設 計 自 動 化 60−14
（1991．12．12）

二線式論理回路の多段合成に関する一考察

高橋隆一　張 長江　南谷 崇

東京工業大学 工学部 電気・電子工学科

二線式論理は、１ビットの情報を、２本の信号線で伝達する論理設計手法である。否定要素を必要としないことから単方向性故障に対するフォールトセキュア性を保証できるなど高信頼化設計に有用であり、データ自身に時間情報を持たせられることから非同期式の超高速設計にも有用である。しかし、対になる関数は、符号語に関しては、共通の積項を全く含まず、除数の構成に指針がなければ、除算による多段化も有効ではない。対称項探査法は、代数除算を前提とし、シャノン展開の拡張に相当する関数の分解法を用いることで、対をなす関数が共有可能な除数を構成する手法である。これにより、従来技術のＭＩＳを用いるより、小規模な回路が得られた。

On Multilevel Logic Synthesis for 2-rail Logic Circuits

Ryuichi TAKAHASHI, Changjiang Zhang, Takashi NANYA

Department of Electrical Engineering
Tokyo Institute of Technology
2-12-1 Ookayama Meguro-ku Tokyo 152 Japan

The 2-rail logic is a design method to carry one bit of information by using two signal lines. The circuits are fault secure because of the inverter-free structure and the code-words enable us to implement high-speed-asynchronous-systems. Yet, prior simplification algorithms based on coverage of cubes can not be efficient like the divisions that require appropriate kernels, since the dual expressions are always disjoint for code-words.An algorithm called STS (Symmetrical Terms Scanning) - method searches common divisors by using a calculation called ESE (Extended Shannon Expansion) that enables the STS-method to generate better results than a typical prior synthesizer called MIS.

1 はじめに

二線式論理(2-rail logic)は、1ビットの情報を、2本の信号線（2ビット）で表現する論理設計手法である。否定要素を必要としないことからフォールトセキュア性を保証できるなど高信頼化設計に有用であり[5]、00の入出力によって回路が休止している相を表すことで、データ自身に時間情報を持たせられることから非同期式の超高速設計にも有用である[6, 7]。従って、その論理最適化は重要である。

論理最適化の技術は、大きく、二段合成（two-level synthesis）と多段合成（multilevel synthesis）に分けることができる。前者は、PLAによる実現を前提とし、積項数の削減を行なうもので、すでに多くの研究がなされ、技術は、ほぼ、成熟している[1]。これに対し、多段合成は、ゲートアレーやスタンダードセルなどのデバイステクノロジーによる実現を前提とし、面積、クリティカルパス遅延の最小化を行なうもので、ここ数年、活発な研究が行なわれている[3, 4]。従来の二段合成は、プロセッサの制御部を実現するPLAが適用対象だと考えられるのに対し、多段合成は、データパスにも有効だと考えられる。

本稿では、上述したように、高信頼化設計、超高速設計に有用な二線式論理回路を対象に多段合成を議論する。二線式論理も、出力の数が2であるような、多出力関数とみなすことができるが、対をなす、fとf'の真理値表は、まったく重なるところがなく、両者をあわせると真理値表全体を埋めるという著しい性質をもっている。従って、fとf'の積項（キューブ）間の包含関係を利用した、従来技術の簡単化は適用できない。しかし、除算を用いた多段化による簡単化には議論の余地がある。除算は、除数と被除数の積が空でなければ常に可能である。カーネルとして知られる、共通の部分積項を含まない論理式による除算の有効性も報告されている[2]。しかし、カーネルの探査は、関数が、式によって、どう表現されているかに依存し、除数の構成に指針がなければ、良い結果は期待できない。

本稿では、はじめに、二線式論理回路と多段合成技術それぞれを概説し、二線式論理回路の多段合成を議論する。

2 二線式論理回路の概要

2.1 構成の基本

二線式論理では、1ビットの情報が、2本の信号線で表現される。各信号線は、0と1という2値のいずれかをとる。従って、00、01、10、11という組合せがある。このうち、01と10が、それぞれ、0、1を表す符号語に用いられる。

