三段 NAND ゲート回路の論理設計法について（II）
—— P－N 項法の改良 ——

宮腰　隆，松田　秀雄，大澤　一人，畠山　豊正

1．はじめに

本号の先の論文で，三段 NAND ゲート回路の計算機援用設計法の一つとして，P－N項法[1] を発表した。P－N項法はどのような関数も三段 NAND ゲート回路で実現するため，必ずしも最適な回路が得られるとは限らない。本論文では，まず，はじめにそのような回路の例を挙げて検討し，その改良法を述べる。

次に，計算機プログラムする際，項表現法を用いることについて述べる。先の論文では，真理値表をそのまま，ブールベクトルで表した。この方法によると，関数の否定を求めるときなど，各ビットの0と1を反転させるだけで得られるという簡単な面もあるが，次のような記憶量の上での難点がある。すなわち，n 変数の関数を表すブールベクトルのサイズは 2^n となる。また，P－N項法では，真理値表に基づく方法では許容項行列のために $2^n \times 2^n$ のサイズの記憶量を要し，P－N項最小被覆表にも 2^n に比例した記憶量が必要となる。n が大きくなると，2^n という数量は著しく大きくなり，20変数，30変数といった関数は到底扱えない。そこで，項表現を用いて，P－N項法を改良する手法を述べる。

最後に，改良した方法をプログラム化し，最近発表された他の方法と比較し，非常に早く回路が求まることを示す。

2．P－N 項法の改良

2.1　3変数関数での検討

P－N項設計法については既に文献1）で詳述した。ここでは簡単な例により手法のあらましを述べるにとどめる。図1の真理値の関数が与えられたとする。図中網かけのセルが true で，セルの数字は入力変数の組み合わせ $X_1X_2X_3$ を2進数字とみて1の数の少ない順に，また同一のときは数として小さい順に並べたときの番号である。許容項とは否定変数を含まぬ論理積項で，X_1，X_2，X_3，X_1X_2，X_1X_3，X_2X_3，$X_1X_2X_3$ と1（全項）の8（n 変数関数の場合で 2^n）個ある。これらはすべてセル8（座標（1，1，1））を含む。図1（a）の網かけセルで最小の数2を通るP（許容）項 X_3 を選ぶ。これで2，5，8がカバーされる（図1（b））。残った網かけセル4で，4を通るP項 X_1 をとる（図1（c））。これらを $P_1(X_3)$，$P_2(X_1)$ と表す。これで網かけセルをすべてとりつくしたが，P項 $P_1(X_3)$ がセル6を，$P_2(X_1)$ がセル6，7とそれぞれ不要なセルを含むので，これらをN項で打ち抜く。$P_1(X_3)$ は6を通るN項 X_1X_3（$N_{11}(X_1X_3)$ と表現），$P_2(X_1)$ は6を X_1X_3 で，7を X_1X_2 の各N項で打ち抜く（図1（b），（c）の点線の囲み）。ところがこの結果セル8まで除去されてしまったので，これを実現すべく，再びP項をとるが，5を復活させて，X_2X_3 を選ぶ（図1（d），図

中の r_5 は復活セル)。これですべての網かけセルが実現できたので第一段階であるP－N項の選択は終る。

次の段階は，各P項が必要とするN項間の互換性を利用して，最小個数のN項を選ぶ。これは最小被覆問題を解くことになる。本例では，上で求めたN項のうち $N_{22}(X_1X_2)$ だけが $N_{22}(X_2)$ に変わる。得られたP－N項の組み合わせは $P_1(X_3)N_{11}(X_1X_3)$，$P_2(X_1)N_{21}(X_1X_3)N_{22}(X_2)$，$P_3(X_2X_3)$ となり，P項を二段ゲート，N項を三段（入力）ゲートで構成して図4(a)の回路が実現できる。

