設計自動化　45—10
(1988. 12. 16)

CMOS用多段論理合成

Multi-level Logic Synthesis for CMOS Technology

竹田　信弘　　三浦　順子　　神戸　尚志
Nobuhiro TAKEDA　　Junko MIURA　　Takashi KAMBE

シャープ株式会社　コンピュータシステム研究所
Computer System Laboratories　SHARP Corporation

あらまし　組み合わせ回路の合成手法の一手法である多段論理合成では、テクノロジーに独立な方法が用いられることが多い。本論文では合成する回路をCMOSテクノロジーに限定することによって、よりCMOSテクノロジーに適した多段論理回路を合成する手法について提案する。本文では、まずテクノロジーに独立した従来手法の問題点について述べ、次にCMOSテクノロジー情報の利用方法について説明する。さらに回路を多段化する際の共通因子の選択に用いる選択行列と、それを利用した共通因子の選択方法を提案する。最後に実験により本手法の効果を示す。

Abstruct　A multi-level logic synthesis is a technique to synthesize a combinational logic circuit. It usually synthesizes a combinational logic circuit independent of technology. In this paper, we propose a method of multi-level logic synthesis, which produces a combinational circuit more suitable for CMOS technology, by the use of the information about CMOS technology. And we introduce "Selection Matrix" to select a common divider used in factoring logic functions.

1．はじめに

組み合わせ回路の合成では、2段論理の簡単化、weak-division[1][2]による多段化、およびルールベースシステム等による局所的最適化を併用した手法が報告されている[3][4][5]。しかし、この手法においてはweak-divisionにおける共通因子の選択方法、テクノロジー情報の利用方法等、検討が必要な問題がある。

まずテクノロジー依存性導入時期に関しては、2段論理の簡単化・多段化においては、その回路を実現するテクノロジーとは独立に処理を行い、局所的最適化でテクノロジーに適した最適化を行うのが一般的である[6][7][8][9]。しかし多段化において、すでに論理回路の構造はほぼ決定されている。さらに局所的最適化によって得られる回路はその初期解に依存する。そのため回路を実現するテクノロジーの情報を利用して多段化を行えば、局所的最適化処理を行った後に得られる回路についても、そのテクノロジーにより適した回路を得る可能性がある。

さらにweak-divisionによる多段化では、選択される共通因子によって合成される論理回路の品質が大きく変わる。そのため、共通因子の選択方法が重要な問題となる。

本報告では、weak-divisionを用いた多段論理合成における、選択行列を用いた共通因子の選択、

およびテクノロジー情報の積極的な活用による最適化について提案する。さらに、実際にＣＭＯＳテクノロジーに特化した多段化処理を開発し、実験を行ったので、その結果について報告する。

２．システム構成

本論文で述べる多段化処理は、図１に示す論理合成システムに組み込まれて利用される。

論理式・真理値表・論理回路からなる機能論理図[12]から抽出された論理関数は、まず２段の積和形表現に変換され２段論理の簡単化アルゴリズム[14][15]を用いて論理の簡単化を行う。多段化はこの簡単化された論理関数に対して施される。多段化された論理関数は、セル割り付け・局所変換が行われ，最終的にセルのネットリストが合成される。