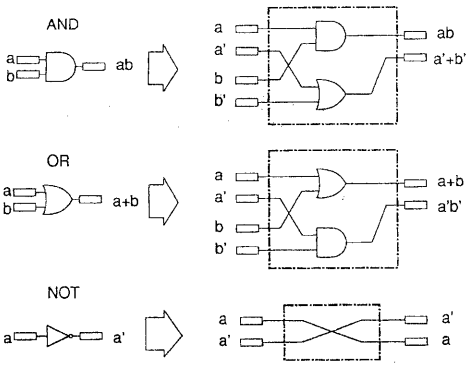

図1: 二線式論理回路の基本素子

図1に、二線式論理回路構成に用いられる、基本素子を示す。ANDゲートは、これと双対のORゲートとあわせた形、ORゲートは、これと双対のANDゲートとあわせた形で実現され、NOTゲートは、信号線をいれかえるだけで実現される。二線式の論理回路は、与えられた回路のAND,OR,NOTの各ゲートを、このような、基本素子に置き換えることで得られる。

図2に$f = (ab')'c + d'$を与えられ、これを、二線式論理回路として実現した例を示す。

2.2 高信頼化設計への応用

高信頼化設計に用いる場合、00と11の組合せは非符号語として扱われる。

二線式論理回路は否定要素を含まず、入出力は非順序符号をなしているため、任意の単方向故障に対する強フォールトセキュア（strongly fault secure）性が保証される。実際、否定要素を含まないことから、単方向故障が発生すると、出力は正しいか、故障と同じ方向への誤りが発生するかのいずれかになる。他方、出力は非順序符号で符号化されているため、誤りが発生すれば、それは必ず非符号語になる。[5]

2.3 超高速設計への応用

超高速設計に用いる場合、00の組合せは演算を行なっていない相(phase)を表すことに用いられる。すなわち、有効な符号語の間に無効な符号語を挟むことで、クロック信号を用いない、非同期式の超高速な設計が可能になる。実際、入力が、無効な符号語00から、有効な符号語01あるいは10に変化したことで、データの到着を検出することができ、出力が有効な符号語に変化したことで、演算の完了を検出することができる。[6]。

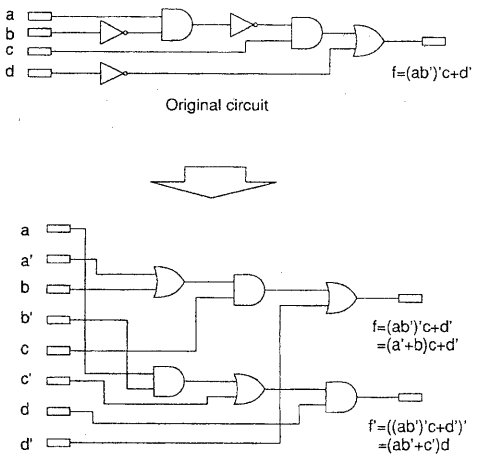

図 2: 二線式論理回路による $f=(ab')'c+d'$ の実現

2.4 多段化の対象とその性質

上述したように、二線式論理回路はきわめて有用だが、その構成方法から明らかなように、回路規模が、通常の2倍近くになるという欠点がある。この規模を、削減することが、本稿の課題である。

二線式論理回路の関数 f と f' は、AND-OR の、積和形二段論理回路として与えられるものとして、議論をすすめる。NOT ゲートを含まない、積和形二段論理で実現することは、常に可能である。仕様を表現する論理式のどの位置に否定があっても、De Morgan 則によって、必ず入力変数に対する否定にまで変形することができ、入力には、必ず、すべての変数とその否定が供給されている。

積和形二段論理回路とすることにより、問題は、見かけ上、一般の多出力論理回路の簡単化になる。しかし、対をなす関数 f と f' は、符号語の入出力に関するかぎり、決して同時に1になることがなく、いずれか一方のみが1になるという著しい性質をもっている。すなわち、

