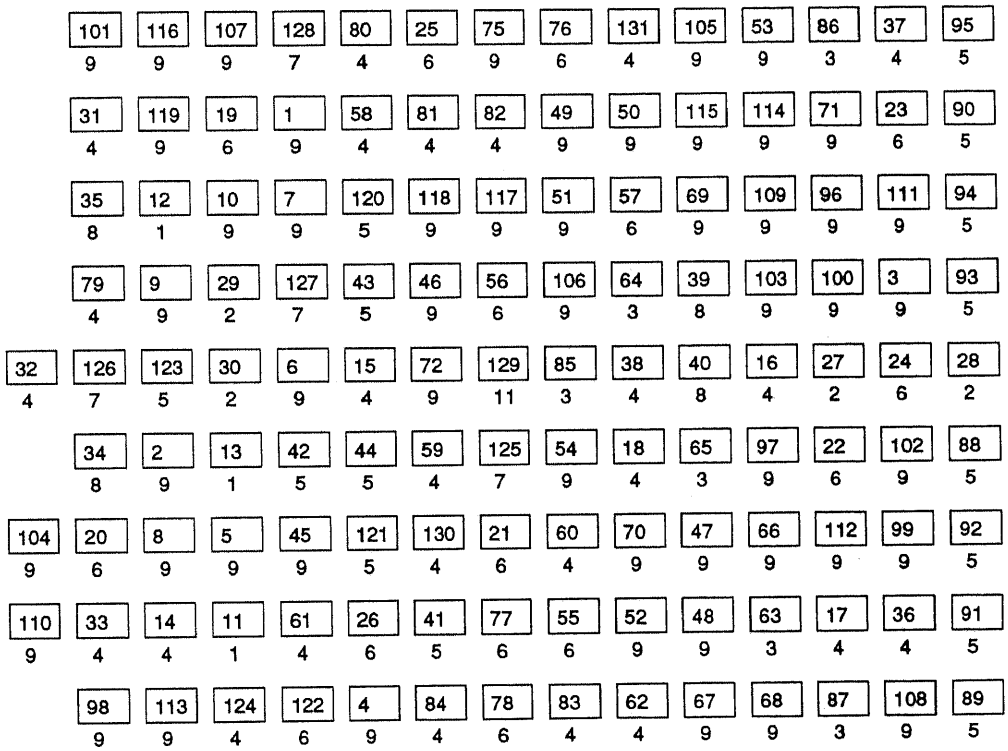


図 7.5: Counter 回路の最終配置（配線容量の削減後）

参考文献

[1] 飯塚 哲哉. *"CMOS 超 LSI の設計 "*. 培風館, 1989. (菅野 卓雄 監修).

[2] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen. "Low-Power CMOS Digital Design". *IEEE Journal of Solid-State Circuits*, 27(4):473–484, April 1992.

[3] M. Kakumu. "Process and Device Technologies of CMOS Devices for Low-Voltage Operation". *IEICE Transactions on Electronics*, E76-C(5):672–680, May 1993.

[4] J. Akita and K. Asada. "A method for reducing power consumption of CMOS logic based on signal transition probability". *The Proceedings of EDAC-ETC-Euro ASIC'94*, Feb.28 - Mar.3, 1994. (to be presented)

[5] NTT Network Information System Laboratory. *"Parthenon User's Manual"*. 1989.

第 8 章　結論

本論文は、CMOS 集積回路の高密度レイアウト手法における速度と消費電力の最適化について議論したものである。本論文の要点、特徴と新しい考え方は以下通りである。

1. セル ライブラリの制約をなくし、探索空間を広げれば、より高速、低消費電力の回路を発見することができる。これを実証するため、任意な論理回路に対しても、短い時間でコンパクトなレイアウトが自動設計できるモジュール ジェネレータを製作した。

2. 与えられたタイミング制約によって求められたレイアウトは、タイミング制約を守ることができるが、信号遅延が最小となるレイアウトを発見することができない。タイミング要求は与えられるものではなく、回路に固有なものであり、クリティカル パスの探索と遅延時間の解析によって求められるものである。

3. 消費電力の削減に関する研究はデバイス テクノロジー、アーキテクチャ設計、論理設計などの部門に繰り広げられているが、レイアウト設計における消費電力の削減は、今のところでは、本研究だけである。

本研究を通じて以下の結論が得られた。

第 2 章の結論：

- レイアウト面積、信号遅延、消費電力などを削減することができるソース - ドレーン共有型レイアウトを 2 次元レイアウトに拡張すると、1 次元レイアウトより、最大 20% 平均 10% 程度面積が減少することが可能である。しかし、2 次元レイアウトには、トランジスタのゲートが曲がることがあるなどの問題が存在するため、本研究では、1 次元レイアウトのスタイルを採用した。

第 3 章の結論：

- 処理順序に依存しない一括処理型トランジスタ ペアリング手法は、トランジスタの接続関係によってトランジスタ ペアを作成するため、非デュアルな CMOS トランジスタ回路を含めて任意な CMOS トランジスタ回路に適用できる。
- 一括処理型トランジスタ ペアリング手法によって見つかったトランジスタのペアは必ずしもデュアルな関係を持つわけではないが、デュアルな関係で作ったペアより最適な解であることが実験から示された。

第 4 章の結論：

- トランジスタの接続関係をコンパチブル マトリクスで表現することにより、非直並列回路を含めて、任意の CMOS 回路を表すことができる。
- コンパチブル マトリクスには、探索する必要のないトランジスタ ペアに関する情報も含まれているので、無駄な探索を避けることができるので、従来のアルゴリズムより数倍から数百倍速いという結果が実験から示された。

