設計自動化 55－6
(1990.12.12)

多段組合せ論理回路合成システム SOLDIER

平峰 正信　石川 淳士　野村 和男　佐藤 晴美
東田 基樹　数馬 好和
三菱電機（株）

あらまし　SOLDIER はテクノロジに依存する最適化機能を強化した多段組合せ論理回路合成システムである。本システムの特徴は、合成回路の面積を縮小するために、多段論理簡単化アルゴリズムを強化したこと、および遅延性能を改善するために、テクノロジマッピングの段階でクリティカルパスを圧縮することである。各セルのセル特性（占有面積、遅延時間等）を含むセルライブラリを十分に活用することができるため、遅延改善に伴う面積の増加を最小限に抑え、目標性能に適した論理構造を得ることが可能である。論理合成システムのベンチマークデータを使って評価したところ、クリティカルパスの圧縮とともに遅延が改善され、高品質な回路を生成することが確認できた。

SOLDIER: A Multilevel Logic Synthesis System

Masanobu HIRAMINE　Junji ISHIKAWA　Kazuo NOMURA　Harumi SATO
Motoki HIGASHIDA　Yoshikazu KAZUMA
MITSUBISHI ELECTRIC CORP.
5-1-1 OFUNA, KAMAKURA, KANAGAWA 247, JAPAN

Abstract　This paper describes SOLDIER, a multilevel logic synthesis system with strong technology-dependent optimization capabilities. One of the special features is that the algorithm for optimization of multilevel functions is enhanced. Another one is that logic structures along critical paths are improved at the stage of technology mapping in order to reduce the path delay of the circuit. It takes full advantage of area and timing information of a given cell library and creates good logic structure of the circuit under the given constraints. Applied to the open bench mark data for synthesis systems, the system successfully generated high quality circuits in terms of both area and delay.

1 はじめに

組合せ論理の自動合成に関する研究、開発の歴史はPLAの簡単化に利用できる二段組合せ論理最小化手法[1, 2]から始まった。しかしながら、一般に論理式を二段論理で表現するよりも多段論理で表現する方が簡潔になるため、その後、多段論理簡単化手法[3]の研究が活発になった。この多段論理を現在主流となっているゲートアレイや標準セルLSIを使って実現するためには、抽象論理ゲートで構成された論理回路から多種多様なマクロセルで構成された論理回路を生成する論理変換手段（テクノロジマッピング）[5, 6]がさらに必要となる。現実に、多段組合せ論理回路合成システムは、上記のような二段論理最小化手段、多段論理簡単化手段そして論理変換手段とから構成される。

ところで、これまでに開発された代表的な多段組合せ論理回路合成システムにMIS[3]とSOCRATES[4]がある。MISは強力な大域的多段論理簡単化アルゴリズムを用いた論理合成システムではあるが、ゲートアレイや標準セルLSIで強いられるマクロセルライブラリに依存した最適化を指向するものではなかった。そのため、論理回路をターゲットテクノロジの下で実現するためには、他の論理変換手段と組合せる必要がある。一方、SOCRATESはゲートアレイや標準セルLSIをターゲットとし、多段化した組合せ論理から、ルールベース手法に基づく局所的論理変換の手法により論理回路を合成する。この方式では、合成回路の性能を左右する論理構造は多段化の際に決定される。しかし、多段化フェーズでは、クリティカルパスを正確に識別することは難しく、必ずしも目標性能に適した論理構造が得られるとは限らない。

SOLDIERはゲートアレイや標準セルLSIをターゲットとし、マクロセルライブラリに依存する最適化機能を高めた多段組合せ論理回路合成システムである。本システムでは、多段論理簡単化手法については、MISと同様の方法を用いているが、さらに合成回路の面積を縮小するために多段化アルゴリズムを強化している。また、論理変換手法においては、合成回路の性能を向上させるために論理構造を改善する機能を追加している。この方式に従えば、論理変換後の回路に対する正確な性能評価に基づいてクリティカルパスを識別した後、ライブラリの特徴を十分に活用して、このクリティカルパスを短縮することができる。従って、目標性能を達成するための論理構造の変更に伴う面積増加を最小限に抑えることができ、高品質な回路を生成することが可能となる。