[性質 1] 二線式論理回路の構成に用いられる、2つの関数は、符号語入力に関するかぎり、両者を1にする、共通の積項を含まない。

このため、ある積項が、複数の関数の実現に係わっていることに注目した、従来技術の多出力関数簡単化手法は適用できない。

この性質は、0 0入力を考慮しても明らかに成り立つ。本稿では、「1 1入力を禁止し、don't care として扱う」ことは考えない。

以下、共通な積項がなくても可能な、除算を用いた多段合成の従来技術を概説し、二線式論理回路への適用を議論する。

3 多段合成の基本技術

二段合成が、PLA による実現を前提とし、積項数の最小化を行なうのに対し、多段合成では、ゲートアレーやスタンダードセルによる実現を前提として、面積の最小化が行なわれる。ここでは多段合成の基本技術 [2, 3] を概説する。

3.1 論理関数の扱い

リテラルは、変数あるいはその否定である。積項 (product term) はリテラルの集合で表現できる。リテラルの集合はキューブ (cube) といわれる。積和形論理式 (sum-of-products) はキューブの集合で表現できる。たとえば、{ { a} ,{ b,c'} } は、a+bc' を表している。主項は、それ以上リテラルを除くことのできないキューブとして特徴づけられる。積和形論理式は、どの積項（キューブ）を除いても関数の形が変わるとき、非冗長 (irredundant) だといわれる。

リテラルの数は、これを実現する回路規模（面積）の、良い評価尺度であることが知られている。

因子形 (factored form) は、次のように再帰的に定義される。1）リテラルは因子形である。2）因子形の和は因子形である。3）因子形の積は因子形である。積和形論理式は、しばしば、簡単のために、因子形で表現される。

3.2 代数除算とブール除算

2つの積和形論理式 f,g が同じ変数を含まないとき、積 fg は、代数積 (algebraic product) だといわれる。そうでないときは、ブール積 (Boolean prodct) だといわれる。代数積は、ブール代数の公理系における、可換則、結合則、分配則のみで計算できる。他方、ブール積の計算には、ベキ等則を含むすべての規則が用いられる。$(a+b)(c+d)=ac+ad+bc+bd$ は代数積、$(a+b)(a+c)=a+bc$ はブール積である。

積和形論理式 f を別の積和形論理式 g で割った商 (quotient) f/g は、$f=qg+r$ を満たす、最大なキューブの集合が表す積和形論理式 q として定義される。r は余り (remainder) といわれる。

qg を代数積に限った場合は代数除算 (algebraic division)、そうでない場合はブール除算 (Boolean division) といわれる。$f=ad+bcd+e$ を $f=(a+bc)d+e$ と割るのは代数除算、$f=(a+b)(a+c)d+e$ と割るのはブール除算である。代数除算の商は一意的だが、ブール除算の商は一意的でない。

表 1: $f = a(b(c+d)+e)(f+g)+h$ のカーネル

カーネル	コカーネル	レベル
$c+d$	abf, abg	0
$f+g$	abc,abd,ae	0
$b(c+d)+e$	af, ag	1
$(b(c+d)+e)(f+g)$	a	2
$f = a(b(c+d)+e)(f+g)+h$	1	3

3.3 カーネルとその性質

積和形論理式 f のカーネル (kernel) は、f を積項で代数除算した商のうち、特に、2つ以上の項からなり、各項に、同じリテラルを共通に含まない積和形論理式と定義される。

カーネルを用いて代数除算することで、積和形論理式から、因子形が得られる。

あるカーネルを得るのに用いられた積項は、このカーネルのコカーネル (co-kernel) といわれる。カーネルに対しては、「レベル」と呼ばれる指標が、1）カーネルを含まなければ0、2）レベル $N-1$ のカーネルを含むカーネルのレベルは N、と再帰的に定義されている。

レベルを見やすくするなどの目的で、カーネルは、しばしば、因子形を用いて表現される。

表1に、$f = a(b(c+d)+e)(f+g)+h$ のカーネルとコカーネルを例示する。

多出力論理関数とそのカーネルにつき、次の性質が成り立つことが知られている。[2, 3]

[性質2] 代数除算を前提とするならば、2つの積和形論理式 g と f は、それぞれの、すべてのカーネルの集合の要素どうしを比較した場合に、項が2つ以上一致している組がなければ、2つ以上の項からなる、共通の除数を含まない。