本システムでは、複合セルジェネレータ[13]を用いることによりセルライブラリ中にない複合ゲートも使用することができ、複合ゲートによる設計品質の向上を図っている。

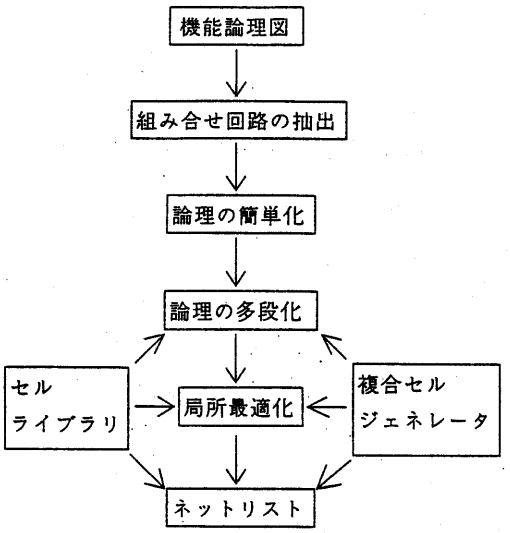


図1.論理合成システム

３．論理式のゲートモデル

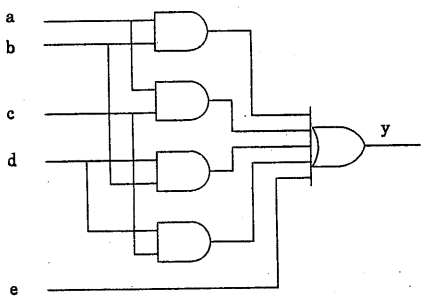
本章では、論理式のＣＭＯＳ論理ゲートによるモデルについて説明する。論理式を論理ゲートでモデル化することにより、多段論理合成においてテクノロジー情報を利用することが可能となる。

多段化処理における共通因子選択に際して、従来手法では論理式中の変数とその否定の数（リテラル数）、変数のファンアウト数、論理回路の段数などテクノロジー独立な指標を用いることが多い[3][5][7][10]。本手法では、ＣＭＯＳにより適した多段回路を得るために、論理式をＣＭＯＳ論理ゲートで実現した回路の面積と遅延時間を指標として用いる。まず論理式をＡＮＤ／ＯＲ等の基本ゲートでモデル化し、そのモデルの面積・遅延時間を求める。次に複合ゲートモデルについて面積・遅延時間を求める。複合ゲートを考慮することにより、合成された論理回路は、より多くの部分が複合ゲートに割り付け可能になる。

積和形でない論理式は、積和形のみの多段の論理式で表現可能であるので、与えられた論理式はすべて積和形の論理式とする。

$$y=a\cdot b+a\cdot c+b\cdot d+c\cdot d+e$$

a.積和形で表された論理式



b.AND/ORゲートで実現された論理式

図2.論理式のCMOSゲートによるモデル

３．１　基本ゲートモデル

積和形で表された論理式は、その和項をＯＲゲートに、その積項をＡＮＤゲートに対応させることにより実現することが可能である（図２）。ただし１つのゲートに入

力可能な信号の数に制限（ファンイン制限）があるため、１つの積項、１つの和項を実現するのに複数のＡＮＤゲート，ＯＲゲートが必要となる事がある（図３．ａ、ｂ）。さらに出力として接続可能なゲートの数にも制限（ファンアウト制限）があるため、制限を越える場合にはバッファゲートが必要となる（図３．ｃ）。

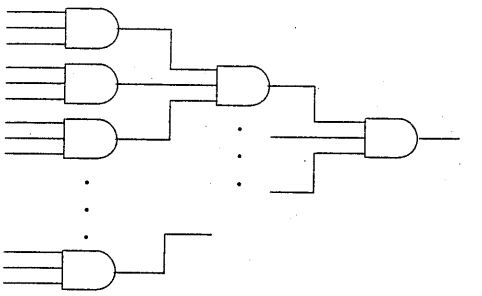
上記のように論理式をＡＮＤゲート・ＯＲゲート・バッファゲートでモデル化することで面積・遅延時間の計算が可能となる。ファンイン制限（ファンアウト制限）がｎで、要求された入力数（出力数）がｍのとき、必要なゲート数は、ｎ進木において葉の数ｍの木を実現するのに必要な葉以外のノード数に等しい。また遅延時間は木の高さに比例する。したがって論理関数ｙ＝ｆ（ｘ）の面積評価関数ａ（ｆ），遅延評価関数ｔ（ｆ）を計算すると次式になる。

$$a(f) = \alpha a \cdot (hf - nf) + \beta a \cdot (nf - 1) + \gamma a \cdot Ny \qquad \text{式（1）}$$

