

一線入力 3 段 NAND ゲート回路の生成の一手法

後 藤 公 雄

A method for generating NAND gate circuits
with single rail inputs

Kimio GOTO

Abstract

Concerning the design method of three-level cascaded NAND gate logic circuits with single rail inputs, studies such as were made by Gimpel, Muroga et al. are well known. The methods elaborated by these workers, however, are generally difficult and based on complex algorithms, and are partly heuristic. From such standpoint as has been stated above, a computer-oriented method is proposed in which the cells in karnaugh's map are dealt with. In this method, the permissible loop matrices, especially the permissible ones for being inhibited, are used in order to discern the positions of 1-cells and 0-cells from each other between all permissible loop row lines. In addition, the methods for a minimal covering are applied in many portions throughout the process for driving the results.

1. 緒 言

一線入力論理 3 段 NAND ゲート回路の設計法については、Gimpel¹⁾ や室賀²⁾ らによって研究が行われている。しかし一般にアルゴリズムが複雑であったり、発見的・試行錯誤的であったりする。ここではカルノーマップ上のセルを処理する計算機向きの手法を提案する³⁾。

2. 定義と定理

一線入力論理 3 段 NAND ゲート回路の設計に関連する定義、定理とアルゴリズムをつぎに述べる。

2.1 定義と定理

〔定義 1〕 構成リテラルがすべて肯定形の最小項を最大最小項と呼ぶ。

〔定義 2〕 最大最小項セルを含む隣接する 2^i 個のセルグループを許容ループと呼ぶ。

〔定義 3〕 1 セルのみのセルグループを生成するために、許容ループ a を別の許容ループ b で禁止すると

き、 a を被禁止用許容ループ、 b を禁止用許容ループと呼ぶ。

〔定義 4〕 被禁止用許容ループをいくつかの禁止用許容ループで禁止して得られた 1 セルのグループが、同様にして得られた他のどの 1 セルグループにも含まれなければ、この 1 セルグループを主許容項と呼ぶ。

〔定義 5〕 ある 1 セルがある主許容項のみにしか含まれないとき、この主許容項を必須許容項と呼ぶ。

〔定理 1〕 同じ許容ループが 2 回以上被禁止用許容ループとして選ばれることはない。

〔定理 2〕 1 セルのみを含む許容ループは禁止用許容ループにはなり得ない (全セルの許容ループも同様)。

〔定理 3〕 1 セルのみを含む許容ループの中で最大のものは主許容項であり、被禁止用許容ループとはならない。

〔定義 6〕 全セルより成る許容ループを u 許容ループと呼ぶ。

〔定理 4〕 0 セル全部が u 許容ループの 0 セル全部と一致するリテラル 1 個の許容ループ x_i が存在すれば、 $\overline{x_i}$ は必須許容項である。

〔定義 7〕 許容ループを行線とし、0 セル列線と 1 セル列線上の該当箇所それぞれ×印と○印を記入した

ものを許容ループ行列と呼ぶ。すなわち許容ループに相当する論理積に含まれる 0 セル番号の列線には×印を、1 セル番号の列線には○印を記入する。また禁止用許容ループおよび被禁止用許容ループのための行列を、それぞれ禁止用許容ループ行列および被禁止用許容ループ行列と呼ぶ。

3. アルゴリズム

前章で述べた定義と定理にしたがって、一線入力論理 3 段 NAND ゲート回路の設計のアルゴリズムをつぎに述べる。

〔ステップ 1〕 与えられた関数より許容ループ行列を生成する。

〔ステップ 2〕 許容ループ行列の中で 1 セルしか持たない行線を探し、これを主許容項として記憶し、定理 2 と定理 3 により禁止用許容ループ行列と被禁止用許容ループ行列からこれを除去する。同時にこの主許容項の含む 1 セル列線も除去される。

〔ステップ 3〕 0 セル全部が u 許容ループのそれと一致するリテラル 1 個の許容ループで u 許容ループを禁止すると、定理 4 よりこれは必須許容項となる。このとき定理 1 より u 許容ループを被禁止用許容ループから除去する。

〔ステップ 4〕 許容ループ行列の行線上で 0 セル列線を調べ、×印が 1 個しか列線上に存在しない場合、その×印の存在する行線を被禁止用許容ループから除去する。