この性質は、代数除算を前提として、複数の論理式の間に、複数の項からなる、共通の除数が存在するかの判定に用いられる。

4 二線式論理回路の多段合成

4.1 基本的な方針

前述したように、符号語にかぎれば、f と f' は、両者を1にする、共通の積項を含まないが、除数と被除数の積が空でなければ除算は常に可能である。両者を割ることのできる、共通の除数が得られれば、この除数に対応する回路を共有できるため、回路規模の削減が期待できる。問題は、効果的な除数の構成方法である。図3に、この基本方針を示す。

Basic concept:

Extraction of efficient common divisors
for f and f' that cannot be 1 simultaneously

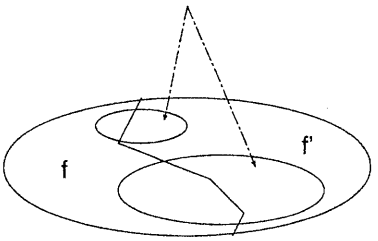

図 3: 簡単化の基本方針

4.2 前提とした除算とその性質

除算には、代数除算を用いるものとした。ブール除算を前提にすれば、より効果的な除数を探すことができると考えられるが、計算量を節約しようとしたためである。

代数除算を前提とする場合は、[性質2] を用いて、共通に用いることのできる除数が存在するかを調べることになる。この場合、f と f' の表現内容によっては、除数の発見に失敗する。周知のとおり、与えられた関数の、積和形論理式による表現は、主項のみからなる非冗長な形に限っても、一意的でない。そして、異なる表現には、異なるカーネルが対応している。このため、表現が適切でなければ、リテラル数削減に有効な除数を得ることはできない。

本稿では、符号語に議論を限っているが、ここで、非符号語を考慮した場合について簡単に触れる。二線式とすることで、入力変数は倍に増えているとみることができる。非符号語を考慮することは、ある変数とその否定が独立だと見なすことと同じである。つまり、非符号語を考慮すると、関数の表現に含まれていた $a', b'c', ...$ は、それぞれ、$a, b, c, ...$ と無関係な、別の変数 $x, y, z, ...$ として振舞うことになる。これについては、次の性質が成り立つ。

[性質3] 代数除算を前提とするかぎり、二線式論理回路の、符号語入力を前提とする、2つの、積和形論理式の間に、2つ以上の項からなる共通の除数が存在しないならば、非符号語入力を前提として得られる積和形論理式の間にも、2つ以上の項からなる共通な除数は存在しない。

代数除算を前提として、各変数とその否定の両者が混在する論理式を対象に、共通の除数が存在するかの判定

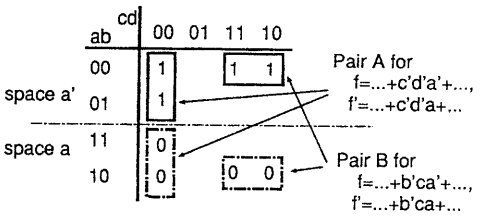

図 4: 対称な主項の組

が行なわれる場合には、ある変数の否定が、もとの変数の補元であることは考慮されない。このことは、ある変数の否定は、非符号語入力を考慮する以前から、もとの変数と無関係だとして判定が行なわれていたことを意味する。ある変数の否定が、もともと、この変数とは無関係な別の変数とみなされていた以上、これを別の変数に置き換えても、共通な除数が存在するかの判定結果は変わらない。

4.3 除数の求め方

ここでは、共通の除数が存在するとしたら、どのような性質を持つかを考え、その探し方を考察する。

まず、共通の除数が存在したとすると、対応する2つの商について、つぎの性質が成り立つ。

[**性質 4**] 双対な関数 f と f' の積和形論理式が、ある共通な積和形論理式を含んでいるとすると、対応する商の積は空である。すなわち、g, h, i, j, k を積和形論理式としたとき、f と f' が、

$$f = gh + j$$
$$f' = gi + k$$

の形で表されるならば、$hi = 0$ である。

この性質は、$ff' = 0$ でなければならないことから明らかである。h、i はただひとつの変数でもありうるし、$a + b$ と $a'b'$ のような、積和形式でもあり得る。