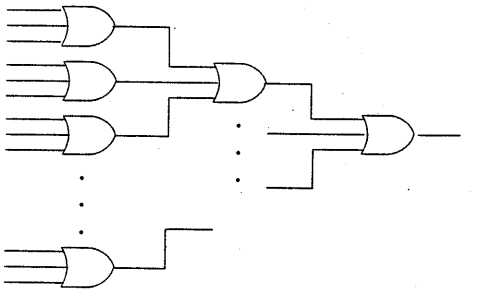
$$t(f) = \alpha t \cdot \log_{Ri} hfmax + \beta t \cdot \log_{Ri} nf + \gamma t \cdot \log_{Ro} Ny \qquad \text{式（2）}$$

αa：ＡＮＤゲートの面積パラメータ
βa：ＯＲゲートの面積パラメータ
γa：バッファゲートの面積パラメータ
αt：ＡＮＤゲートの遅延パラメータ
βt：ＯＲゲートの遅延パラメータ
γt：バッファゲートの遅延パラメータ
Ｒｉ：ＡＮＤゲート・ＯＲゲートのファンイン制限
Ｒｏ：バッファゲートのファンアウト制限
ｈｆ：ｆ（ｘ）中の変数の数（リテラル数）
ｈｆｍａｘ：ｆ（ｘ）の最大積項中の変数の数
ｎｆ：ｆ（ｘ）の積項の数
Ｎｙ：変数ｙのファンアウト数

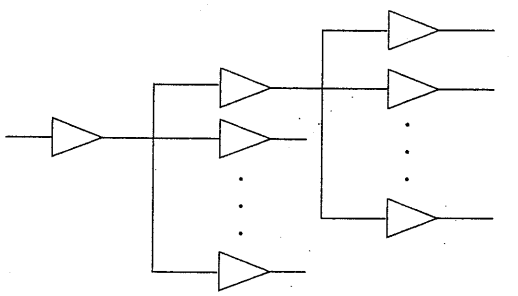
式（１）、式（２）の第１項、第２項、第３項はそれぞれＡＮＤゲート、ＯＲゲート、バッファゲートで実現された回路の面積・遅延時間である。面積パラメータ・遅延パラメータ・ファンイン制限・ファンアウト制限は実現すべきテクノロジーによって決まる定数である。また実際のセルレイアウトにおいてはＡＮＤゲート・ＯＲゲートの面積はほぼ同じになるので$\alpha a \fallingdotseq \beta a$とする。



a. 積項のANDゲートによるモデル



b. 和項のORゲートによるモデル



c. バッファゲートによるモデル

図3 論理式のCMOSゲートによるモデル

３．２　論理関数の複合ゲートモデル

複合ゲートで実現された論理式は基本ゲートの組み合わせで実現された場合に比べ、一般に面積が小さく遅延時間も短くなる。しかし実際に論理式をセルに割り付ける際に、用意されている複合セルの種類が少なければ多段化で複合ゲートを考慮する効果が小さくなる。複合ゲートによる面積・遅延削減を効果的に利用するため、

複合セルジェネレータを用いることにする。複合ゲートのモデルは、この複合セルジェネレータによって生成される複合ゲートについて考える。

論理式を複合ゲートで実現するには、その論理式が単調であることや電気的制約・レイアウト的制約を満たしている必要があり、まず与えられた論理式が複合ゲートで実現可能であるかどうかを判断しなければならない。単調であること以外に論理式が満たさなければならない制約には、

①電気的制約

直列接続されるトランジスタの段数が増えると複合ゲートの出力抵抗が増加し、さらには正常動作しなくなるなどの不具合が生じる。

②レイアウト的制約

複合セルジェネレータが生成するセルには、高さ・幅ともに制限があり、トランジスタ間の結線要求が複雑になるとセル内での配線が困難になり、セルジェネレータによるセル生成が不可能となる。

等がある。論理式がこれらの制約を満たし、セルジェネレータによるセル生成が可能かどうかを判断するのは困難である。しかし論理式の入力変数の数・積項の数・和項の数等から複合セルによる実現可能性を推定することは可能である。そこで与えられた論理式の複合ゲートによる実現可能性を判断する関数として次式を用いる。

$$Jcmp(hf, hfmax, nf) = (hfmax < Ns) \wedge (nf < Ns) \wedge (hf < Ny) \quad 式(3)$$