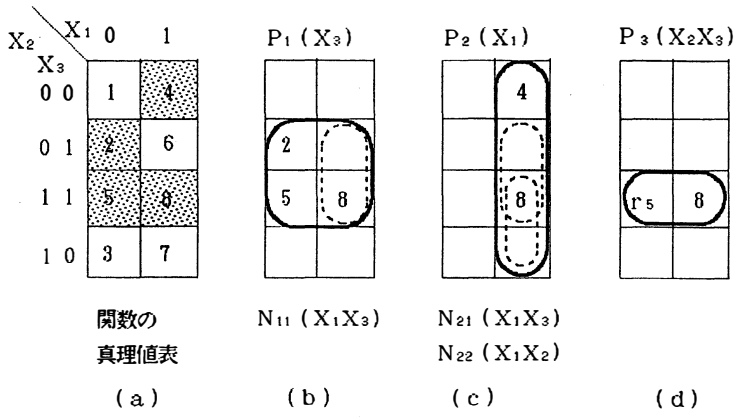

図1　P－N項法の例

P－N項法は多変数関数の設計にも適用できるが，3変数関数については既にHellerman[2]によって（ゲート数，総入力線数の最小化という意味で）最適な回路が表となって発表されている。そこで本方法で得られた回路とHellermanの表とを比較すれば，有効性が明らかとなるので，以下3変数関数について検討する。なお本節で最適回路といえば，Hellermanの表の回路を指すものとする。

3変数関数は $2^8=256$ 個ある。このうち入力変数の置き換えによって一致する関数を区別しないとすれば80個の関数で代表できる。このうちさらに恒等的に0および1となる2つの関数も特殊なので除外する。78個の関数をP－N項法で設計した回路と最適回路とを比較検討した結果は一致した回路

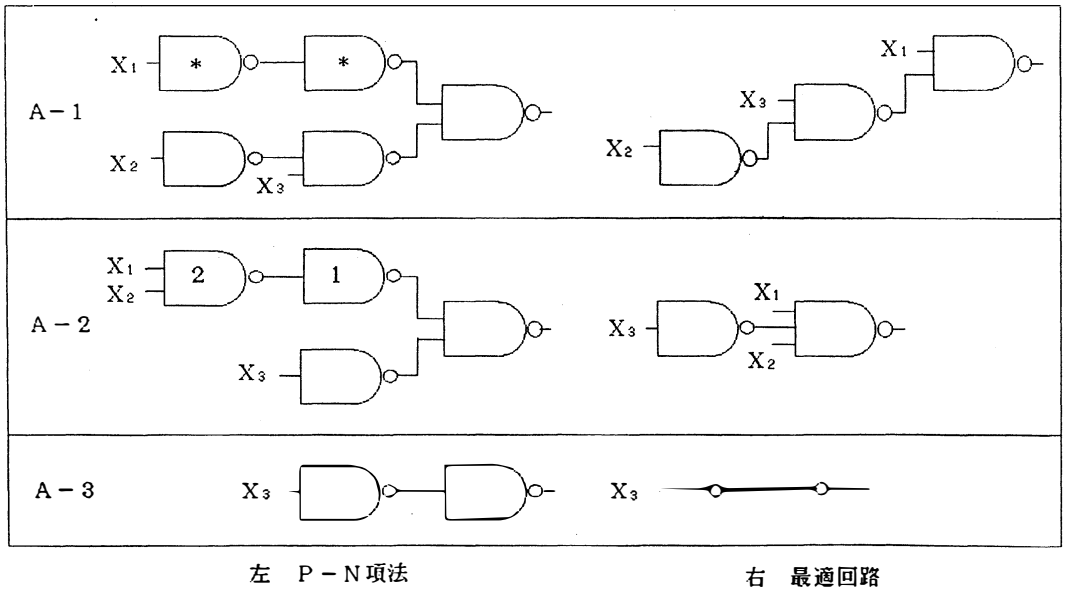

図2　Aの型の回路

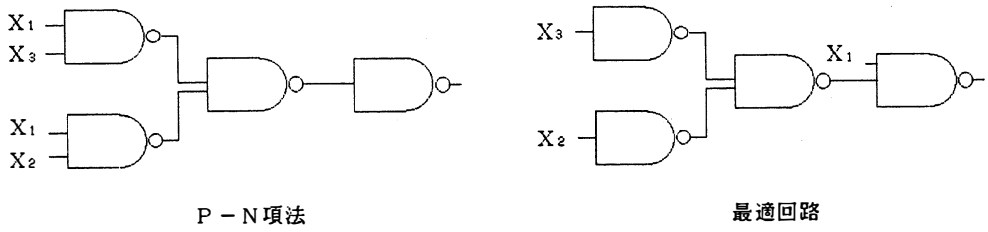

図3　Bの型の回路

が51個，相違した回路27個（内訳(Ⅰ)ゲート数が2多くなった回路11，(II)ゲート数が1多くなった回路6，(Ⅲ)入力線数が1～2本多くなった回路10）であった。相違した回路を誤りの型（最適でないという意味）で分類すると次のA，B，Cの3つにわかれる。