このことから、共通な除数は、両者を、互いに他の否定になるような積和形論理式で割ることで、見い出せることがわかる。そこで、シャノン展開の拡張を考え、次のように定義する。

[**定義 1**] 積和形論理式 f の

$$f = gv + hv'$$

ただし、g, h, v は積和形論理式であり、g, h は積項 v に含まれる変数を含まないものとする。

という展開を拡張シャノン展開と呼び、積和形論理式 v を分離式と呼ぶ。

分離式 v がただ1つの変数からなるとき、拡張シャノン展開は、この変数に関する、シャノン展開になる。分離式が n 変数の積項だった場合は、$v = x_1x_2...x_n$ に対し、$v' = x_1' + x_2' + ... + x_n'$ となる。$v = x_1x_2 + x_3$ ならば、$v' = (x_1' + x_2')x_3'$ である。

さらに、共通の除数を見い出すために、拡張シャノン展開における対称性を次のように定義する。

[**定義 2**] 同じ分離式を用いた、2つの拡張シャノン展開

$$f_1 = g_1v + h_1v'$$
$$f_2 = g_2v + h_2v'$$

において、g_1 と h_2 あるいは h_1 と g_2 に共通に含まれる積項に、対応する v、v' を掛けて得られる積和形論理式は対称であるといい、共通に含まれる積項を対称項という。

図4に、$f_1 = f, f_2 = f'$ における、対称な積和形論理式、この場合は主項、の組を例示する。図4において、f と f' は、分離式 a によって（拡張）シャノン展開されており、A と B、2組の対称な主項が存在する。前者においては、f における $a'c'd'$ と f' における $ac'd'$ が対称であり、$c'd'$ が対称項になっている。後者においては、$b'c$ が対称項である。

拡張シャノン展開によって、多くの対称項がえられれば、2つの関数の、両者を代数除算可能な2つ以上の項からなる、それだけ多くの最小項を含む除数を求めることができる。すなわち、

[**性質 5**] 同じ分離式を用いた、2つの拡張シャノン展開で見い出される、対称項の和は、代数除算を前提とし、この分離式については最大の除数、つまり、この分離式を除数とする商である。

たとえば、v につき、g_1 は最大である。実際、g_1 以外にも、av という積項があったとすると、それは、h_1v' に含まれていなければならない。しかしこれは空である。[**定義 1**] において、g と v の間、h と v' の間に同じ変数が含まれないことは、この展開を用いて得られる除数によって、代数除算が可能なことを保証している。

図4の場合、このような除数、この場合はカーネル、が $... + c'd' + b'c + ...$ と求まる。a' と a は、この場合、それぞれの関数における代数除算のコカーネルになっている。

4.4 対称項探査法

ここでは、上述した指針で、二線式論理回路を代数除算によって多段化する一手法として、対称項探査法を提案する。

代数除算を前提とした場合は、表現内容によっては、有効な除数を見い出せない、すなわち、多くの対称項が

```
対称項探査法(積和形論理式の組){
  for (すべての被覆につき) {
    被覆を修正
    対称項に注目して
    除数を抽出
    代数除算
    共通キューブを抽出
    利得を計算    }
  利得最大の多段化を選択
  得られた式の中から,変数の組が同じ
  積和形論理式の組を抽出
  for (すべての組につき) {
    対称項探査法(積和形論理式の組) }}
```

A

図 5: 対称項探査法の再帰的手続き

存在していても、それを見い出せるとは限らないことはすでに述べた。

対称項探査法は、関数の被覆 (cover) を変え、対称項を増加させる。

図5に、対称項探査法の、再帰的な、手続きを示す。

図にAと記した部分が中心的な試行錯誤の過程である。すなわち、被覆の形を様々に変えて、対称項を探査する。被覆ひとつに対し、複数の対称性がありうる。つまり、積和形論理式の、あるいくつかの積項の組には、あるひとつの分離式に対する対称性があり、別のいくつかの積項の組には、別の分離式に対する対称性があるという状況は一般的である。どの対称性にも該当しなかったために、除数として選ばれなかった主項からは、共通なキューブ（積項）を探す。利得はリテラル数の減少で評価する。Aの動きは、二段合成技術において被覆の方法を変える変形（reshape）と同じだが、対称項の数を増加させて、除数の共有によるリテラル数削減の効果を大にすることを目的とする点が相違している。

