設計自動化 34－2
（1986. 10. 21）

論理合成システム：MACDAS

笹尾 勤　　東田基樹
大阪大学工学部電子工学科

MACDASはファンイン制限のあるANDゲート,ORゲートおよびインバータからなる多段論理回路の自動設計システムであり，次のようにして論理合成を行なっている．1)与えられた仕様を，AND-OR２段論理回路に変換する．2)入力変数を二つずつ対にして、二変数関数発生器(TVFG)を用いた２段AND-OR論理回路に変換する．3)一部の出力に対しては、その関数を実現するかわりに関数の否定を実現し、ゲート数の少ない回路を求める．4)２段回路をファンイン制限のあるANDゲート,ORゲートおよびインバータからなる多段論理回路に変換する．5)最後に、ゲートレベルの論理変換により冗長なゲートを除去する．本システムはFORTRANおよびCで記述されており，MS-DOSを用いたパーソナル・コンピュータ上で動作する.20種類の関数を合成した結果を示す．

MACDAS: A LOGIC SYNTHESIS SYSTEM

Tsutomu SASAO and Motoki HIGASHIDA

Department of Electronic Engineering
Osaka University, Suita 565, Japan

MACDAS designs a multi-level circuit with fan-in limited AND and OR gates and inverters. In MACDAS, 1)a given specification is converted into an AND-OR two-level circuit; 2)input variables are paired to produce an AND-OR circuit with Two-Variable Function Generators; 3)some of the outputs are complemented to produce a circuit with fewer AND gates; 4)the circuit is transformed into a multi-level fan-in limited AND-OR circuit; 5)and, finally, the circuit is optimized by local transformations. MACDAS is programmed in FORTRAN and C, and runs on a personal computer using MS-DOS. Synthesis results for 20 functions are shown.(23rd ACM/IEEE DESIGN AUTOMATION CONFERENCE. p.p.86-93)

1. まえがき

LSIの論理設計において，誤りのない論理を短期間に合成することは極めて重要である.Programmable Logic Array(PLA)は，規則的構造をしており，自動合成が容易であるため，現在多数のLSIで利用されている[SAS 86a]. 論理設計にPLAを用いる理由は，1)規則的構造をしているため，論理設計が容易でほとんど自動的に行える．2)設計変更が容易．3)検査が容易などである．しかし，多段論理回路(つまりランダムロジック)は，AND-OR 2段のPLAよりも小型になる場合が多いので，特に性能を重視する場合や素子を大量生産する場合には，多段論理回路を使用することが多い．従来，多段論理回路の自動合成は極めて困難であり，自動合成で得られた多段論理回路の品質は，人手設計のものに比べ大きく見劣りするものであった．しかし，最近の論理合成技術の進歩により，人手設計に匹敵する品質の多段論理回路を合成するシステムが出現し始めている([END 84],[ENO 85],[DEG 85][BAR 85]).これらの技術を支える背景は

1) 大規模論理式の簡単化手法(ESPRESSO-Ⅱ[BRA 84],MINI-Ⅱ[SAS 84][SAS 86a]).
2) 多段論理回路合成の理論(Factering[DIE 78] TVFG[SAS 82],Weak division,Strong division [BRA 82]).
3) 回路を目的のテクノロジーにうまくマッピングするための変換手法(LORES[NAK 78],ISS[DAR 80], SOCRATES[DEG 85]).

等である．

本報告では，二変数関数生成器(TVFG:Two-Variable Function Generator[SAS 82])を使用した多段回路の合成法を述べ，パーソナルコンピュータ上に実現した論理合成システム(MACDAS)で種々の回路を合成した結果を報告する．

2. ＴＶＦＧを用いた論理設計

TVFGを用いた論理設計法では，入力変数を二つずつ組になるよう分割する．このとき各変数の組Ｘ＝(Ａ，Ｂ)は４つの値00,01,10,11をとる超変数を表す．

【定義2.1】超変数Ｘに対し，Ｘのリテラルを次のように定義しよう：

$X^s=0$ （$X \notin S$ のとき），$X^s=1$（$X \in S$ のとき）

ここで，$S \subseteq B^2$，$B^2=\{00,01,10,11\}$である．

Ｘのリテラルは16通り存在するが，$S=B^2$のとき$X^s=1$，$S=\phi$のとき$X^s=0$であり，これらのリテラルを自明なリテラルという．自明でないリテラルは14個ある．

【定義2.2】 $X_1^{S1}X_2^{S2}\ldots X_n^{Sn}$を項という．

$$f(X_1,X_2,\ldots,X_r)=\bigvee_{(S1,S2,\ldots,Sn)} X_1^{S1}X_2^{S2}\ldots X_n^{Sn}$$