Aの型は図2のようにさらに3つに細分できる。A－1はセル1のP項があって，それを打ち抜くN項が1変数の場合で＊印のゲートのように冗長な部分回路が現れる。これを簡単化すると最適回路となる。この種の回路は6個ある。A－2の回路はセル1のP項があって，それを打ち抜くN項が他のP項に利用されない場合で，図中番号2のゲート入力変数を直接出力ゲートへ加えると，最適回路となる。これに類似の回路は3個ある。A－3の回路はP項 X_3 がN項をもたないので，このためのゲートと出力ゲートが組み合わされて除去できる。この種の回路は1個ある。

Aの型の誤りは三段NAND回路の設計に特有なもので，この設計法が出力ゲートと少なくとも1個のP項が存在するという前提がなされていることにより生ずる。この種の誤りは分類説明文中の処理を行うようプログラムの一部を若干修正して正すことができる。

次がBの型で図3で示した回路である。P項がセル1の許容項で，これを打ち抜くいくつかのN項に共通な変数 X_1 がある。この場合，共通入力変数 X_1 を直接出力ゲートへ入れて入力線数の節約ができる。このような回路は2個ある。

あと一つがCの型で，図4の例のように最適回路では四段ゲート回路となる。この種の回路は15個ある。

P－N項法は論理関数を高々三段のゲート回路で実現しようとするもので，これまでの方法ではCの型の最適回路は得られない。しかし，以下に述べる関数変換の技法を用いると，四段回路も実現で