Ns：直列接続トランジスタの段数
Nv：入力数制限

多段化を行う際、上式が真のときには、与えられた論理式が複合ゲートで実現可能であるとみなす。

複合セルジェネレータで論理関数f（x）の複合ゲートを生成したとき、その面積と遅延の見積もりは次式で近似される。

$$ac(f) = \alpha ca \cdot hf + \beta ca \cdot hf / nf + \gamma ca \cdot hfmax + \delta ca \cdot nf \quad 式(4)$$

$$tc(f) = \alpha ct \cdot hmax + \beta ct \cdot nf \quad 式(5)$$

なおαca、βca、γca、δca、αct、βctは、複合セルジェネレータによって決定される定数である。

基本ゲートによる面積・遅延の評価が同じ値の共通因子の候補があった場合、複合ゲートで実現可能な因子を共通因子とする。さらにどちらも複合ゲートで実現可能であれば、その面積・遅延時間の小さい因子を共通因子に選ぶ。

4．論理式の多段化

本章では、論理式の多段化について説明し、3章で述べたモデルに従って多段化による論理回路の面積変化の評価関数を求める。この評価関数は5章で述べる共通因子の選択において使用される。

ｗｅａｋ－ｄｉｖｉｓｉｏｎによる多段論理回路の合成は、与えられた各論理式の共通因子を新たな変数で置き換えることにより行われる。例えば論理式

$$y1 = a \cdot c + a \cdot d + b \cdot c + b \cdot d + g$$
$$y2 = c \cdot e + c \cdot f + d \cdot e + d \cdot f + h$$

が与えられたとき(図4．a）、ｙ１の因子はａ＋ｂ、ｃ＋ｄ、ｙ２の因子はｃ＋ｄ、ｅ＋ｆとなる。共通因子ｃ＋ｄを新たな変数ｚ１で表し、ｙ１，ｙ２の置き換えを行う。

$$y1 = a \cdot z1 + b \cdot z1 + g$$
$$y2 = e \cdot z1 + f \cdot z1 + h$$
$$z1 = c + d$$

さらに多段化を行うことにより、次の論理式が求まる（図4．b）。

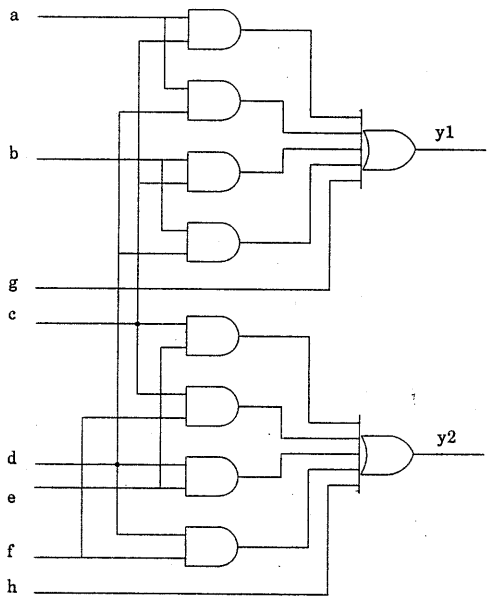
$$y1 = z1 \cdot z2 + g$$
$$y2 = z1 \cdot z3 + h$$
$$z1 = c + d$$
$$z2 = a + b$$
$$z3 = e + f$$

4．1　多段化処理

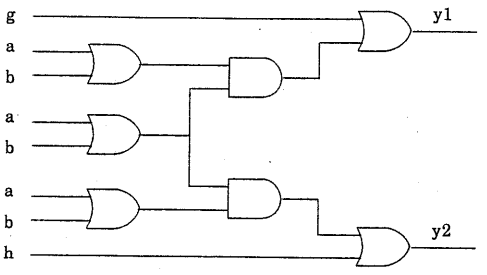
論理式の多段化処理は次の3つの処理に分かれる。

①与えられた各論理式からその論理式を因数分解可能な論理式を求める（因子の計算）。
②複数の因子の共通部分から、多段化することによる効果が最も効果の高い論理式を選ぶ（共通因子の選択）。
③共通因子を中間変数で表し、与えられた論理式をその中間変数で置き換える（共通因子による因数分解）。