第 5 章の結論：

- 一括処理型トランジスタ ペアリング アルゴリズムとコンパチブル ペア探索アルゴリズムを取り入れたため、CMOS トランジスタ回路モジュール ジェネレータ **GmC** は非直並列、非デュアルな回路を含めて、任意の CMOS トランジスタ回路に対して適用できる。

- 与えられた CMOS トランジスタ回路にデュアルなオイラー パスが存在すれば、**GmC** は、必ずそれを見つけることができる。

- 典型的な CMOS トランジスタ回路について、**GmC** によって見つかったパスの数は論理上の最小値に達した。

- 与えられた CMOS トランジスタ回路をカバーする最小のパス数を求める問題において、最適なトランジスタ ペアが見つからなければ、網羅的なパス探索アルゴリズムがあっても、その回路をカバーする最小のパス数を見つけることができない。

第 6 章の結論：

- レイアウト自動設計における信号遅延の削減手法について研究した結果では、モジュール ジェネレータ **GmC** を用いた論理回路の再合成手法と論理回路の物理的な構造を考慮し、論理的等価である接続を入れ換える手法による信号遅延の削減が効果のあることがわかった。

- クリティカル パスの探索と遅延時間の解析によって求められたタイミング要求にしてがって、セルを配置する手法は信号遅延が最小となるレイアウトを発見することができる。

第 7 章の結論：

- レイアウト自動設計における消費電力の削減手法について研究した結果では、クリティカル パス以外の論理ゲートに対して、モジュール ジェネ

レータ GmC を用いた論理回路の再合成手法と論理回路の物理的な構造を考慮し、論理的等価である接続を入れ換える手法による消費電力の削減は効果があることがわかった。この 2 つ手法を用いて、PARTHENON で設計された回路より、平均 10% の消費電力削減が可能である。

- 論理シミュレーションによって各ノードの充放電を調べ、頻繁に充放電される節点につながっているセルを近くに配置し、配線容量、そして、消費電力が最小となるようにレイアウトする方法について研究した結果では、回路のスピードを維持する前提で、およそ 20% の消費電力削減が可能である。回路のスピードを緩めるならば、消費電力の削減率はさらに 50% まで達することが可能である。

本研究に関する公表文献

1. 張 洪明、 浅田 邦博, “２次元 MOSFET セルを用いたレイアウト設計”, 電子情報通信学会論文誌 (A), **J75-A**, 5, pp. 960-962, １９９２年５月。

2. H. Zhang and K. Asada, “A General and Efficient Mask Pattern Generator For Non-series-parallel CMOS Transistor Network”, in “Synthesis for control dominated circuits”, Edited by G. Saucier, Elsevier Science Publishers B. V., 1993

3. H. Zhang and K. Asada, “A General Layout Algorithm for Non-Series-Parallel CMOS Transistor Networks Based on Compatible Matrix Representation”, Submitted to IEEE Transaction on CAD.

本研究に関する発表

1. 張 洪明、 浅田 邦博, “配線容易化のための配置設計手法”, 電子情報通信学会秋季全国大会, **A-49**, １９９１年９月。

2. H. Zhang and K. Asada, “A General and Efficient Mask Pattern Generator For Non-series-parallel CMOS Transistor Network”, WG 10.5 IFIP Workshop on Synthesis, Generation and Portability of Library Blocks for ASIC Design (France), pp. 132-138, Mar., 1992.

3. 張 洪明、 浅田 邦博, “非直並列 CMOS 回路のレイアウト最適化手法”, 電子情報通信学会秋季全国大会, **A-61**, １９９２年９月。

4. H. Zhang and K. Asada, “An Improved Algorithm of Transistors Pairing for Compact Layout of Non-Series-Parallel CMOS Networks”, IEEE Proc. Custom Integrated Circuits Conference (U.S.A), pp. 17.2.1-17.2.4, May 1993.

5. 張 洪明、 浅田 邦博, “複合ゲート CMOS 回路のレイアウトのためのトランジスタペアリング手法” 電子情報通信学会春季全国大会, **A-95**, １９９３年３月。

6. 張 洪明、 浅田 邦博, “コンパチブルペア探索アルゴリズムによる CMOS 回路のレイアウト最適化” 第 6 回 回路とシステム軽井沢ワークショップ, pp. 61-65, １９９３年４月。

7. 張 洪明、 浅田 邦博, “PARTHENON の設計結果からレイアウト設計の一案”, 電子情報通信学会秋季全国大会, **A-68**, １９９３年９月。

謝辞

本研究を行なうに当たり、常に私に訓戒をしていただき、懇切な御指導と適切な助言を賜りました指導教官の浅田邦博先生に深く感謝致します。

大学院在学中、いろいろな面で御指導、御協力をいただいた、大学院生の戴志堅氏（現在松下電送株式会社）、池田誠氏、張子誠氏、池野理門氏、秋田純一氏、三堂哲寿氏、技官の鈴木真一氏、鐘紡株式会社の佐藤純一氏、日産自動車株式会社の岩崎靖和氏と青柳稔氏、日本モトローラ株式会社のマイク リー氏、と他の方々に感謝致します。

本研究の一部は、財団法人国際コミュニケーション基金 (ICF) の第９期留学生助成援助金のもとに行われたもので、この場を借りて、心からお礼申し上げます。また、援助期間中にお世話になった国際コミュニケーション基金の太田中一専務理事、稲尾登志高事務局長、小島伸介主管、小林孝江主事と国際電信電話株式会社 (KDD) 関係者の方々に感謝致します。

財団法人電気、電子情報学術振興財団からは国際研究集会出席助成をいただきました、心からお礼申し上げます。