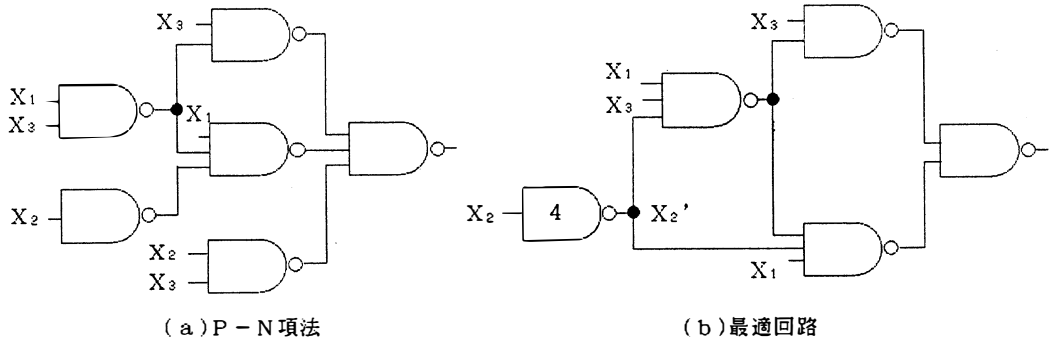

図4　Cの型の回路

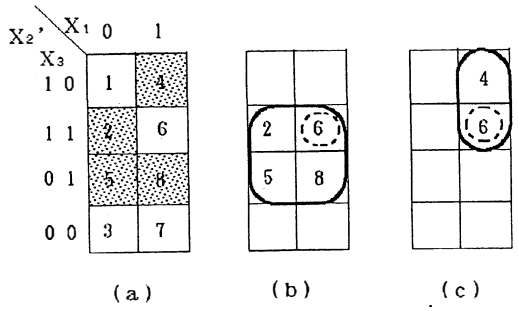

図5　許容項の変換

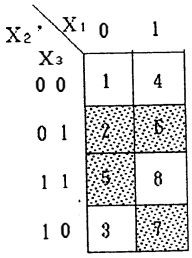

図6　関数の変換

き，Cの型の誤りをかなりの数まで正せる。またこの方法でBの型の誤りも除去できる。

ここで図1(a)の関数について考える。P－N項法ではセル8（座標（1，1，1））を中心に考え，必ずこのセルを含む許容項で，PおよびN項を選んだ。しかし，いま何らかの方法でP項として，項{2，5，6，8}と{4，6}をとり，これを{6}なるN項で打ち抜ければ（図5(b)，(c)参照），前述の方法に比べ少ない個数の項で網かけのセルがすべて実現できることに注目する。各項1個が回路のゲート1個に対応するので，このほうが最適化に有利である。これにはセル6を中心に考えセル6を含む許容項を考えればよい。図5(a)は図1(a)のカルノー図を変数変換$X_2'=\bar{X}_2$（X_2の否定）としたものでセル6が座標（1，1，1）に変わっている。つまりセル8の代わりにセル6を含む項を得るには，セル6を表す2進数（X_1，X_2，X_3）＝（1，0，1）に応じて，$X_1\rightarrow X_1'$，$\bar{X}_2\rightarrow X_2'$，$X_3\rightarrow X_3'$とすればよいことがわかる。これをセル6による許容項の変換と呼ぼう。このように約束すると，セル8の変換は$X_1\rightarrow X_1'$，$X_2\rightarrow X_2'$，$X_3\rightarrow X_3'$，すなわち恒等変換に相当する。さてこの変換した許容項を用いる以外ではさきのP－N項法と全く同じように設計手順を進めればよい。図5では2個のP項$P_1(X_3)$，$P_2(X_1X_2')$と1個のN項$N_{11}(X_1X_2'X_3)=N_{21}(X_1X_2'X_3)$が選択される。これらを二段および三段ゲートで実現することも同じである。ただし，$X_2'=\bar{X}_2$を得るための否定ゲートが四段目に現れる点が異なってくる。いまの場合，回路図は図4(b)で，図中番号4のゲートが否定ゲートを表す。この回路が与えられた関数（図1(a)，図5(a)）の最適回路となっている。このようにセル8以外のセルを中心とする許容項でP－N項を選出することにより，よりよい回路が得られることがある。ところで，いまの操作は関数を固定して許容項を変換した。このことは許容項を固定して，関数をセルで変換しても同一の効果を得る。図6は図1の関数をセル6で変換，$X_1\rightarrow X_1'$，$\bar{X}_2\rightarrow X_2'$，$X_3\rightarrow X_3'$した真理値表である。この図でセル8を含む許容項でP－N項選択すれば，この場合も図4(b)の回路が実現できる。プログラムではあとの処理法を採用している。関数が与えられると，セル8からセル1まで順次関数を変換し，その都度P－N項選択をして，最良の回路を見い出すようにする。このとき四段否定ゲートが二段，三段ゲートと誤りの型Aで見たような接続関係を生ずることがあり，不要なゲートを取り除く操作も加味する必要がある。

以上述べたように，誤りの型Aを除去する対策と，関数変換の技法をP－N項法のプログラムに追加した結果，78回路中73個までHellermanの最適回路と一致した。なお計算時間は1分21秒で使用計算機はFACOM 230-45Sである。

2.2　項表現によるP－N項法の改良

前節（あるいは文献1））のP－N項法の計算機プログラムでは，関数は真理値表そのものをブールベクトルで表す方法をとっているので，2^nの大きさの配列が必要となった。また，許容項も関数と同

じ形式で表現するので，2^n のサイズの配列が必要となった。

ところが，関数は項 C_i（$i=1,2,\cdots,t$）の論理和

$$F=C_1+C_2+\cdots+C_t \qquad (1)$$

の形でも与えられるので，ここでは項表現を使う。但し，項とは，例えば $C_p=X_1X_2$ や，$C_q=\bar{X}_1X_3\bar{X}_4$ のように入力変数 X_j，あるいは $\bar{X}_j$（これらをリテラルという），（$j=1,2,\cdots,n$）の積項で表される。但し，恒等的に1をとる変数は省略してよい。

これらの項はマップで表すと図7のように，複数個のセルを含んでいる。これらの項がもし，関数 F に含まれておれば，図7の項の中のセルは関数値1となる。もし，F の否定 $\bar{F}$ に含まれれば，同じセルは関数値0とみなす。