対称性の判定は、可能なすべての拡張シャノン展開に対して行なうことが望ましいが、分離式に用いる積和形論理式の変数の数を、分離の「クラス」と呼んで、適宜制限する。クラスが1なら、1変数からなる分離式に限り、$a, b, c, ...$と、順に変数を選んで、対称項を探査する。判定結果は、複数の分離式（対称性）に対するそれを記録しておき、それぞれに対応する代数除算を行なう。つまり、ひとつの被覆に複数の対称性があるなら、積和形論理式のある部分はある除数で、別のある部分は別の除数で代数除算する。

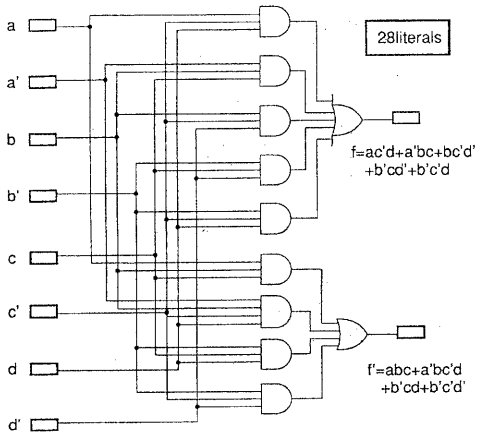

図 6: はじめの仕様

除算によって、変数の数が減少した、新たな、積和形論理式が得られる。これらが、はじめの組と同じような性質をもつという保証はないが、同じ変数の組に対する積和形論理式は抽出し、再帰的に対称項探査法を適用する。これは、「たがいに共通の積項を持たない場合に効果的である」という特徴を強めるためである。特に、パリティ関数に適用した場合は、2変数のEXORにまで、再帰呼びだしが続く。

4.5 具体例

図6に、$f = ac'd + a'bc + bc'd' + b'cd' + b'c'd$、$f' = abc + a'bc'd + b'cd + b'c'd'$を示す。ここでは、この回路の多段化を例に、対称項探査法を説明する。はじめのリテラル数は28である。

図5の、Aにおける試行錯誤の結果、図7に示したような、abc'とabc、$b'c'd$と$b'cd$という、2組の対称な主項が選ばれる。分離式はc'である。真理値表の$f = 1$はf、$f = 0$はf'を表している。$b'c'd'$と$b'cd'$も対称だが、分離式はcであり、上記とは別の対称性である。

被覆 (cover) が変わったことで得られた$a'bc'$と、もとからあった$a'bc$とからは、$a'b$が共通なキューブとして抽出される。

図8に、この様子を示す。abc'とabc、$b'c'd$と$b'cd$という2組は、c'とcというコカーネルに割られて、$ab + b'd$というカーネルを作る。$b'c'd'$と$b'cd'$の組は、1組であるため、カーネルにはならず、$b'd'$という共通キューブになる。これを作ることで、リテラルは減少しないが、対称だったため、形式的に除数とされている。二線式論理回路の場合、否定要素を含めないため、$c + d'$と$c'd$の組が互いに他の否定であるという性質は利用できない。

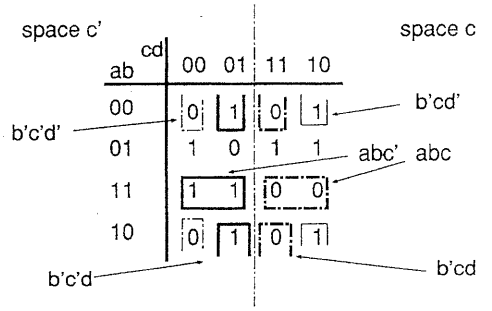

図 7: 仕様の中の対称な主項

上述した、はじめの呼びだしで、$ab+b'd$ という新たな式が作られたが、他に同じ変数の組からなる式は得られなかったため、再帰呼びだしは行なわれず手続きは停止する。