を超変数の論理和形という．

【定理2.1】 任意の論理関数$f(x_1,x_2,\ldots,x_n)$は超変数の論理和形で表現できる．ここで$X_i=(x_{2i-1},x_{2i})$である．

図2.1にTVFGを示す.TVFGは，Ｘのすべての自明でないリテラルを生成する.TVFGを用いて図2.2に示すAND-OR二段論理回路を実現できる．

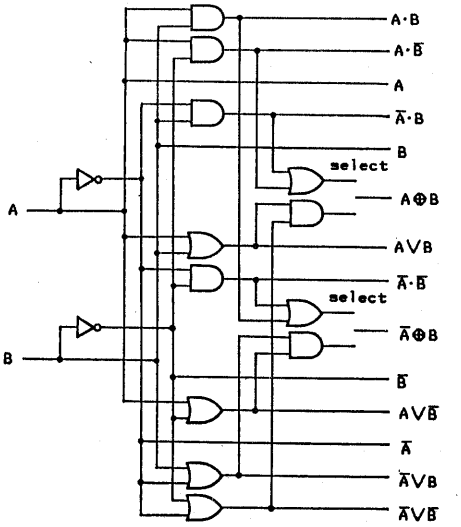

図2.1 2変数関数生成器(TVFG)

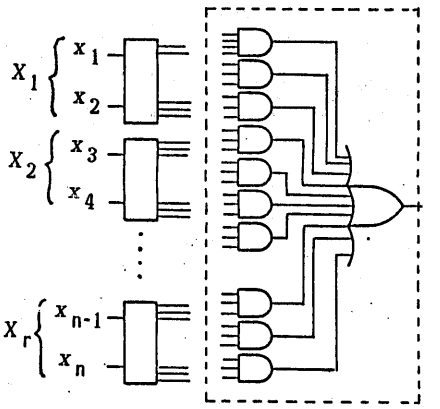

図2.2 TVFGを用いた2段AND-OR回路

【定理2.2】 TVFGを用いたAND-OR二段論理回路(図2.2)は,超変数の論理和形を実現する．

次に,TVFGを用いた最小論理回路の設計について考察しよう．通常ゲートのコストは接続のコストより大きいので,TVFGを用いたAND-OR最小二段論理回路を次のように定義できる．

【定義2.3】 TVFGを用いたAND-OR二段論理回路は以下の条件を満足するとき最小であるという．

1) ANDゲートの個数が最小．
2) 1)の条件の下で接続線数が最小．

論理和形の項数はANDゲートの個数に等しく，各項の自明でないリテラルの個数は各ANDゲートの入力線数に等しいので，最小論理和形を次のように定義できる．

【定義2.4】 論理和形は以下の条件を満足するとき最小であるという．

1) 項数が最小．
2) 条件1)の下で自明でないリテラル数が最小．

上の二つの定義により，TVFGを用いたAND-OR二段論理回路の最小化問題は，超変数の論理和形の最小化問題に帰着された．超変数の論理和形の簡単化はMINI[HON 74],MINI-II[SAS 84],ESPRESSO-MV[RUD 85]等で実行できる．

定理2.1と2.2は従来の二値論理回路の合成理論の自然な拡張になっている．従来の二値論理の場合，$2^2=4$個のリテラル$\{x,\bar{x},0,1\}$が存在するが，そのうち自明でないのはxと$\bar{x}$である．本論文では，xと$\bar{x}$を生成する回路を**OVFG**(One-Variable Function Generator)と呼ぶ．

TVFG内部の回路を無視すると,TVFGを用いたAND-OR二段論理回路は(通常の)OVFGを用いたAND-OR二段論理回路に比べ次のような特徴をもっている．

1) TVFGを用いた回路のゲート数はOVFGを用いた回路のゲート数より少ない．例えば，最小項の個数が102であるランダムな8変数論理関数の場合，ANDゲートの個数は平均32%少ない[SAS 81]．

2) TVFGを用いた回路のANDゲートの接続線数は,OVFGを用いたものに比べ少ない．これは，超変数の個数が通常の変数の個数の半分になり,ANDゲートの入力線数は高々n/2個となるからである．

TVFGを用いた二段論理回路の設計は，2ビットデコーダ付PLAの設計と基本的に同じであり，PLAの自動設計プログラムがそのまま流用できる．また，入力変数割当(論理式ができるだけ簡単になるように2ビットデコーダに入力変数を割り当てる)や，出力位相最適化(論理式ができるだけ簡単になるように，いくつかの出力関数を否定する)を行うと回路の大きさを10～30%減らすことができる．TVFGやOVFGを用いたAND-OR二段最小回路を出発点とし，ゲートの入力線数(ファンイン)を越えないように因子化(Factoring[DIE 69])の手法を用いて多数の多段論理回路を合成した結果,TVFGを用いた回路の方がOVFGを用いたものよりも，ゲート数や接続線数が少ないことが明らかになっている[SAS 82],[ENO 83],[ENO 84],[SAS 86b].