〔ステップ 5〕 残された被禁止用許容ループ行列から、ステップ 3 で作られた必須許容項に含まれる 1 セルに相当する列線を除去し縮約行列を作成する。

〔ステップ 6〕 得られた被禁止用許容ループの縮約行列の行線により、1 セルをカバーする最小被覆を作る。

〔ステップ 7〕 ステップ 6 で求めた最小被覆の各要素（行線）ごとに、それらの行線のもつ 0 セル全部をカバーする行線の最小被覆を求める。用いる行列は禁止用許容ループ行列とし、この行列から自分自身の行線は除くものとする。

〔ステップ 8〕 ステップ 6 と 7 によって得られる被禁止用および禁止用許容ループの組によって主許容項を求め、その 1 セル要素を記憶する。

〔ステップ 9〕 これまでに得られた必須許容項と許容項による最小被覆を求める。

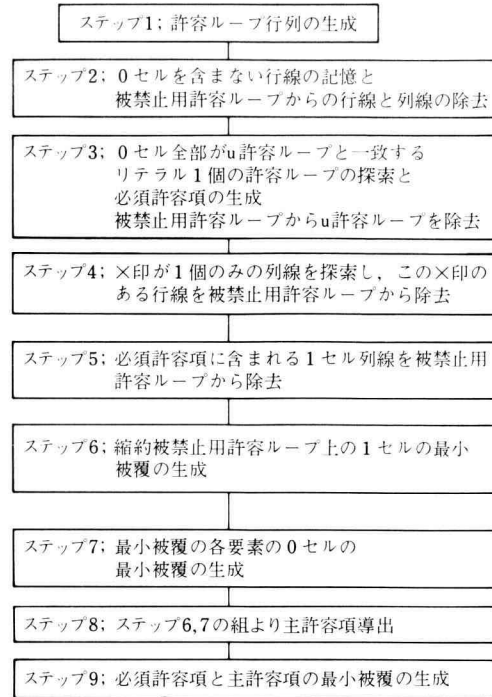


図 1. フローチャート

Fig. 1. Flowchart

4. 計算例

前章で述べたアルゴリズムにしたがった手計算の一例をつぎに示す。そのため

$$f = \sum (0, 1, 4, 5, 6, 8, 9, 10, 12, 13, 15) \quad (1)$$

で与えられる関数の一線入力論理 3 段 NAND ゲート回路を求めることとする。

ステップ 1 により、この関数 f の許容ループ行列は図 2 のようになる。

ステップ 2 で行線 m により主許容項 $x_1x_2x_4$ が求まる。この主許容項は被禁止用許容ループ行列から除去される。この様子を図 3 に示す。

ステップ 3 として、行線 d (許容ループ x_3) の 0 セルは行線 a (u 許容ループ) の 0 セル全部と一致するので、被禁止用許容ループ a を禁止用許容ループ d で禁止して必須許容項 $a-d$ が求まる。この必須許容項は \bar{x}_3 となる。このとき行線 a (u 許容ループ) は定理 1 より被禁止用許容ループから除去される。この模様を