本稿では、特にSOLDIERの遅延改善方法に焦点をあてて説明する。まず、2章ではSOLDIERのシステム構成と処理概要を説明する。3章では遅延改善の中心的役割を担う論理変換法の処理概要を述べ、続く4章でこの遅延改善法について詳述する。5章ではベンチマーク回路の合成結果を示し、考察・検討を行なう。最後に6章で結論を述べる。

2 SOLDIERの概要

2.1 システム構成

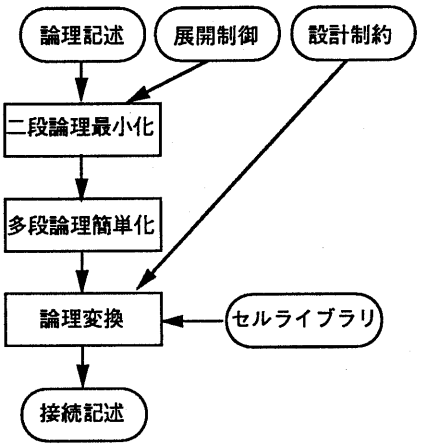

図1: SOLDIERの構成

図1にSOLDIERのシステム構成を示す。合成システムへの入力となる論理記述には次の形式が可能である。

1) ブール式

2) 真理値表

3) ネットリスト

また、設計制約とは最大許容面積や最大許容遅延を指定するものである。遅延に関しては各プライマリ入力ピン毎に 立ち上がり信号到達時刻と立ち下がり信号到達時刻、各プライマリ出力ピン毎に立ち上がり信号要求時刻と立ち下がり信号要求時刻を指定することも可能である。また、プライマリ入力ピンの駆動能力やプライマリ出力ピンの負荷容量を自由に指定することもできる。展開制御とは二段論理最小化の前処理として変数の展開方法を指定するものである。例えば、論理記述内の中間変数を全て展開して二段論理形式にするとか、あるいはこれとは反対に、中間変数や排他的論理和は展開せずに保存するといった展開方法がある。合成結果を論理記述の仕方に依存しないようにするためには前者のように、

また、論理記述の論理構造を反映した最適化処理をしたければ後者のように指定すればよい。

2.2 処理概要

SOLDIERにおける最初の最適化処理では、入力された論理記述を展開制御に応じて展開した後、論理の冗長性を二段論理最小化プログラムMINI2[1]により排除する。その後、Weak division[3]により共通論理を括りだして、多段の論理式を生成する。この際、多段化された論理記述の2変数間に否定関係が成立する場合、一方の変数を他方の変数を用いて置換するように多段化アルゴリズムを新たに強化した。これにより一層の論理式の単純化に成功している。この処理は、Boolean division[3]を用いた再置換処理の特殊な場合と考えることができるが、その処理の簡単さにもかかわらず、面積を大幅に縮小する効果を発揮する。

次のような論理式を考える。

$$X = a*b*\bar{c} + \bar{a}*\bar{b}*\bar{c} + a*\bar{b}*c + \bar{a}*b*c \quad (1)$$

上式を代数除算による因数の括りだしを行うことにより、次式が得られる。

$$X = (a*b + \bar{a}*\bar{b})*\bar{c} + (a*\bar{b} + \bar{a}*b)*c \quad (2)$$

さらに、括弧内の論理式を中間変数A、Bで置換することにより、次式のような多段論理式が得られる。

$$\begin{aligned} X &= A*\bar{c} + B*c \\ A &= a*b + \bar{a}*\bar{b} \quad (3) \\ B &= a*\bar{b} + \bar{a}*b \end{aligned}$$