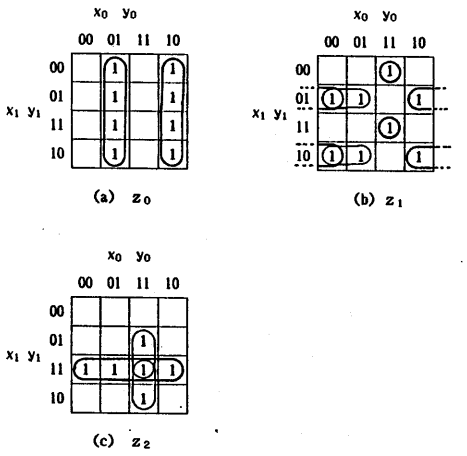

図2.3 2ビット加算器のカルノー図(OVFG)

【例2.1】 3入力のANDゲートと3入力のORゲートを用いて2ビット加算器を設計しよう．

$$\begin{array}{rccc} & x_1 & x_0 & \\ +) & y_1 & y_0 & \\ \hline z_2 & z_1 & z_0 & \end{array}$$

OVFGを用いた方法(従来の方法)

1) 2ビット加算器のカルノー図を図2.3に示す．これより簡単化された論理式

$$z_0 = x_0\bar{y}_0 \vee \bar{x}_0 y_0$$
$$z_1 = x_1\bar{x}_0\bar{y}_1 \vee x_1\bar{y}_0\bar{y}_1 \vee \bar{x}_1\bar{x}_0 y_1 \vee \bar{x}_1\bar{y}_0 y_1 \vee x_1 x_0 y_1 y_0 \vee \bar{x}_1 x_0\bar{y}_1 y_0$$
$$z_2 = x_1 y_1 \vee x_1 x_0 y_0 \vee x_0 y_1 y_0$$

を得る．この式を3入力のANDとORを用いて実現すれば，図2.4の回路が得られる．

図中，ゲート内の数字は3入力ゲートを用いて4入力ゲートあるいは6入力ゲートを実現するために必要なゲートの個数である．回路2.4のゲート数は22，接続線数は47である．

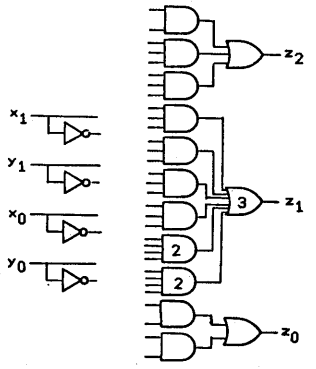

図2.4 OVFGを用いた2ビット加算器(2段回路)

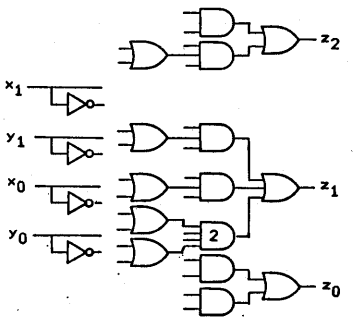

図2.5 OVFGを用いた2ビット加算器(多段回路)

2) 上の論理式を因子化すると，

$$z_0 = x_0\bar{y}_0 \vee \bar{x}_0 y_0$$
$$z_1 = (x_1\bar{y}_1)(\bar{x}_0 \vee \bar{y}_0) \vee (\bar{x}_1 y_1)(\bar{x}_0 \vee \bar{y}_0) \vee (\bar{x}_1 \vee y_1)(x_1 \vee \bar{y}_1)(x_0 y_0)$$
$$z_2 = x_1 y_1 \vee (x_1 \vee y_1)(x_0 y_0)$$

が得られる．この式から得られた回路を図2.5に示す．因子化によりゲート数は20に，接続線数は41に減っている．

TVFGを用いた設計法(著者の考案した方法)

1) 2ビット加算器の出力関数は，変数 $\{x_1, y_1\}$ 及び $\{x_0, y_0\}$ に関して対称である．そのため，超変数として $X_1=(x_1, y_1)$，$X_2=(x_0, y_0)$ とおく．本システムは，入力変数割当機能をもっており，この割当は自動的に求まる．図2.6のカルノー図から，論理式

$$z_0 = X_2^{(01,10)}$$
$$z_1 = X_1^{(01,10)} \cdot X_2^{(00,01,10)} \vee X_1^{(00,11)} \cdot X_2^{(11)}$$
$$z_2 = X_1^{(11)} \vee X_1^{(01,10,11)} \cdot X_2^{(11)}$$