4（一般に n）変数の場合，4つのリテラルが全部現れる項，例えば $m_0=\bar{X}_1\bar{X}_2\bar{X}_3\bar{X}_4$，$m_{12}=X_1X_2\bar{X}_3\bar{X}_4$，$m_{15}=X_1X_2X_3X_4$ などはマップの1つのセルを表し，最小項という。肯定のリテラルを1で，否定のリテラルを0で置き換えると，$m_0=(0000)$，$m_{12}=(1100)$，$m_{15}=(1111)$ と4桁の2進数字となり，それぞれ10進数に換算すると m_0 は0，m_{12} は12，m_{15} は15となる。これらの例のように16（一般に 2^n）個ある最小項（あるいはセル）は0から15までの番号をふることができる（図8）。

また，最小項 m_0 に許容項1（マップ全体），m_{12} に許容項 X_1X_2，m_{15} に許容項 $X_1X_2X_3X_4$ すなわち，一般に最小項 m_i に m_i の肯定のリテラルのみを残した項を許容項として対応させて，最小項（あるいはセル）m_i の許容項と呼ぶ。

セルに対するこの番号付けはこれまでのP－N項法のセルの番号付けと異なってくる。しかし，各セルの許容項の定義には変わりはない。ただ，前のP－N項法ではセル番号の小さな許容項程，（含むセルの数の大小で比較して）大きい許容項であるという性質になっていたが，新しい番号付けでは，必ずしもこの性質は成り立たない。しかし，各最小項 m_i の否定のリテラルの数 d_i を算出しておいて，この d_i の大きな（同じなら，2進数として小さい）最小項（あるいはセル）の許容項から生成すると，より大きな許容項から生成できる。

更に，項 C_i に対し，C_i に陽に現れていない変数の否定のリテラルを C_i に付け加えた項 C_{imin} を項 C_i の最小番号の最小項と呼ぶ。C_i が $\bar{X}_1X_2X_3$ なら C_{imin} は $\bar{X}_1X_2X_3\bar{X}_4$ であり，C_i が $X_1\bar{X}_3$ なら C_{imin} は $X_1\bar{X}_2\bar{X}_3\bar{X}_4$ となり，図9に＊印で示したセルがそれぞれの項に含まれるセルのうち，最小番号の最小項（セル）を表す。

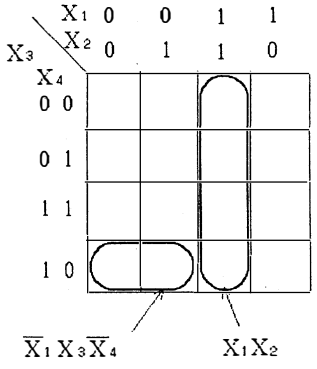

図7　項　　例

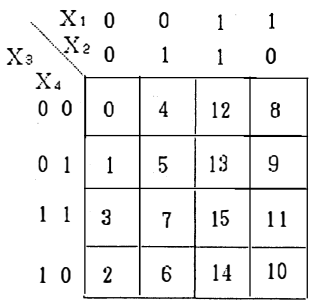

図8　改良P－N項法の
セルの番号付け

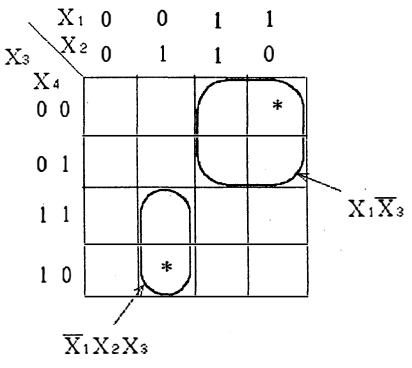

図9　最小番号の最小項

さて，項表現による改良Ｐ－Ｎ項法の基本的な手法は次のとおりである。

関数Fが式（１）のように項表現で与えられているとする。Fの否定$\bar{F}$を求める。初め，$r=1$，$e=1$とする。また，F^eはeが偶数ならF，eが奇数なら$\bar{F}$をそれぞれ表すものとする。

