琉球大学学術リポジトリ

三値論理関数の完全系と任意関数の簡単化について

メタデータ	言語:
	出版者: 琉球大学理工学部
	公開日: 2013-06-10
	キーワード (Ja):
	キーワード (En):
	作成者: 鉢嶺, 元助, Hachimine, Gensuke
	メールアドレス:
	所属:
URL	http://hdl.handle.net/20.500.12000/26294

三値論理関数の完全系と任意関数の簡単化について

鉢　嶺　元　助*

A Functionally Complete for Ternary Logic Functions and Simplification Method for Arbitrary Functions.

by Gensuke HACHIMINE*

Summary

In general, ternary logic system deals with more information than the binary logic system.

But, the number of elements being construct the ternary logic system lots of. Therefore, it is important problem to decrease the number of elements.

In this paper, A new functionally complete functions, which are $H_1(\mathbf{x})$, $H_2(\mathbf{x})$, $H_3(\mathbf{x})$, are defined and developed.

The functions set is showen to be capable of representing all one or two variable three valued functions, and method of simplifing these functions from nature of the functionally comple functions is showen.

Designs for ternary half and full adder that could be composed out of the $H_1(\mathbf{x})$, $H_2(\mathbf{x})$, $H_3(\mathbf{x})$, $\mathrm{Min}(\mathbf{x})$, $\mathrm{Max}(\mathbf{x})$ operators are given.

They are not only simplicity in structure but also high stability on the fluctua-tions in power supply.

1. あらまし

一般に，三値論理系は二値論理系に比して取扱う情報量（単一機能素子当り）が大きい半面，回路（単一機能素子回路）を構成する素子数がかなり多くなる。したがって，いかにして素子数を軽減させるかというのが三値論理演算回路を構成するうえで重要な課題となる。

本論文では，三値論理関数を構成するために必要な基本演算子を3つ定義し，それらを各々 $H_1(\mathbf{x})$, $H_2(\mathbf{x})$, H_3 と名付け，トランジスタとダイオードでもって回路を実現するとともに，それら基本演算子の諸性質をあげ，それによる任意関数の簡単化について述べてある。また，3つの基本演算子と Min $(\mathbf{x}_1, x_2, \cdots, xn)$，Max $(x_1, x_2, \cdots, xn)$ を使って三値半加算器，および三値全加算器を構成したところこれまで発表されているものに比して回路構成が容易で，かつ安定性の面でも良好なものであると思われるのでここに発表する。

受付：昭和49年5月9日

*琉球大学理工学部電気工学科

2. 本　論

m値n変数関数は全部で m^{m^n} 個存在する。これらすべての関数を実現するためには如何なる命題演算が必要であるか，またその中で Prime Implicant なものはどのようなものかということと，それらが現存する物理的素子で実現可能であるかという問題は別個のものとして取扱われているのが現状のようである。しかしながら，実際に情報処理機械を構成していくためには，基本演算子の導出とそれの具体的な回路構成については関連して研究をすすめなければならないのではないかと考える。

本論文では，三値1および2変数関数に対する基本

演算子を定義するとともに，それらの具体的な回路構成，さらには任意関数の簡単化等についても論じる。

ただし，以後三進数を表示するときには Mod-3 (Modulus Three) 方式+ を用いることにする。

2－1：基本演算子の定義と回路構成

三値1変数関数は $3^3=27$ 個あり，これらを実現するための基本命題演算には Table-(1) のようなものがある。

x	Ⅰ			Ⅱ				Ⅲ		
0	1	2	2	2	2	0	1	2	1	2
1	2	0	1	0	2	1	1	0	0	1
2	0	1	0	0	0	1	2	0	0	1
0	2	2	2	2	0	0	0	2	2	※
1	2	1	2	0	2	0	2	0	2	※
2	0	0	1	0	0	2	2	2	0	※
x	Ⅲ			Ⅳ						※

Table-(1)　functionary Complete functions.