を得る．この例からも判るように，多値論理式では，多くの場合，対応する二値論理式よりも積項数が少なくなる．

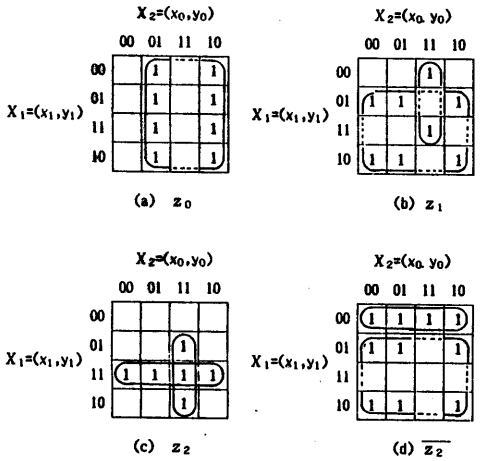

図2.6 2ビット加算器のカルノー図(TVFG)

2)次に，出力位相の最適化[SAS 84]機能を用いて出力位相($\bar{z}_2 z_1 z_0$)を求める．つまり，z_2のかわりに，その否定

$$\bar{z}_2 = X_1^{(01,10)} \cdot X_2^{(00,01,10)} \vee X_1^{(00)}$$

を実現する．この式の第1項は，z_1の第1項と同一なので，二つの出力で共有できる．図2.7にTVFGを用いた2ビット加算器を示す．

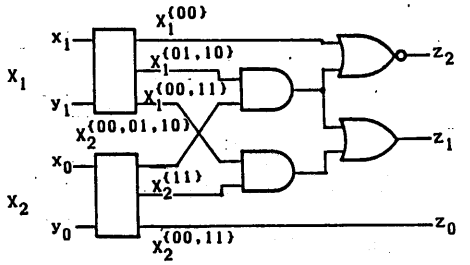

図2.7 TVFGを用いた2ビット加算器(2段回路)

3)次に，TVFGをANDとORに展開する．この操作をマクロ展開という．図2.1からわかるように $X^{(00,11)}$ や $X^{(01,10)}$ の実現法は，二通りある．このうち，得られる回路が出来るだけ簡単になる方を選ぶ．マクロ展開では，出力が他のゲートに接続されているもののみ生成し，不要なゲートは除去する．次に，TVFGの最適化を行う．これは，二つのゲートの出力が互いに補元をとるときに，実現が容易なゲートのみを実現し，その出力にインバータを接続することにより補元を生成する操作であり，他方のゲートは除去できる．例えば，図2.8では，$\bar{x}_1\bar{y}_1$ は $x_1 \vee y_1$ から生成し，$\bar{x}_1 \vee \bar{y}_1$ は $x_1 y_1$ から生成し，$\bar{x}_0 \vee \bar{y}_0$ は $x_0 y_0$ から生成する．このようにして，図2.9に示すようにゲート数10のTVFGを得る．