『　　TANT（f，r，e）

〔許容項の生成法〕　によりm個の許容項P_r，P_{r+1}，P_{r+2}，…，P_{r+m-1}が生成したとする。

各許容項P_k（$k=r$，$r+1$，$r+2$，…，$r+m-1$）について，以下の操作を行う。

$f_k=P_k\cap F^e$

f_kがϕ（空）ならRETURN。

f_kがϕでなければ，$r=r+m$，$e=e+1$ とおいて，

TANT（f_k，r，e）を呼ぶ。

但し，

〔許容項の生成法〕　$j=r$ とおく。

1）　$f=C_1+C_2+\cdots+C_t$とし，各項C_i（$i=1,2,\cdots,t$）の最小の値の最小項C_{imin}を求める。また，C_iに含まれる否定のリテラル数d_iを算出し，その数が最大のC_{imin}（複数個あれば，番号の小さい方の最小項）を選んでC_jとする。

2）　セルC_jの許容項を生成する。これをP_jとする。f㊋P_jの結果の関数をあらためてfとする。fがϕなら許容項の生成はこれで終る。そうでなければ，$j=j+1$ として，ステップ１）へいく。　』

但し，手順中許容項の添字は許容項が生成されるごとに順次増えるように書いたが，実際は次のようにする。TANTの手順中，eが奇数の時生成したm個の許容項はＰ－項であり，呼び出しごとに生じた数だけ添字を増やしていく。eが偶数の時生成した許容項はＮ－項で，これはその都度生じた許容項の数だけN_{k1}，N_{k2}，…，N_{km}とする。但し，添字kは一つ先の呼び出しで作られた関数$f_k=P_k\cap F^e$の添字に合わせる。こうして得られた一連の項を基本P_0-N_0項と呼ぼう。

ここで，上記手順TANTの呼び出し手順中eが３以上の奇数の時，次のようにＰ－Ｎ項の数を減らしたり，より大きな項を含むようにするため，次の許容項の拡大操作を行う。

〔許容項の拡大〕　TANTの手順によって，m個のＰ－項P_r，P_{r+1}，P_{r+2}，…，P_{r+m-1}が生成されたとする。Fに含まれる項を適当に加えて，より大きな許容項P'を作り，P'に含まれるP_r，P_{r+1}，P_{r+2}，…，P_{r+m-1}を取り除く。但し，P'はそれまでに生成されていない許容項とする。このような許容項の組に対し，さらにＮ－項を選ぶTANTの手順を続けると，別のＰ－Ｎ項が選ばれる。

基本P_0-N_0項と比較して，コストの小さい方を採用する。この拡大操作は，さらに奇数のeで生じたＰ－項があるごとに行う。

上記手順中現れたディスジョイント・シャープ演算㊋とは，関数fから，ある項C（Cはfに含まれているかどうかわからない）を取り除くとき使われる[3)]。f㊋Cによって，fに含まれる最小項でCにも含まれる最小項は全部取り除かれ，残りのfの最小項のみを含む項の和として表される。

また，Fの否定$\bar{F}$を求めるプログラムも必要である。前のＰ－Ｎ項法では，例えば関数がブールベクトル（110100110111）で表していたので，その否定は０→１，１→０として（001011001000）とすぐに求まった。項表現の関数では，特に変数の数nが大きい場合の否定を求めるプログラムは難しく，我々はそれを所有している。

項は計算機上次のように表している。例えば，$X_1X_3X_4$なら01-11-01-01であり，$\bar{X}_2X_3\bar{X}_4$なら11-10-01-10である。すなわち，一つのリテラルX_jを表すのに２ビットを使い，X_jなら01，$\bar{X}_j$なら10，陽に現れなければ11で示す。また，左より変数X_1，X_2，X_3，X_4と各２ビットずつ割り当て

ている。このため，1ワードで16変数の項が表されて，2ワードつないで32変数，7ワードで100変数の項が表される。項表現では最小項ではなく，もっと大きな項で与えられるので，関数の項数は 2^n より，ずっと小さい。故に改良P－N項法が多変数まで適用可能となる。

更に，前のP－N項法では，すべての許容項をあらかじめ用意して持っているため $2^n \times 2^n$ の記憶量が必要となったが，許容項は必要な都度生成することにする。また，P－N項最小被覆表も行数として 2^n 個あらかじめ用意していたのを，P－項と，それのN－項をみれば互換性のある許容項が即時生成できるので，必要な都度，許容項を発生させるようにして，記憶量の節減が可能である。