図９に合成結果を示す。リテラル数は、２８から、２２に減少している。

4.6 従来技術との比較

従来技術 [2, 3] を用いると、図６に例示した f からは、$(b)*(a'c+c'd),(b')*(cd'+c"d),(c)*(a'b+b'd'),(c')*(ad+bd'+b'd),(c'd)*(a+b'),(d')*(bc'+b'c)$ というカーネルが生成される。はじめの括弧内は、コカーネルである。f' からは、$(b)*(ac+a'c'd),(b')*(cd+c'd'),(c)*(ab+b'd),(c')*(a'bd+b'd'),(d)*(a'bc'+b'c)$ が生成される。後者には、対称項探査法で得られる $ab+b'd$ が含まれているが、前者から、これは得られない。

図１０に示す回路は、従来技術の実現として代表的な、ＭＩＳ [2] の合成結果である。リテラル数が２６で、対称項探査法の２２より多い。しかも、本質的な問題点ではないが、この結果はNOTゲートを含んでいる。これは、合成の過程において、過渡的に得られる式の否定を用いると回路規模を小さくできるという性質を利用しているためである。しかし、NOTゲートを含めば、フォールトセキュア性が成り立たず、二線式論理回路として適切な合成結果とはいえない。

従来技術でも、f を、$abc'+a'bc+a'bd'+b'cd'+b'c'd$ と変形しておくと、$(a'b)*(c+d),(b)*(ac'+a'c+a'd'),(b')*(cd'+c'd),(c)*(a'b+b'd'),(c')*(ab+b'd),(d')*(a'b+b'c)$ というカーネルが得られる。もちろん、この場合でも、$ab+b'd$ が除数として選ばれるとは限らない。実際、選ばれずに、やはりNOTゲートを含む、リテラル数２６という結果に終る。

図１１に別の例を示す。対称項探査法では、リテラル数が３３から２５に減少するが、ＭＩＳでは、NOTゲートを含めているにもかかわらず、リテラル数３０という結果になる。