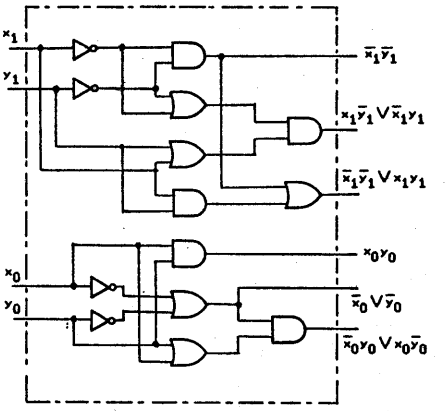

図2.8 マクロ展開後のTVFG

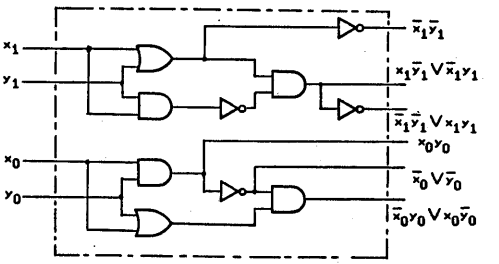

図2.9 最適化後のTVFG

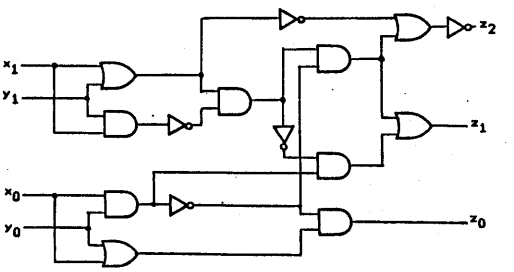

図2.10 最終的に得られた2ビット加算器

4)図2.10は最終的に得られた2ビット加算器で，ゲート数15,接続線数25である．

３．多段論理回路の生成

本章では，二段論理回路を多段に変換するための方法を述べる．

3.1 因子の抽出

図3.1(a)に示すAND-OR回路が与えられたとしよう．このとき，共通のリテラルをｗ個含むようなANDゲートがｈ個存在すれば，図3.1(b)の回路に変換できる．この回路ではANDゲートとORゲートがそれぞれ１個ずつ増えているが，入力段のANDゲートの入力線がhw本減っている．また，出力段のORゲートの入力線も減るため，ファンイン制限超過のための必要なゲートが減る．このため，全体のゲート数も減る場合がある．この操作により削減できる接続線数は，hw-(w+2)=w(h-1)-2である．ファンイン制限の下でゲート数を最小にする因子を求めるのは困難なので，本システムでは，接続線数を最小にする因子で代用している．w(h-1)を評価関数とし，この値を最大にする因子を分枝限定法で求める．

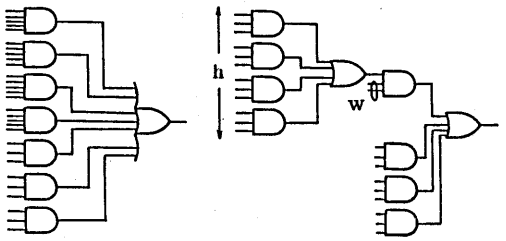

図3.1 2段回路の因子化

3.2 最適回路の選択

評価関数の値が最大の因子を見つけるごとに，その因子をもつ論理式を図3.2で示す三つの回路で実現し，コストが最小の回路を選択する．(a)はAND-OR回路,(b)は因子化を行ったAND-OR-AND回路,(c)はOR-AND回路である．

3.3 回路コストの計算

回路のコストを次のように定義する．

回路のコスト
＝(ゲートのコスト)+(接続のコスト)

ゲートのコスト
＝(ゲート数)×(ゲートのコスト係数)

接続のコスト
＝(接続数)×(接続のコスト係数)

ファンイン制限ｔのとき，ｎ入力ゲートを実現するためのゲート数は

NGATES(n,t)= 0　　n＝1
= 1+GTYPE*[(N-2}/{t-1)]　n≧2

で計算する．ここで,[a]はaの整数部を示す．また

GTYPE=1(NOR/OR出力をもつECLゲートの場合)
=2(TTL/MOSのNAND又はNORゲートの場合)

である．

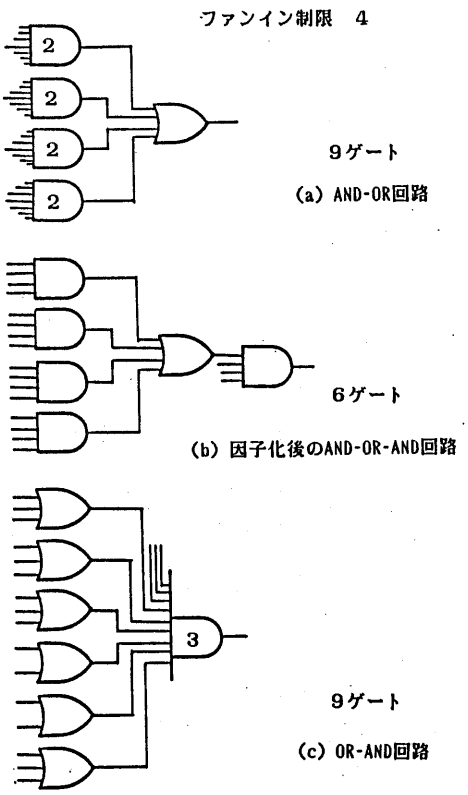

図3.2 最適回路の選択

４．ゲートレベルの論理変換

本章では，３章の方法で生成したAND，OR及びNOTから構成された回路に対して，ゲートレベルでの論理変換を，ゲートのファンイン制限数t(≧2)を考慮して行う．

4.1 基本操作

論理変換は，次の3つの操作を用いる．

(1)重複ゲート削除

入力信号及びその機能が同一のゲートがあれば1つのゲートのみで実現し，他のゲートは除去する(図4.1)．

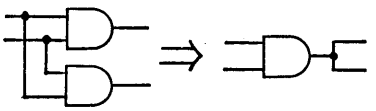

図4.1 重複ゲート除去

(2)ゲート併合

同種のゲートが直列に接続している場合，1つのゲートに併合する．ゲート併合により，論理段数が減り，また次に述べる因子共用化が行い易くなる(図4.2)．

(3)因子共用化

同種の2つのゲートに共通な入力線を2ゲート間

の因子，また因子の数を因子数と呼ぶ．因子数が２以上の場合，因子を入力とする同種のゲートを付加して，ゲートの接続線数を削減する(図4.3).