上式を代数式と考える限り、これ以上、簡単化することはできない。しかし、ブール式として捕らえると、AとBが互いに否定関係を有することから、Bを$\bar{A}$で置換することが可能で、上式はさらに次式のように変形できる。

$$\begin{aligned} X &= A*\bar{c} + \bar{A}*c \quad (4) \\ A &= a*b + \bar{a}*\bar{b} \end{aligned}$$

最後に、多段論理式と同等の機能をセルライブラリに登録されているマクロセルを利用して構成する、論理変換処理を行なう。図2は(4)式に対して論理変換を施した結果を示す図である。この図からわかるようにSOLDIERはXとAの論理から排他的論理和のパターンを認識して簡単な接続記述を合成する。

ところで、多段論理式の構造は論理変換の自由度を束縛し、最終回路の性能に大きな影響を与えている。そ

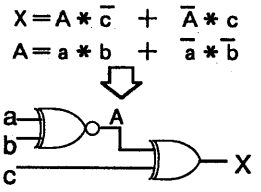

図 2: 論理変換の例

のため、目標性能を実現するためには、それに適した論理構造を得る必要がある。SOLDIERがターゲットとしているテクノロジは複合マクロセルや駆動能力の異なった多種多様な素子をもつため、論理変換前の回路から最終回路の性能を正確に見積もることは難しい。そこで、SOLDIERでは、最初の抽象論理を扱う多段化フェーズでは、遅延について考慮することなく、総リテラル数の最小化を目的としている。そして論理式の総リテラル数と回路面積との間には論理変換後においても高い相関があることから、多段論理化において生成された論理構造は面積においては最適であると仮定している。一方、遅延については、この論理構造を初期構造と考え、論理変換時に後述する素子の展開と圧縮という処理によって論理構造を変更しながら改善を繰り返す方式を採っている。この展開・圧縮処理は遅延解析の結果から得られたクリティカルな部分に対して、マクロセルのレパートリーを十分に活用して行なわれる。この様な方法に従えば遅延制約を満足し、しかも小さな回路が合成できるようになる。

3 論理変換

3.1 構成

論理合成の最終ステップは多段論理式からゲートレベルの回路を生成する論理変換である。前述したように、SOLDIERでは、多段化処理された論理式は面積については最適な論理構造をしていると仮定している。そこで、論理変換処理は、最初に面積最小回路を生成した後、遅延を逐次改善する手順をとることにしている。

ここでは、設計制約を満足しながら面積の小さな回路を得ることが論理変換の目的であるが、この目標の達成のためには、遅延を改善することによる面積の増加を最小限に抑えなければならない。論理変換フェーズで面積と遅延の間にトレードオフが生じる最大の要因の一つがマクロセルの割り付け方の相違であるので、高性能な回路を効率良く得るためには、対象テクノロジの特徴を最大限に利用して割り付け方を決定することが重要となる。特に複合マクロセル（OR-NAND、OR-AND-NOR）

やCMOSの場合には正論理マクロセル（AND、OR等）などを利用すると回路の性能が飛躍的に向上する可能性がある。

そこで、SOLDIERでは、対象テクノロジの特徴を活用するため、また種々のテクノロジの変更に対し柔軟に対応するために、論理変換処理部はルールベースシステムとして構成している。本システムについては、既に[6, 7]で発表しているため、本稿では、新たに加わった遅延改善処理のメカニズムを中心に説明する。論理変換は次のような手順に従って動作する。まず、多段論理式に対し初期割り付けを行なう。割り付けが終了すると、これに対し、面積と遅延の評価を行なう。ここで遅延制約違反が生じる場合にはクリティカルパスに対して、遅延改善ルールが適用される。この評価と遅延改善の一連の処理は以下の終了条件のいずれかが成立するまで繰り返される。