多段化を行うと一般に回路の面積は減少し、遅延時間が増加する。従って与えられた時間制約を満たす間、①～③の処理を繰りかえし続けることにより、時間制約を満たす論理回路が合成される。



a. 2段論理回路



b. 多段論理回路

図4. 論理回路の多段化

4．2　多段化による面積削減見積もり

因数分解によって共通因子がくくり出されると論理式を実現する論理回路の面積が変化する。論理関数の因数分解による面積の変化には、

①共通因子による因数分解を行った論理関数の面積削減
②共通因子を表す論理関数を実現するための面積増加
③ファンアウト数が変化することによる面積の増減

がある。

これらの値について考える。多段化処理において、共通因子 $z = k(x)$ で論理式 $y = f(x)$ をくくり出す。このとき、論理式 $y = f(x)$ と、$z = k(x)$ で因数分解が行われた論理式 $y = f'(x, z)$ は、次のように表される。

$$y = f(x) = \sum_{i}^{n_c} \sum_{j}^{n_k} ci \cdot kj + r \qquad 式(6)$$

$$y = f'(x, k) = \sum_{i}^{n_c} ci \cdot z + r \qquad 式(7)$$

$$z = k(x) = \sum_{j}^{n_k} kj \qquad 式(8)$$

これを対応する論理回路を図5に示す。式（7）、式（8）で表される論理式の面積を式（1）から計算することにより、①～③の面積の増減を求めることが可能である。①は、$a(f) - a(f')$ になる。②は $a(k)$、③は $f(x)$、$f'(x, z)$ それぞれのリテラル数の差から計算が可能である。以上のことから①～③の面積増減は、次の値になる。

$$-\alpha a \cdot (nk \cdot hc + nc \cdot hk - hc - nc) \qquad 式(9)$$

$$-\gamma a \cdot \{(nk - 1) \cdot hc + nc \cdot (hk - 1)\} \qquad 式(10)$$

$$\alpha a \cdot (hk - 1) + \gamma a \cdot Nz \qquad 式(11)$$

5．共通因子の選択

本章では、論理式の因子のすべての部分集合から共通因子を選択することが可能となる選択行列を提案する。さらに選択行列を用いて、共通因子を選択する方法を示す。

与えられた論理式から共通因子を選択するためには、

複数の因子から共通部分を求め、求めた共通部分について評価値を計算し、評価値最大のものを選ぶことが必要である。多くの因子から共通部分を求められれば、それだけ多くの論理式に対して一度に因数分解が可能となる。

因子の共通部分を求めその評価値の計算を効率的に行うために、選択行列を導入する。選択行列を用いればすべての因子の共通部分から評価値最大の論理式を共通因子として選ぶことが可能になる。

5．1　選択行列

各因子を行に、因子の各積項を列に対応させ、積項が因子に含まれる場合には対応する行列の要素を1、含まれない場合は0とした行列を選択行列と定義する。さらに選択行列の0要素を含まない部分行列をブロックと定義する。このとき、因子の共通部分はブロックと1対1に対応する。例えば3つの因子

a・b・c＋a・d＋c・e
a・d＋c・e＋f
a・d＋c・e＋g

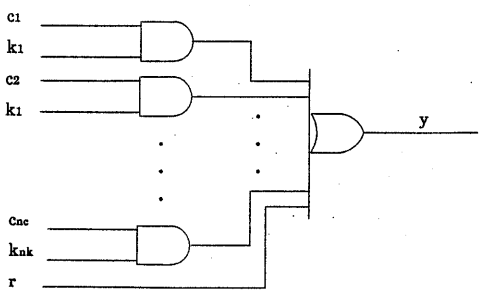
があったとき、行1、2、3には上の3式をそれぞれ対応させ、各列には積項a・b・c、a・d、c・e、f、gをそれぞれ対応させると、選択行列は図6になる。この選択行列のブロックには、