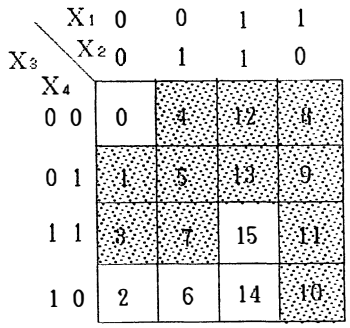

図10　関数 NO. 7 の真理値

2.3　計算結果と他の方法との比較

項表現によるP－N項法は，多変数に適用するにはまだ，いくつかプログラム化の段階で完成していない点があり，ここでは4変数の関数で実行した結果について，後藤が提案した方法[4]（以下行列法と記す）と比較する[5]。

行列法では，被禁止用許容ループ行列と禁止用許容ループ行列を用いて三種類の主許容項（否定主許容項，肯定主許容項，禁止付き主許容項）を求め，それらの最小被覆の中から最小回路を得ている。我々はこの度，この手法を FORTRAN でプログラム化した。

表1は，n（入力変数の数）＝4の関数20個を生成し，ゲート数（G）と入力線数（IN）を比較したものである。この表では関数は，4変数の最小項を0，1，2，…，15のように昇順に並べ，真理値（0または1）を対応する最小項に記入したビット列を16進数に変換した数字で表示している。例えば，表中 NO. 7 の関数は，図10の真理値表で表される関数で（網かけの最小項で関数値1，それ以外は0），最小項のセル番号順に真理値を並べると，0101-1101-1111-1100であり，それぞれ4ビットずつ区切って16進数で読むと 5 DFC と表される。その他の関数も同様の方法で，もとの真理値を表したものである。表1では最小回路と一致する場合には何も記入せず，一致しない場合のみ（G，IN）を記入してある。なお，NO. 7 の関数をそれぞれの方法で実行して回路図にかくと，行列法では図11，本方法では図12となる。

以下に，二つの方法の比較を要約しよう。

(1)　行列法は，最小項をカバーする最小被覆のすべてを求める厳密かつ複雑な方法なのでほとんど

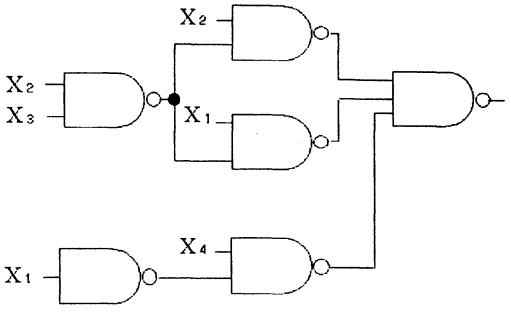

図11　関数 NO. 7 の行列法による回路構成

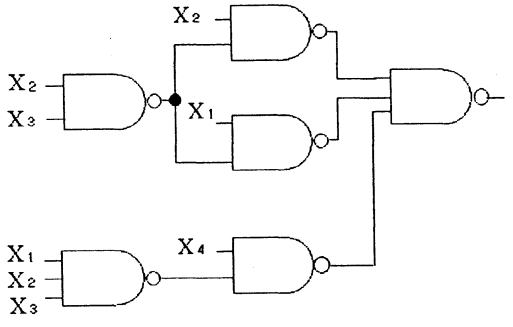

図12　関数 NO. 7 の本方法による回路構成

最小回路と一致する。これに対して，本方法は多変数の関数にまで適用しようとしているので，方法が簡略でありながら，行列法とはほとんど差のない良い回路が求められる。

(2) 行列法は，被禁止用許容ループ，禁止用許容ループ行列（サイズ$2^n \times 2^n$）の記憶のため，および主許容項の最小被覆や最小回路設計の可能な組み合わせのすべてを調べつくすため，多くの記憶量と計算時間を必要とするが，本方法では最小被覆表以外，それほど記憶量を必要としない。

(3) 表1の関数20個の平均計算時間は行列法で192ミリ秒だったが，本方法では52ミリ秒と約1/4の時間で結果が求まった。但し，使用計算機はIBM 3081-KX 4である。