(1) 面積が許容値を越える。

(2) 遅延が許容域に入るか、改善が見られない。

3.2 回路分割

一般に、ルールベースシステムの処理速度は回路規模に対し、指数関数的な増大傾向を示す。そこで、図3(a)が示すように全体回路を論理木とバッファ木に分解して、論理変換問題を、個々の木に対する小さな論理変換問題に置き代えることを考える。このことにより、回路の分割と分割された回路の復元に要するオーバーヘッドを除くと、回路規模にほぼ比例する処理速度が得られる。ここで問題となるのが回路分割が与える変換品質への影響である。もし2つ以上の木に跨るような変換処理がないとすれば、分割による影響はまったくなく、個々の木に対する最適解は、元の問題の最適解である。しかしながら、一般には、信号の極性や出力負荷の調整などは隣接する木の間で発生する問題であり、また、元来、局所変換は最適性を保証するものではない。そこで回路分割と局所変換による回路品質の低下を極力小さくするために、変換処理の適用順序を以下に述べるように制御している。

まず、回路を論理木とバッファ木に分割する。マクロセルの出力極性の決定や、負荷分散は、論理木とバッファ木の両領域に関連する問題であることから、論理木とこの出力に接続するバッファ木の対を一つの部分回路として扱うことにする。この部分回路を節点とし、信号の依存関係を有効枝として表してグラフを構成する（図3(b)）。このグラフを用いて、変換可能な節点を、出力側に接続された全ての節点が変換済みであるような節点に限定する。この規則によると最も出力側に位置する部分回路が最初に変換され、順次入力側の部分回路が処理されることになる。また、個々の部分回路内においても、出力側から変換を行なう。このように、変換順序を定めることにより以下の効果を得ることができる。

- 素子の出力負荷が既知となるため、負荷に応じたマクロセルの選択が可能である。
- 割り付け時に生じたインバータを入力側で吸収することができるため、面積縮小や遅延短縮を実現する極性の選択が可能となる。

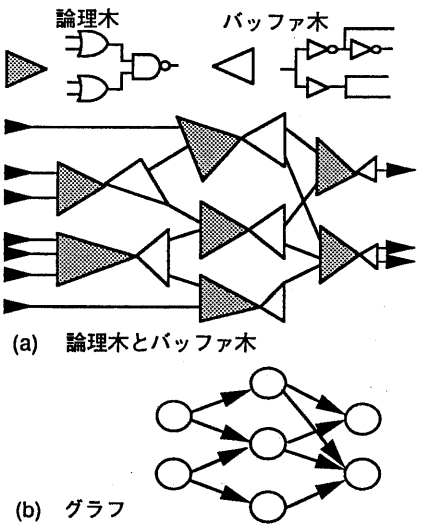

図3: 回路分割

3.3 遅延の評価

遅延を改善するために、以下の式を用いて、クリティカルパスを算出する。ここで信号到達時刻はプライマリ入力の信号到達時刻とドライブ能力をもとに全ての信号に対して計算でき、信号要求時刻はプライマリ出力の信号要求時刻と負荷から全ての信号に対して計算できる。余裕時間は信号要求時刻と信号到達時刻の差として定義され、この値が負であるパスがクリティカルパスとなる。なお遅延解析は立ち上がり信号と立ち下がり信号は別個に計算される。式中のDはマクロセルの入力ピンから出力ピンまでの固有遅延と負荷容量に依存する配線遅延の和を表しており、ライブラリに記載されているパラメータを使って計算される。

- 信号到達時刻

$$A(y) = max\{A(x) + D(x, y, load)\} \quad x \in fanin(y)$$

- 信号要求時刻

$$R(y) = min\{R(x) - D(y, x, load)\} \quad x \in fanout(y)$$

- 余裕時間

$$S(y) = R(y) - A(y)$$

4 遅延の改善

マクロセルが割り付けられた回路では、経験的に言って素子段数と遅延の相関は非常に高い。そこでSOLDIERでは、以下に示すような論理変換法に従って、最大素子段数の増加を抑止するとともに、可能ならばクリティカルパス上に並ぶ素子を圧縮して遅延を改善する。