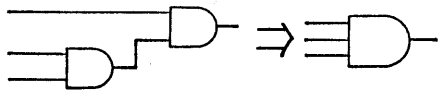

図4.2 ゲート併合

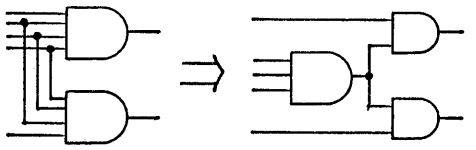

図4.3 因子共用化

4.2 ゲートレベル論理変換アルゴリズム

論理変換は，基本操作(1)～(3)を用いて，総接続線数を減らす方向で進める．各ステップでの総接続線数の減少を利得と呼ぶ．基本操作(1)～(3)の適用順序は，いろいろ考えられるが，本システムでは，次のアルゴリズムを用いている．

【アルゴリズム4.1】
(ゲートレベル論理変換アルゴリズム)
1)重複ゲートを除去する．
2)ゲートを併合する．
3)因子による分解をする．
4)重複ゲートを除去する．
5)2)～4)の処理で利得があれば，2)へ戻る．ここで，1)～4)の処理はそれぞれ回路全体に対して行い,3),4)の処理により,因子共用化が行われる．

4.3 因子の共用化アルゴリズム

因子共用化は，因子の選び方によって大きく解が変る．そのアルゴリズムを次に述べる．

【アルゴリズム4.2】 (因子による分解)
1)ゲートを，接続線数の大きい順に並び換えGに入れる．
2)Gから順にゲートG_iを取り出し，ゲートG_iに関する全ての因子を求めTに入れる．
3)ゲートG_iの入力線を最大かつt以上含んだ因子をTから1つ選ぶ．さらに，含まれなかった入力線に対して，因子が選べなくなるまで同様の処理を繰り返す．
4)3)で選ばれた因子によって，ゲートG_iを分解する．
5)Gのゲートがなくなるまで2)～4)の処理を繰り返す．

アルゴリズム4.2は，以下の2点の特徴を持つ．

(1)各ゲートごとの因子選択

アルゴリズム4.2では，1)で選んだゲートのみを4)で分解する．したがって，各ゲートは，他のゲートの因子の選択にかかわらず，独立に最大の利得の因子で分解できる．そして，次に重複ゲート削除により，因子共用化を行う．共用化されなかった因子はゲート併合により分解前に戻し，再度因子共用化処理にかけることで，論理変換の効果を増している．

(2)高速な因子選択

Weak Division[BRA 82]の方法では，全てのゲート間の因子からdisjointな最適の因子を得るために，多大な計算時間が予想される．また,ANGEL[END 84]の方法も，全てのゲート間の類似度を計算しなければならない．アルゴリズム4.2では，これらの点を改善するために，因子の選択はゲートごとに行い，その中から有効なもののみを取り上げる方法をとっている．そして，上の2点の特徴により，利得を減らすことなく高速な因子選択を実現している．

５．実験結果

実験には16ビットパーソナルコンピュータおよび，MS-DOS上で動く図5.1のようなソフトウエアシステムを用いた．

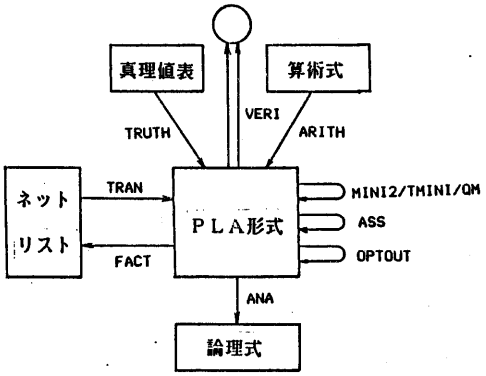

図5.1 MACDASにおけるソフトウエア

5.1 実験用データ

表5.1に使用データを示す．最初の11個は算術演算用あるいは特殊関数用のPLAで，いずれも,MACDAS附属の関数発生用プログラムARITHで生成できる．残りの９個は，制御用あるいはその他のPLAで，企業から提供されたため内容は不詳である．

5.2 積項数

表5.1に下記の４つの異なった方法で実現したAND-OR二段回路の積項数を示す．

1) OVFGを使用し，出力位相を考慮しない場合
2) OVFGを使用し，出力位相を考慮した場合
3) TVFGを使用し，出力位相を考慮しない場合
4) TVFGを使用し，出力位相を考慮した場合