（行；1、列；1、2、3）
（行；2、列；2、3、4）
（行；3、列；2、3、5）
（行；1、2、3、列；2、3）

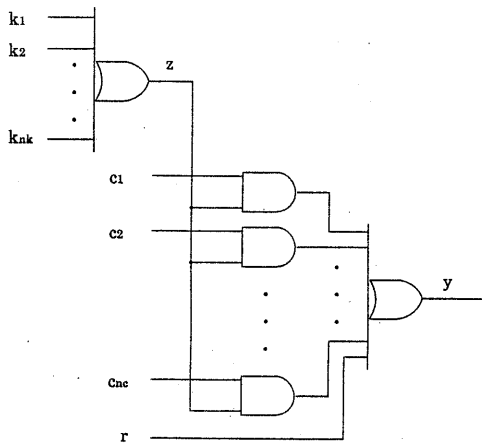
等があり、これらのブロックに対応する共通部分は、

a・b・c＋a・d＋c・e
a・d＋c・e＋f
a・d＋c・e＋g
a・d＋c・e

である。



a. くくり出し前の論理式



b. くくり出し後の論理式

図5. 共通因子のくくり出し

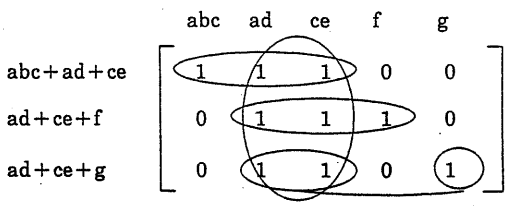


図6.選択行列と部分行列

5．2　選択行列の重み

選択行列の行・列・要素それぞれに重みを与え、ブロックの重みをそれに含まれる行・列・要素の重みの和とする。選択行列の行・列・要素に適当に重みを与えることにより、ブロックの重みを、そのブロックに対応する論理式を共通因子として多段化を行ったときの評価値と等しくすることが可能である。式（9）・（10）・（11）で与えられる多段化により、削減される面積を評価値として採った場合の重みの与えかたについて述べる。

（1）行の重み

行には、共通因子によって因数分解が行われた論理式を実現する回路の面積を重みとして与える。

$-\alpha a \cdot (nc+hc)$　　式（12）

（2）列の重み

列には、共通因子の積項を実現するのに必要な回路の面積を重みとして与える。

$-\alpha a \cdot hk_i+\gamma a \cdot hc \cdot (nk-1)/nk$　式（13）

（3）要素の重み

要素には、対応する因子の積項により削減される面積を重みとして与える。

$\alpha a \cdot (nc \cdot hk_i+hc)+\gamma a \cdot hk_i \cdot (nc-1)$
　　式（14）

5．3　重み最大のブロックの計算

5．2章で述べたように選択行列に重みを与えたならば、共通因子の選択は、重み最大のブロックを求めることに帰着する。重み最大のブロックを求めるためにグリーディ法を用いる[11]。グリーディ法は、種となる行（列）をまず求めその行（列）からなるブロックを列（行）方向、行（列）方向に順に拡張していって重み最大のブロックを求めるヒューリステックなアルゴリズムである。そのアルゴリズムを図7に示す。

```
SELECT（I：選択行列）
{
S0～S4：ブロック
    S1←Iの重み最大の列からなるブロック
    S2←Iの重み最大の行からなるブロック
    S0←0
    repeat {
        S3←S1
        S4←S2
        S1←S4の列方向への拡張
        S2←S3の行方向への拡張
        S0←max {S0, S1, S2}
    } until (S1=S3 or S2=S4)
    return (S0)
}
```

図7．重み最大のブロック選択アルゴリズム

6．　実験結果

本章では、多段化におけるテクノロジー情報利用の効果を調べるために行った実験について述べる。

実験では、選択行列に5章で述べたCMOSゲートモデルの面積減少による重みを与えた場合と、論理関数中に現れる変数の数（リテラル数）の減少による重みを与えた場合に対して、種々の回路で比較を行った。実験結果を表1に示す。表のゲート数は多段化された論理式を単純にセル割り付けしたときの2入力NANDゲート換算のゲート数である。ゲート数（1）はリテラル数最小化による結果、ゲート数（2）が本手法による結果である。ゲート数（2）はゲート数（1）にくらべ、0～25%減少している。ゲート数に違いのなかった回路は、規模が小さく自由度が少ないので同じ回路が合成されたためと考えられる。