(4) 行列法は，せいぜい$n=6$（それもそのごく一部）ぐらいまでしか対応できないが，本方法はそれ以上の変数の関数に拡張可能である。

表1　4変数関数20個の比較結果

NO	関数	最小回路 G, IN	行列法 G, IN	本方法 G, IN
1	F 7 7 7	6, 9		
2	8 0 F F	6, 9		
3	C 4 4 4	6, 10		
4	8 8 D F	6, 11		
5	8 9 F F	6, 11		
6	E B F F	6, 11		6, 12
7	5 D F C	6, 12		6, 14
8	8 9 B B	6, 12		
9	7 D F F	6, 12		
10	A B D D	6, 13		
11	3 F 0 7	6, 13		
12	3 F D F	6, 13		
13	1 8 F F	6, 13		
14	2 8 F F	6, 13		
15	4 8 F F	6, 13		
16	0 D 5 1	6, 15		
17	2 6 E F	6, 15		
18	1 1 B E	6, 15		
19	8 3 7 F	8, 17	10, 21	10, 21
20	8 7 7 F	8, 20	11, 24	11, 24

但し，G;ゲート数，IN;入力線数，また，NO.19とNO.20の最小回路の段数はそれぞれ4と5である．

3．まとめ

我々は，すでに今から15年も前に三段NANDゲート回路の論理設計法の一つとして，P－N項法を口頭発表し，最近論文としてまとめた。P－N項法は簡単な手法で近似的に良い回路を得ることを目標としている。そこで本論文では，3変数の関数について，どのような場合に最適な回路が得られなかったかを調べ，最適にするには関数変換の手法などを取り入れればよいことをまず示した。

また，P－N項法を項表現で表して，多変数の関数にも適用できるようアルゴリズムに改良を加えた。最近発表された他の三段NANDゲート回路を実現する一手法（行列法）とも比較を行い，非常に高速に求まることを示した。

プログラムはまだ多変数関数にまで適用できるようには至っていない。許容項の拡大でのヒューリスティックの導入など残された問題がいくつかある。

参 考 文 献

1) 松田，宮腰，畠山：三段NANDゲート回路の論理設計法について(Ⅰ) —P－N項法—，富山大学工学部紀要，**44**，(1993).

2) L. Hellerman: A Catalog of Three-Variable Or-Invert and And-Invert Logical Circuits, IEEE Trans. Electron. Comput., **Vol. EC-12**, No. 3, pp. 198-223 (1963).

3) 松田，宮腰：論理式を分離加法形で表現する一手法，情報処理学会論文誌，**33**，(1992).

4) 後藤：一線入力3段NANDゲート回路の行列法による最小化手法，情報処理学会論文誌，**32**，(1991).

5) 大澤，宮腰，松田，畠山：一線入力三段NANDゲート回路の一設計法(2)，平成4年度電気関係学会北陸支部連合大会講演論文集，**B-173**，(1992).

平成4年度電気関係学会北陸支部連合大会で一部発表

The Logical Design of Minimal Three-level NAND Circuits (II)
—— The Improvement of P-N Cube Method ——

Takashi MIYAGOSHI, Hideo MATSUDA, Kazuto OOSAWA,
Toyomasa HATAKEYAMA

We presented the P-N cube method for the synthesis of minimal NAND circuits in the previous paper of this publication. The method does not provide necessarily a optimum circuit as it realizes any function with three-level NAND circuit. A catalog of minimal three-variable NAND circuits has been given by Hellerman. Our synthesis circuits are compared with that catalog and the way of improving our circuits which are in disagreement with the minimal circuit is shown. It is described that if we present the function with a term representation for computer program, whereas we present it with the truth table in the previous method, the improved method becomes to be able to apply to NAND circuits of a large number of variables. It is also shown that our improved method produces the minimal circuit in shorter computing time than the other method reported recently.

〔英文和訳〕

三段 NAND ゲート回路の論理設計法について（II）
—— P－N 項法の改良 ——

宮腰　　隆，松田　秀雄，大澤　一人，畠山　豊正

我々は本紀要の前論文で最小 NAND ゲート回路の設計法として，P－N項法を提案した。本方法はどのような関数も三段 NAND ゲート回路として実現するので，必ずしも最適回路とはいえない。3変数の NAND ゲート回路の最小回路はヘラーマンによって与えられている。我々の合成した回路がカタログと照らし合わされ，最小回路と一致しなかった回路を改善する方法が示されている。もし，関数を前の手法のように真理値ではなく，項表現で表すならば多変数の NAND ゲート回路に適用できるようになるということが述べられる。改良した本方法が最近発表された他の方法より短い計算時間で最小回路が求められることも示される。