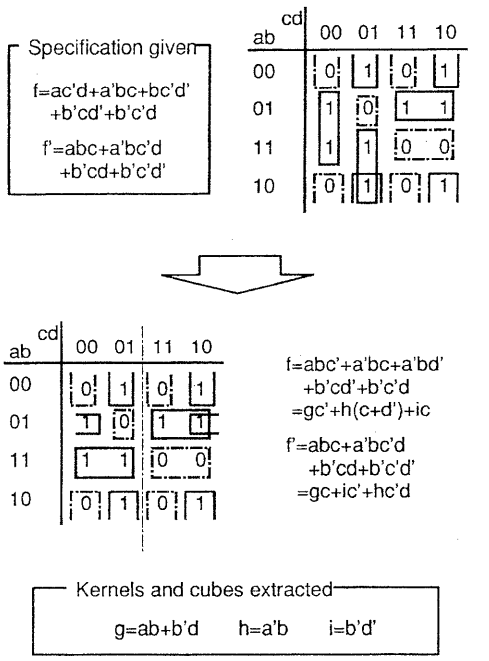

図 8: 対称項探査法によるカーネルとキューブの抽出

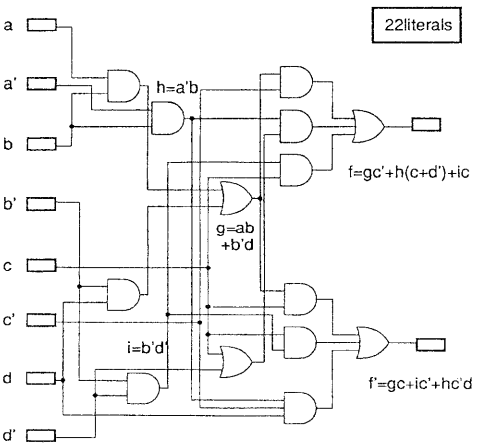

図 9: 対称項探査法による合成結果

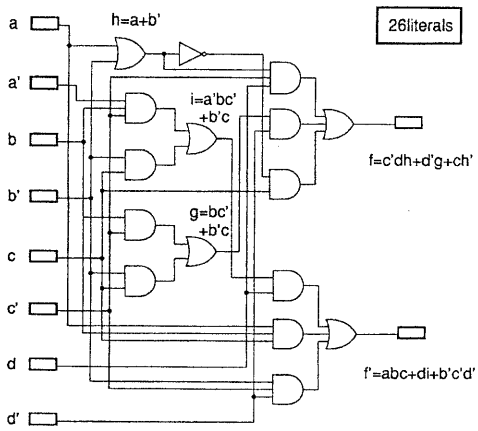

図 10: ＭＩＳによる合成結果

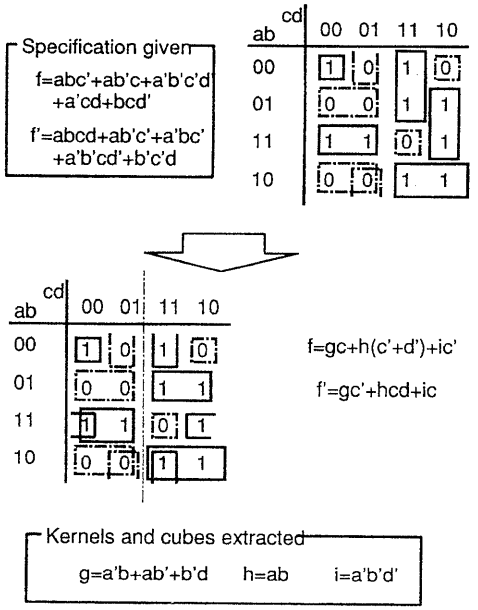

図 11: $f = abc' + ab'c + a'b'c'd' + a'cd + bcd'$の場合

5　まとめと今後の課題

本稿では、対をなす関数の被覆を変えて、二線式論理回路の多段化に有効な除数を構成する手法につき考察した。具体的には、シャノン展開を拡張し、二線式論理回路において対をなす関数に適用することで、代数除算を前提とする除数を効果的に探査する手法を提案し、これにより、従来技術として代表的なＭＩＳより優れた結果の得られることを示した。

本手法は、はじめの仕様と除算の結果を比較すると、ブール除算になっており、代数除算によってブール除算を行なう方法とみることもできる。本手法は、互いに、共通のキューブを、殆んどあるいは全く、持たない多出力関数の、多段合成手法としての汎用性を有していると考えられる。

当面は、二線式論理回路、将来的には一般の回路を対象とした評価が、今後の課題である。

本研究の一部は、科学研究費補助金（一般研究Ｂ：課題番号０２４５２１５６）によるものである。

参考文献

[1] R. K. Brayton, G. D. Hachtel, C. McMullen, and A. L. Sangiovanni-Vincentelli : Logic Minimization Algorithms for VLI Synthesis, Boston:Kluwer Academic Publishers (1984)

[2] R. K. Brayton, R. R. Rudell, A. L. Sangiovanni-Vincentelli, A. R. Wang : "MIS: A Multiple-Level Logic Optimization System", IEEE Trans. CAD, Vol.CAD-6, No.6, pp.1062-1081 (1987)

[3] R. K. Brayton, G. D. Hachtel, and A. L. Sangiovanni-Vincentelli : "Multilevel Logic Synthesis",Proc. IEEE Vo.78, No.2, pp.264-300 (1990)

[4] 藤田昌宏："多段論理合成技術の動向"信学論 (A) Vol.J74-A, No.2, pp152-161 (1991)

[5] 南谷 崇:フォールトトレラントコンピュータ, オーム社 (1991)

[6] 山村良憲、片山徳康、川辺幸仁、南谷崇："非同期式制御モジュールと組合せ回路の一構成法"、信学技法 FTS91-17 (1991)

[7] 上野洋一郎、南谷 崇:" 2線2相2系方式による非同期式レジスタ間転送"、信学技法 FTS91-23 (1991)