7．まとめ

本文では、選択行列による共通因子の選択方法、およびそれを用いてCMOSに特化した多段論理合成について述べ、実験によって多段化におけるテクノロジー情報利用の効果を確かめた。今後の課題を以下に挙げる。

回路名	入力数	出力数	ゲート数（1）	ゲート数（2）	減少率
a u g 1	1 6	8	3 5 2	3 2 0	9 . 1
f 2	4	4	4 4	4 4	0 . 0
h o l l a c s i	1 4	8	2 6 6	2 5 3	4 . 9
r d 5 3	5	3	8 2	8 2	0 . 0
r d 7 3	7	3	1 7 3	1 6 8	2 . 9
s e g 7	1 0	2 3	2 5 1	1 8 6	2 5 . 9

表１．実験結果

①処理速度の向上

選択行列から１つの最適因子を見付けたら、共通因子による因数分解を行い再度選択行列を作りなおしているが、局在性の高い回路では、選択行列のほとんどの部分は変化しない。選択行列の再利用を図ることにより処理速度が向上する。

②複合ゲートの割り付け

複合ゲートの割り付けには、多段化された論理式を直接複合ゲートに割り付ける方法、一度基本ゲートに割り付けた後に局所的最適化で複合セルに割り付ける方法等があり、効果的な複合セルの割り付け方法の検討が必要である。

参考文献

[1] R.K.Brayton,C.T.McMullen,"The Decomposition and factorization of Boolean Expression",ISCAS, pp.49-54,1982

[2] R.K.Brayton,C.T.McMullen,"Synthesis and optimization of multi-stage logic",ICCD,pp.23-28,1984

[3] A.J.DeGues,W.W.Cohen, "A rule-Based System for Optimizing Combinational Logic", IEEE Design & Test,pp.22-32,Aug.,1985

[4] K.Bartlett,W.Cohen,A.J.DeGues,G.Hachtel, "Syntesis and Optimization of Multi-level Logic Under Timing Constraints", ICCAD,pp.290-292,1985

[5] R.K.Brayton,R.Rudell,A.Sangiovanni-Vincentelli, A.R.Wang, "MIS:A Multiple-Level Logic Optimization System",IEEE,Trans.CAD,Vol.CAD-6, No.6,pp.1062-1082,Nov.,1987

[6] 榎本、中島、村井、笹尾、"組み合わせ論理回路合成システム－COMPO－"、情報処理学会設計自動化研究会資料、２０－１、１９８４年２月

[7] 遠藤、星野、唐津、"論理合成システムANGELの最適化手法"、情報処理学会設計自動化研究会資料、２３－５、１９８４年９月

[8] 南谷、"論理合成"、情報処理学会誌、Vol.２５，No.１０、１９８４年１０月

[9] 笹尾、東田、"論理合成システム：MACDAS"、情報処理学会設計自動化研究会資料、３４－１，１９８６年１０月

[10] K.Bartlet,W.Cohen,A.DeGues,G.Hachtel, "Synthesis and Optimization of Multilevel Logic under Timing Constraints",IEEE, Trans.CAD,Vol.CAD-5,No.4,pp.582-596,Oct.1986

[11] R.Brayton,R.Rudell,A.Sangiovanni-Vincentelli, A.Wang, "Multi-level Logic Optimization and The Rectangular Covering Problem",ICCAD, pp.66-69,1987

[12] 谷、他、"機能・論理設計支援システム"、情報処理学会第３６回全国大会"

[13] 山田、築山、白川、"ＣＭＯＳ論理セルジェネレータの一手法、信学会技報 Vol.87,No.149, CAS87-114,pp.47-54

[14] 笹尾、"ＰＬＡの作り方、使い方"、日刊工業新聞社、１９８６

[15] R,K.Brayton,G.D.Hactel,C.T.McMullen, A.Sangiovanni-Vincentelli,"Logic Minimizatioin Algorithem for VLSI Synthesis", Kluwer Academic Publishers,1984