4.1 素子段数の圧縮

(1) 多入力セルの分割

マクロセルを割り付けようとしている論理（ANDゲート、ORゲート等）の入力ピンの数が、そのマクロセルのピン数を越える場合、この大きな論理を小さな論理に分割しなければならない。図4は論理の分割法を説明する図である。図中のCはクリティカルな信号を表している。クリティカル信号を含まない場合には、論理は単に木状に分割されるが、クリティカル信号を有する場合には、SOLDIERでは、図4(a)(b)のようなクリティカルパスの素子段数を増加させない分割方法が選択される。特に図4 (b)の場合はマクロセルが分割形式を決定している。

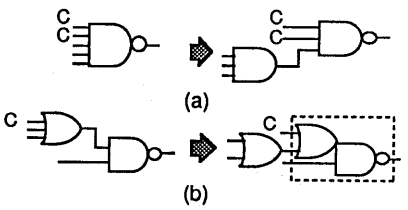

図 4: 多入力セルの分割

(2) 複合セルの活用

複合セルを活用する目的の一つは、面積効率を上げることであるが、遅延の短縮を目的として利用できることも見逃すことはできない。そこで、SOLDIERは複合セルを有効に利用してクリティカルパスを圧縮している。図5は面積を優先した割り付け法と遅延を優先した割り付け法を同一の回路パターンに対して適用した場合の様子を説明している。面積を優先する場合には図5(a)のような割り付け方が選択されるのに対し、遅延を優先する場合には、図5(b)のようなクリティカルパス上の素子段数を圧縮する方の割り付け方が選択される。クリティカルパスがない場合にはいずれの場合においても面積効率を考えた図5(a)のほうが選ばれることは言うまでもない。クリティカルパスの有無や面積効率を、総合的に評価するために導入した評価関数が、これら競合する複数の割り付け法の中から一つの方法を選択してくれる。

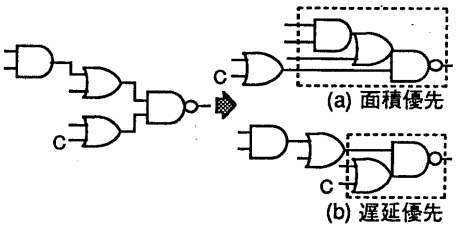

図 5: 複合セルの活用

(3) 極性の最適化

素子のピン極性の選択の仕方によっては、その素子の周辺に付随したインバータを少なくして面積を小さくする場合がある (図6(a))。このピン極性の最適化においてもクリティカルな信号を有する回路については、特別の配慮をはかることでクリティカルパスの素子段数を短縮することができる。図6(b)は出力極性を反転させて、出力側から伝搬されてきたクリティカル上のインバータを吸収している。ただしこの変換が許されるのは、新たに発生したインバータを通過する素子段数がクリティカルであった素子段数より長くならない場合に限られている。

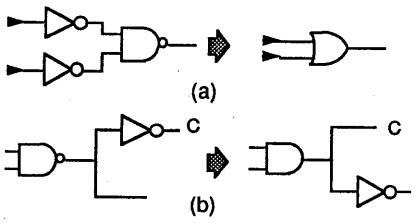

図 6: 極性の最適化

(4) 駆動能力の最適化

ライブラリによっては、論理が同一で、遅延パラメータだけが異なるマクロセルをレパートリーに加えているものがある。この様な時には、単にマクロセルを置換することで、接続関係をまったく変えずに、遅延を短縮できる場合がある。

4.2 素子の展開

クリティカルパス上の隣接する2つの素子が分岐した信号線で結線されている場合には、従来の論理変換ではこれを一つの素子にまとめるような割り付けを行なうことはできない。そこで、同一論理の素子を生成し、これをクリティカルパスに接続する。この処理は括り出されていた共通論理をクリティカルパスについてのみ展開することに相当するもので、これを素子の展開と呼ぶことにする。次にクリティカルパス上のマクロセルを変換前の抽象素子の状態に戻した上で、上述した素子の割り付け法に基づく再変換処理を行なう。その結果、素子段数が圧縮され、遅延は短縮される。従来の論理変換では行なわれなかった信号線の分岐を越えた割り付けを行なうことで、論理構造を改善することができる。しかも、論理変換フェーズでのクリティカルパスは正確な遅延評価に基づくため、論理構造の変更は遅延改善に直接影響を与えることになる。また、クリティカルパス以外の部分は、ほとんど変化がないため、他の部分への悪影響を最小限に抑えられる。