この表からわかるように，全てのサンプルにおいて,TVFGを用いた回路では，OVFGを用いたものより積項数が少なかった．また，算術演算用あるいは特殊関数用PLAでは，殆どの場合，出力位相を考慮することにより積項数を減らすことができた．しかし，制御用ああるいは，その他のPLAでは，出力位相を考慮しても積項数は殆ど減らなかった．表中,POは出力位相最適化のため否定された出力数を示す．論理の簡単化にはMINI-Ⅱ，入力変数の割当にはASS，出力位相の最適化にはOPTOUTを用いた．

5.3 回路の面積

表5.2に上記の４つの条件の下で実現した多段論理回路の面積を示す．回路は2入力AND，2入力OR，およびインバータから構成され，ゲートの面積を１，インバータの面積を.5として計算している．また，変数は肯定のみ与えられ，否定は使用できないものと仮定した．VG2,bwを除いて，面積も,TVFGを用いたものがOVFGを用いたものより小さくなった．

5.4 実行時間

図5.2にRD73（7入力3出力の回路で，入力の１の個数を3ビットで表現する回路）の実行時間を示す.PC-98XA上で,81.5ゲートの回路を得るのに471秒かかった．

5.5 各処理による接続線数の減少効果

OVFGを用いたAND-OR2段論理回路を基にして，TVFGを用いたAND-OR2段論理回路，因子化，TVFGの最適化，ゲートレベルの論理変換の順で多段回路の生成最適化を行った．図5.3に各ステップにおける接続線数の変化を示す．OVFGを用いたAND-OR2段論理回路の接続線数を100としている．ファンイン制限数(t)は2である．また図5.3にOVFGを用いた場合の接続線数の変化も示す．この結果次のことが明らかになった．

1)TVFGを用いた回路はOVFGを用いた回路より，接続線数が少ない．
2)TVFG最適化及びゲートレベル論理変換により接続線数は，18%減った．

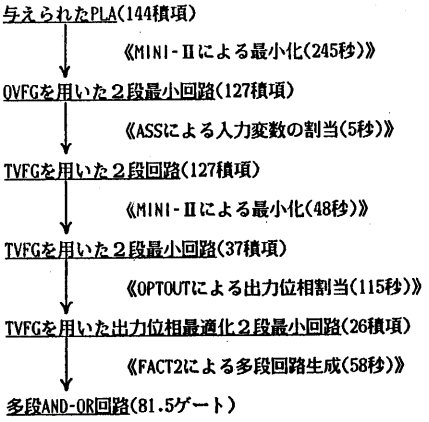

図5.2 RD73合成のための実行時間(PC98XA)

表5.1 AND-OR２段回路の積項数

入力データ				OVFGを用いた回路			TVFGを用いた回路		
回路名	入力数	出力数	積項数	出力位相単純	出力位相最適	P0	出力位相単純	出力位相最適	P0
ADR4	8	5	255	75	61	1	17	14	1
MLP4	8	8	225	126	112	2	89	77	3
NRM4	8	5	255	120	106	2	82	70	2
ROT8	8	5	255	57	48	3	41	35	3
SQR6	6	12	63	51	41	5	41	38	2
SYM9	9	1	420	87	72	1	27	27	0
RD53	5	3	32	31	22	1	12	10	1
RD73	7	3	141	127	93	1	37	26	1
SAO1	8	4	256	255	186	1	54	38	2
5XP1	7	10	75	67	59	2	47	41	2
RDM8	8	8	255	76	76	0	52	52	0
F2	4	4	12	8	8	0	6	6	0
VG2	25	8	110	110	110	0	88	88	0
BW	5	28	87	25	24	17	24	24	0
SAO2	10	4	58	58	38	2	38	28	2
DUKE2	22	29	87	86	86	0	76	76	0
GARY	15	11	167	107	107	0	92	92	0
X1DN	27	6	110	110	110	0	80	80	0
X6DN	39	5	121	81	81	0	63	63	0
Z4	7	4	59	59	45	1	16	13	1