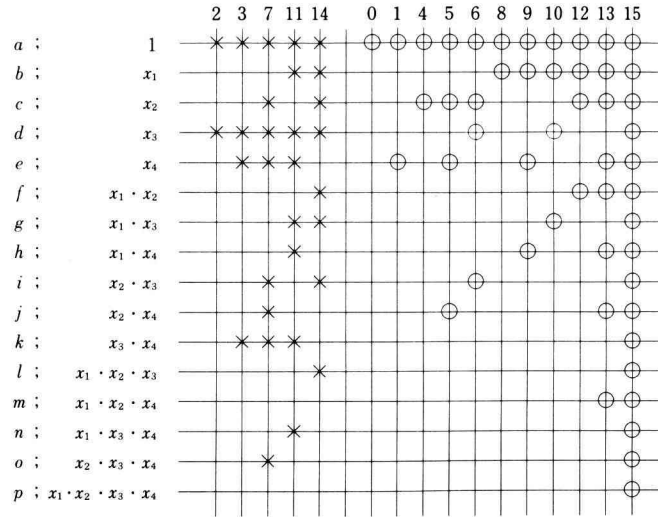


図 2. 許容ループ行列
Fig. 2. Permissible Loop matrix

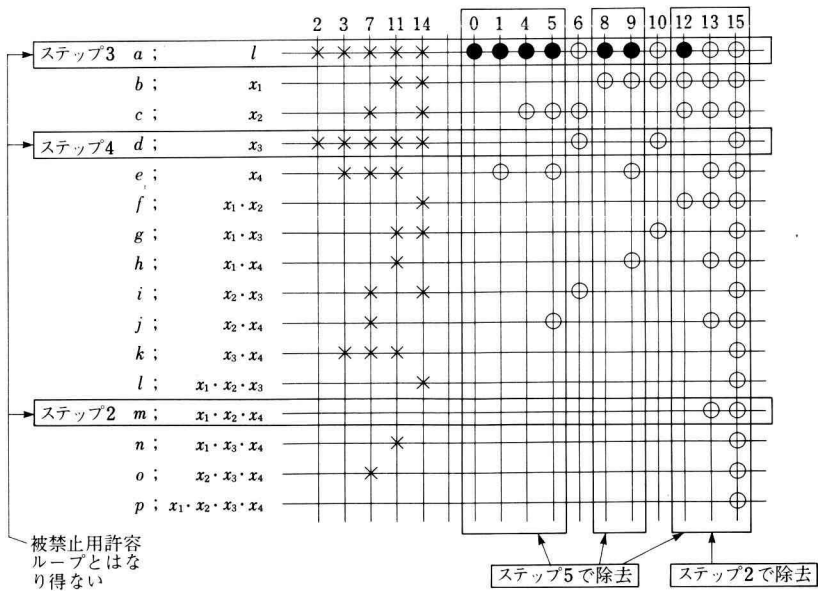


図 3. 被禁止用許容ループ行列の縮約
Fig. 3. Reduction of permissible loop matrix for being inhibited

図 3 に示す。

ステップ 4 で行線 d は被禁止用許容ループから除去される。この様子を図 3 に示す。

ステップ 5 でこれらの主許容項 $x_1 x_2 x_4$ と必須許容

項 \bar{x}_3 に含まれる 1 セル (図 3 で黒丸で示す) を除去してセル番号 6 と 10 のみが残る。

ステップ 6 でこれらの 1 セルを含む最小被覆をベトリック関数を用いて求めると、

$$\begin{aligned}
 [6] \cdot [10] &= (c+i)(b+g) \\
 &= cb+cg+bi+gi. \tag{2}
 \end{aligned}$$

この式で積項 cb を最小被覆として採用する。ステップ7として積項 cb のリテラル c と b を被禁止用許容ループとし、それぞれを禁止する許容ループを探す。そのため c と b の行線の持つ0セルをそれぞれカバーする行線の最小被覆をステップ7で求めると、 c については

$$\begin{aligned}
 [7] \cdot [14] &= (e+i+j+k+o) \\
 &\quad (b+f+g+i+l) \\
 &= i+eb+\dots, \tag{3}
 \end{aligned}$$

b については

$$\begin{aligned}
 [11] \cdot [14] &= (e+g+h+k+n) \\
 &\quad (c+f+g+i+l) \\
 &= g+ec+\dots \tag{4}
 \end{aligned}$$

が得られる。したがってステップ8として c と b をそれぞれ式(3)と(4)の結果で禁止して、

$$\begin{aligned}
 c-i &= (6)-(6)=\phi \\
 c-eb &= (6)-(10)=(6) \tag{5}
 \end{aligned}$$

$$\begin{aligned}
 b-g &= (10)-(10)=\phi \\
 b-ec &= (10)-(6)=(10) \tag{6}
 \end{aligned}$$

を得る。これより主許容項として $c-eb(x_2\bar{x}_4\bar{x}_1)$ および $b-ec(x_1\bar{x}_4\bar{x}_2)$ が求まる。ステップ9として、これまでに得られた必須許容項1個と主許容項3個による最小被覆を示すと図4のようになる。この図より、最小項 $m_0, m_1, m_4, m_5, m_6, m_8, m_9, m_{10}, m_{12}, m_{13}$ および m_{15} がこれらの必須許容項 m と主許容項 $a-d = a, c-eb = \beta$ および $b-ec = \gamma$ にどのように含まれるかを、ペトリック関数 p_c を用いてつぎのように表すことができる。