図7は素子の展開と圧縮を利用して遅延を短縮した例を表している。図7(b)は図7(a)のORに相当するマクロセルをクリティカルパスに対して展開した時の様子を示している。ここで再割り付けを行なうと図7(c)のように複合マクロセルが利用されて素子段数が圧縮される。

素子の展開処理はルールベースに蓄積されている展開ルールにより起動される。このルールは、再割り付け後の素子段数が増加することが明らかな回路パターンに対してはマッチングしないようになっている。

4.3 バッファリング

これまでに述べてきた遅延改善法は主として素子段数の圧縮によるものであった。しかし、大きな負荷を持つ信号線ではその信号を駆動する素子の固有遅延よりも配線遅延のほうが大きくなる現象がしばしば起こるように、過大な負荷に起因する遅延の増加を無視することはできない。そこで、考えられるもう一つの有効な遅延改善方法として、バッファ木の負荷を分散する方法がある。クリティカルパスを含まない過負荷なバッファ木に対しては、通常、図8(a)のように、木状に構成したインバータなどを用いて集中していた負荷を均等に分散するが、バッファ木の中にクリティカルパスがある場合、趣が少し異なり、クリティカルパスを孤立化するように負荷を分散する。図8(b)はインバータを利用してクリティカルパスを孤立化した様子を示している。この変換はクリティカルな信号線の素子段数を変えず、しかも負荷を減ずることで配線遅延を小さくすることを狙っている。ただし、新たにインバータが挿入されたパスの遅延が以前のクリティカルパスの遅延よりも大きくなる場合は、この変換は行なわれない。この時は、遅延改善の効果が若干劣る図8(c)のようなドライバーを利用したクリティカルパスの孤立化が試される。

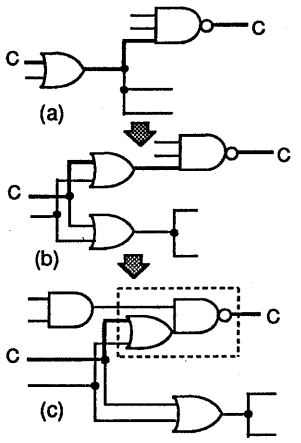

図 7: 素子の展開と圧縮

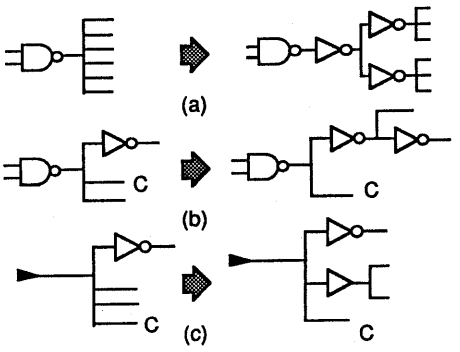

図 8: バッファリング

5 評価・考察

5.1 評価方法

1986DACで使用された、自動合成プログラムの評価用ベンチマークデータと二段論理簡単化プログラムの評価用ベンチマークデータおよび当社の機能マクロを使用して以下の通り、本合成システムの評価を行なった。

(1) 面積最小化、遅延最小化の合成を行ない、それぞれについて、合成回路の回路面積、 遅延時間および最大素子段数を測定する。

(2) rdm6 を例にとり、面積と遅延のトレードオフを測定する。

対象テクノロジは当社の CMOS ゲートアレイである。また、入出力ピンに接続する配線遅延は全体の遅延に対して無視できるものではない。そこで、この配線遅延を考慮するために入力ピンと出力ピンにはインバータ相当の素子が接続した状態を想定して入力ピンの駆動能力と出力ピンの容量を設定して合成している。