表5.2 多段回路の面積

回路名	OVFGを用いた回路		TVFGを用いた回路	
	出力位相単純	出力位相最適	出力位相単純	出力位相最適
ADR4	76.0	69.5	39.5	39.5
MLP4	382.0	328.0	250.5	233.5
NRM4	343.0	293.0	260.5	234.0
ROT8	146.0	125.5	142.0	121.0
SQR6	150.0	130.5	113.0	109.0
SYM9	217.5	119.0	91.5	91.5
RD53	69.5	54.0	37.5	32.0
RD73	173.5	146.0	94.0	81.5
SAO1	305.0	279.5	129.0	116.0
5XP1	162.5	154.5	125.5	118.0
RDM8	150.0	150.0	138.0	138.0
F2	24.0	24.0	21.0	21.0
VG2	148.5	148.5	212.5	212.5
BW	130.5	123.0	123.5	123.5
SAO2	148.0	139.0	127.5	117.0
DUKE2	337.5	337.5	301.5	301.5
GARY	396.5	396.5	369.0	369.0
X1DN	181.5	181.5	140.5	140.5
X6DN	292.5	292.5	273.0	273.0
Z4	102.5	71.0	48.5	42.5

表5.3 接続線数の減少効果

ステップ	TVFG使用	OVFGのみ使用
OVFGを用いた AND-OR２段回路	１００	１００
TVFGを用いた AND-OR２段回路	４８	－－－
因子化後	４０	６２
TVFG最適化後	３７	－－－
ゲートレベルの論理変換後	３３	４２

6. 結論

1.TVFGを用いた回路はOVFGのみの多段回路に比べて，約35%少ない面積の回路を実現する。
2.算術回路に対して，出力位相割当ては有効である。
3.ゲートレベル論理変換により多段回路の面積を約18%減らせる。
4.MACDASは，パーソナル・コンピュータ上で動作し，短時間で、良い回路を実現する。

[参考文献]

[BAR 85] K.A.Bartlett and G.D.Hachtel, "Library specific optimization of multilevel combinational logic,"ICCD-85, pp.411-415, Oct.1985

[BRA 82] R.K.Brayton and C.T.McMullen, "The decomposition and factorization of Boolean expressions," International Symposium on Circuit and Systems, pp.49-54, 1982.

[BRA 84] R.K.Brayton and C.T.McMullen, "Synthesis and optimization of multi-stage logic," ICCD-84,pp.23-28, Oct. 1984.

[DAR 84] J.A.Darringer et.al,"LSS:A system for production logic synthesis", IBM J. RES. Devlop. Vol28. No.5 Sept.1984, PP.537-545.

[DEG 85] A.J.de Geus and W. Cohen,"Optimization of combinational logic using a rule-based expert system," Journal of IEEE Design and Test, pp.22-32, August. 1985.

[DIE 69] D.L.Dietmeyer and Y-H. Su,"Logic design automation of fan-in limited NAND networks,"IEEE Trans. Comput. Vol. C-18, No. 1. pp.11-22,Jan. 1969.

[DIE 78] D.L. Dietmeyer, Logic Design of Digital Systems,Allyn and Bacon,Inc.,Boston, 1978.

[END 84] 遠藤，星野，唐津，"論理生成システム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, 1984-09.

[ENO 83] 榎本，中島，村井，笹尾，"組合せ回路合成システム-ＣＯＭＰＯ-(1)，(2)，"情報処理学会第26回全国大会 6K-10,6K-11,pp.1401-1404, 1983-03.

[ENO 84] 榎本，中島，村井，笹尾，"組合せ回路合成システム-ＣＯＭＰＯ-"情報処理学会設計自動化研究会資料20-1, 1984-02.

[HON 74] S.J.Hong, R.G.Cain,and D.L.Ostapko, "MINI: A heuristic approach for logic minimization,"IBM J. Res, & Develop.pp.443-458, Sept. 1974.

[NAK 78] S.Nakamura, et.al,"LORES-Logic Reorganizaiton System," Proc.15th-Design Automation Conference. June 1978, pp.250-260.

[RUD 85] R.Rudell and A.L.M.Sangiovanni-Vincentelli,"ESPRESSO-MV:Algorithms for multiple-valued logic minimization," 1985 Custom Integrated Circuits Conference, pp.230-234, May 1985.

[SAS 81] T.Sasao,"Multiple-valued decomposition of generalized Boolean functions and the complexity of programmable logic arrays," IEEE Trans. Comput. Vol. C-30, pp.635-643, Sept. 1981.

[SAS 82] T.Sasao,"An application of multiple-valued logic to a design of masterslice gate array LSI," 12-th Inter.Sympo. Multiple-Valued Logic , pp.45-54, May 1982.

[SAS 84] T.Sasao,"Input variable assignment and output phase optimization of PLA's," IEEE Trans. Comput. Vol. C-33, No.10,pp.879-894, Oct. 1984.

[SAS 86a]笹尾 勤,PLAの作り方・使い方，日刊工業新聞社, 1986-04.

[SAS 86b]T.Sasao,"MACDAS:Multi-level AND-OR Circuit synthesis using two-variable function generators,"23-rd Design Automation Conference, June 1986.