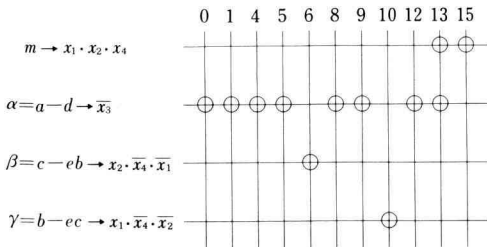


図4. 結果の最小化
Fig. 4. Minimization of the result

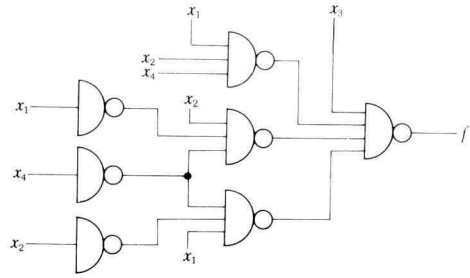


図5. $f = \sum(0, 1, 4, 5, 6, 8, 9, 10, 12, 13, 15)$ の最終回路
Fig. 5. Final circuit of $f = \sum(0, 1, 4, 5, 6, 8, 9, 10, 12, 13, 15)$

$$\begin{aligned}
 p_c &= [0] \cdot [1] \cdot [4] \cdot [5] \cdot [6] \cdot [8] \\
 &\quad \cdot [9] \cdot [10] \cdot [12] \cdot [13] \cdot [15] \\
 &= a \cdot a \cdot a \cdot a \cdot \beta \cdot a \cdot a \cdot \gamma \cdot a(m+a)m \\
 &= a \cdot \beta \cdot \gamma \cdot m \tag{7}
 \end{aligned}$$

これより、 α, β, γ および m のすべての項が採用でき、式(1)は

$$\begin{aligned}
 f &= x_1x_2x_4 + \bar{x}_3 + x_2\bar{x}_4\bar{x}_1 + x_1\bar{x}_4\bar{x}_2 \\
 &= \overline{x_3 \cdot x_1x_2x_4 \cdot x_2\bar{x}_4\bar{x}_1 \cdot x_1\bar{x}_4\bar{x}_2} \tag{8}
 \end{aligned}$$

となる。これより図5の回路が求まる。

この回路の(ゲート数, 結線数, 段数)は(7, 16, 3)となる。

5. 結果の検討

本論文のアルゴリズムはつぎのような特徴を持っている。すなわち、

(1) カルノー図上のセル集合を処理するので、数列の処理が必要である。記号処理言語として作られ、リスト処理が可能なLISP言語は数列をリストで表現できるので、この場合の処理に適している。

(2) 本論文の手法では、許容ループを用いて一方の行線の0セルで他方の行線のそれを抹消するようにし、1セルのみを含む主許容項または必須許容項を作っている。したがって異なる許容ループ間で0セル同士および1セル同士の比較が明瞭、簡単である。

(3) 被禁止用許容ループと禁止用許容ループの選択にあたっては、それらの必要最小限のものを選ぶため最小被覆の考え方を用いる。このため最小数の必須許容項または主許容項が生成できる。

(4) 1セルのみよりなる許容ループの最大のものを利用し、それに含まれる1セルは除去し、残りの1セルのみを含む主許容項を探すようにし、探索の時間を省いている。

(5) 全最小項を含む必須許容項および主許容項の最小被覆を求め、最小解が得られるように配慮してある。一般には前章より複雑な解が得られる。

なお、問題点としてはつぎのような点がある。すなわち、禁止用許容ループおよび被禁止用許容ループを探すとき、最小被覆をベトリック関数の形式で求め、これを展開して求めた積項の最も簡単なものを用いるようにしている。しかし、実際には他の積項からも最小解が得られる可能性があり、ゲート数、結線数の複雑度に依存する複雑指数を配慮し、複数解を求めて相互比較する必要がある。

6. 結 言

現在、LISP 言語によりプログラム作成中であるが、今後は、前述したように回路の複雑指数をも考えたアルゴリズムの検討が必要である。

参 考 文 献

- 1) J.F. Gimpel: The Minimization of TANT Networks, IEEE TRANS., Electron. Comput., Vol. EC-16, pp. 18-38, Feb. 1967.
- 2) S. Muroga: Logic Design and Switching Theory, John Wiley and Sons, 1979.
- 3) 後藤公雄: 一線入力論理3段 NAND ゲート回路の一設計法, 情報処理学会第38回全国大会, 5S-2.