5.2 結果と考察

面積最小と遅延最小を指定して合成を行なった時の回路面積、遅延時間、素子段数を表 1に示す。遅延最小化モードでは面積最小化モードに比べ、遅延は最大で 36 %、平均で 20 %改善された。この時、面積の増加は平均 19 %である。また、素子段数のほうは平均で 22 %短縮されている。次に、rdm6 を例として、面積と遅延のトレードオフを調べた結果を図 9に示す。これらの合成回路から SOLDIER の合成方式の特徴を見つけることができる。つまり、回路 A、B、C の順に遅延が短くなるにつれて最大の素子段数も 7、5、4 段と少なくなっていることが分かる。 図 10は、素子段数の圧縮と遅延の改善との関係を鮮明に表わすために作成したグラフである。この図から分かるように圧縮される素子段数が大きいほど改善される遅延が大きく、両者の相関が高いことを示している。また、圧縮段数が 0 の場合でも、遅延が改善しているように、バッファリング処理も遅延改善に貢献しており、段数圧縮処理と協調して効果的な遅延改善を達成していることがうかがえる。

このように、我々が対象としているセルライブラリに対しては良好な結果を得ることができた。そこで今度は、MCNC のベンチマークで使用されるライブラリをターゲットとして合成を行なってみた。すると、上記の評価結果程の遅延改善効果を見ることができなかった。その原因は、MCNC のライブラリでは、マクロセルのレパートリーが非常に限定されたものであるため、上述したような遅延を改善する展開ルールが十分に適用することができなかったからである。一般に使用されるゲートアレイや標準セル LSI ライブラリのセルレパートリーは、今回、ターゲットとしたものに近く、MCNC に比べ遥かに豊富であるので、本システムは当社ゲートアレイのみならず、一般のゲートアレイや標準セル LSI に対して広く適用可能と考えられる。

回路名	面積最小			遅延最小		
	面積 (bc)*	遅延 (nS)	素子段数	面積 (bc)	遅延 (nS)	素子段数
5xp1	176	4.81	8	216	3.53	6
sao2	273	4.88	8	330	3.97	6
vg2	168	6.70	11	220	4.27	7
bw	390	5.38	9	469	4.09	5
adr4	840	7.48	11	1024	5.69	7
duke2	78	2.71	4	78	2.71	4
mlp3	136	3.29	5	161	2.90	5
rdm6	92	3.66	7	111	2.65	4
rdm8	172	3.86	6	209	3.47	5
log6	215	4.89	7	270	3.53	5
nrm3	176	4.10	6	220	3.70	5
rot6	117	3.35	6	146	2.72	4
sqr6	218	4.34	7	262	3.45	5
cntr	184	5.40	9	213	4.06	7
t085	128	4.23	6	139	3.74	6
t147	59	1.96	4	59	1.46	4

* pチャネル形TrとnチャネルTr形の対

表 1: 評価結果

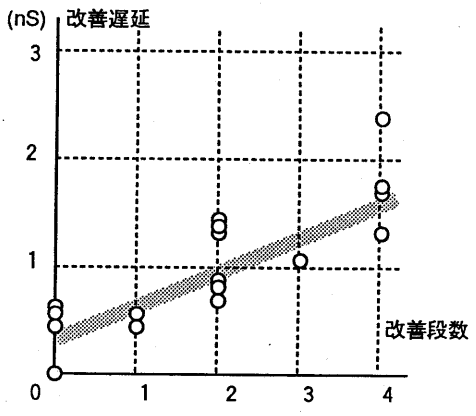

図 10: 改善段数と改善遅延の関係

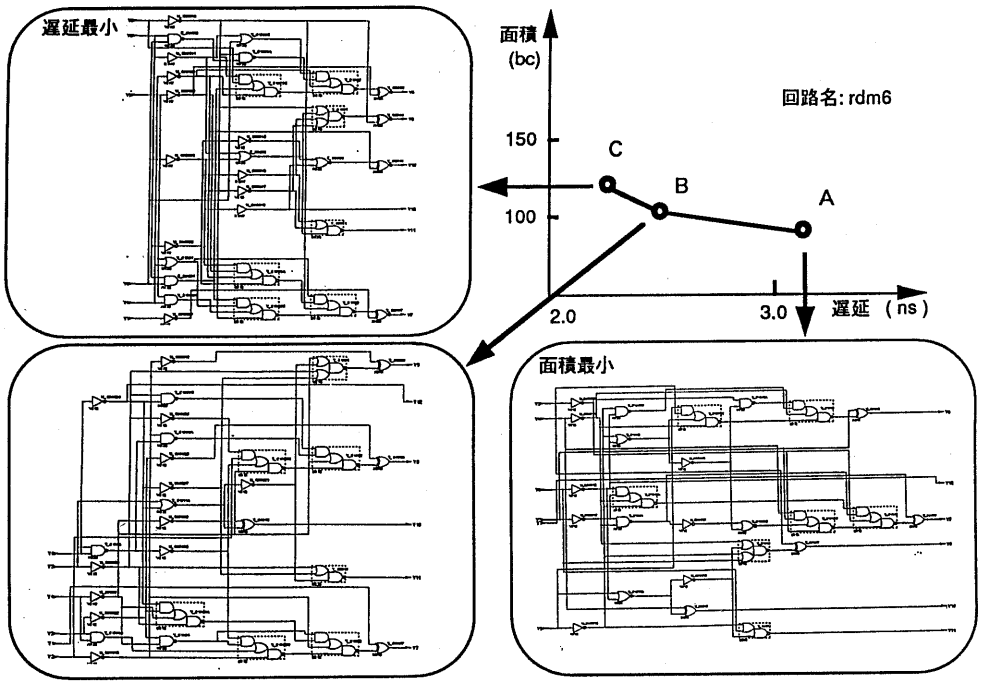

図 9: 面積と遅延のトレードオフ

6 おわりに

ゲートアレイやスタンダードセルをターゲットとし、マクロセルライブラリに依存する最適化機能を強化した組合せ論理回路合成システム SOLDIER を開発した。本システムの特徴は遅延を改善するために論理変換のフェーズで論理構造の変更を伴うような素子段数の圧縮を行なうことである。これにより、評価結果が示すように面積と遅延のトレードオフから所望する高品質な回路を自在に合成することが可能となった。

MCNC のベンチマークで使用されるような、非常に限られたレパートリーのライブラリに対しては、本稿で述べた素子段数の圧縮が期待どおりの効果を発揮しないことがあるため、今後は、素子の展開ルールを充実する必要がある。

参考文献

[1] T.Sasao,"Input variable assignment and output phase optimization of PLA's," *IEEE Trans.Comput.* Vol.C-33, No.10, Oct.1984, pp.879-894.

[2] R.K.Brayton, G.D.Hachtel, C.T.McMullen, A.Sangiovanni-Vincentelli, "Logic Minimization Algorithms for VLSI Synthesis," *Kluwer Academic Publishers*(1984).

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

[4] K.Bartlett, W.Cohen, A.D.Geus, G.Hachtel,"Synthesis and Optimization of Multilevel Logic under Timing Constraints," *IEEE trans. on CAD*, Vol. CAD-5, No.4, Oct.1986, pp.582-596.

[5] K. Keutzer, "DAGON: Technology Binding and Local Optimization by DAG Matching," *Proc. ACM/IEEE 24th DAC*, June 1987, pp.341-347.

[6] J.Ishikawa, H.Sato, M.Hiramine, K.Ishida, S.Oguri, Y.Kazuma, S.Murai, "A Rule Based Logic Reorganization System LORES/EX," *ICCD*, Oct.1988, pp.262-266.

[7] 平峰、石川、石田、小栗、村井：論理回路変換プログラム：LORES/EX、情報処理学会設計自動化研究会 38-2(1987.7.23).