Ⅰは巡回置換を用いたもので，通常 Post[1] の完全系++ といわれている。これらの具体的な回路構成は今西[2] らによってなされている。Ⅱは三根，長谷川[3] らによって定義され，回路化されている。Ⅲは今西[4] らによって各演算子が1個のトランジスタで構成できるということが示されている。また，Ⅳは一致関数といわれるもので具体的な回路構成についてはまだ示されてない。

筆者は，基本演算子として $H_1(x)$, $H_2(x)$, H_3 を定義した。これらの演算機能は Table-(2) に示すとうりである。ところで，$H_1(x)$, $H_2(x)$, H_3 が

x	$H_1(x)$	$H_2(x)$	$H_3(x)$
0	2	2	1
1	0	0	1
2	0	2	1

Table-(2)　Truth tatble for $H_1(x)$, $H_2(x)$, and H_3。

完全系を構成しているかどうかは，関数集合が完全系を成すための必要十分条件[6]

i)：定義した関数集合から少なくとも1つの非縮退関数** が得られること。

ii)：定義した関数集合から巡回関数 $f_1=x\oplus 1$, $f_2=x\oplus 2$ が得られること。

iii)：定義した関数集合の中に非線形関数が含まれていること。

iV)：定義した関数集合の中に非自己巡回関数** が含まれること。

を満足しているかどうかを調べてみればよい。必要十分条件の i) は本論中の例題で示してある関数（非縮退2変数関数）がこれらの演算子で構成されることから明らかである。ii) は，$f_1=x\oplus 1=\mathrm{Max}\,(\mathrm{Min}\,(H_2(x), H_3), H_1(x)))$, $f_2=x\oplus 2=\mathrm{Max}\,(\mathrm{Min}\,(H_1(x), H_3), H_1\,(H_2(x)))$ より明らかである。iii) は，$H_1(x)$, $H_2(x)$ が非線形関数であること

+三値の真理値集合を $L=\{0, 1, 2\}$ とする表示方式をいう。

++存在しうる真理値関数をすべて構成しうる命題演算を完全系という。

** 文献(6)を参照のこと。

から条件を満足している。一方 iv) は，$H_1(x)$，$H_2(x)$，H_3 がそれぞれ非自己巡回関数であることから明らかである。したがって，ここで定義した演算子は三値論理関数の完全系を成す。

そこで，三値1変数関数27個をこれら基本演算子で構成すると Table-(3) のようになる。

	x			基本系による展開式
	0	1	2	
$f(x)$	0	0	0	$x \cdot H_1(x)$
	1	0	0	$H_2(x_2 \cdot H_3$
	2	0	0	$H_1(x)$
	0	1	0	$x \cdot H_1(H_2(x))$
	1	1	0	$(H_1(x) \vee H_1(H_2(x)) \cdot H_3$
	2	1	0	$H_1(x) \vee x \cdot H_1(H_2(x)$
	0	2	0	$H_1(H_2(x))$
	1	2	0	$H_1(x) \cdot H_3 \vee H_1(H_2(x))$
	2	2	0	$H_1(x) \vee H_1(H_2(x))$
	0	0	1	$x \cdot H_2(x) \cdot H_3$
	1	0	1	$H_2(x) \cdot H_3$
	2	0	1	$H_1(x) \vee H_2(x) \cdot H_3$
	0	1	1	$x \cdot H_3$
	1	1	1	H_3

Table-(3)の(1) Three-Valued one variable functions-1。

	x			基本系による展開式
	0	1	2	
$f(x)$	2	1	1	$H_3 \vee H_1(x)$
	0	2	1	$H_1(H_2(x)) \vee x \cdot H_3$
	1	2	1	$H_1(H_2(x)) \vee H_3$
	2	2	1	$H_1(x) \vee H_3 \vee H_1(H_2(x))$
	0	0	2	$x \cdot H_2(x)$
	1	0	2	$x \cdot H_2(x) \vee H_1(x) \cdot H_3$
	2	0	2	$H_2(x)$
	0	1	2	x
	1	1	2	$x \vee H_3$
	2	1	2	$x \vee H_1(x)$
	0	2	2	$x \vee H_1(H_2(x))$
	1	2	2	$x \vee H_3 \vee H_1(H_2(x))$
	2	2	2	$H_2(H_1(x))$
	※	※	※	※

Table-(3)の(2) Three-valued one variable functions-2。

つぎにこれら基本演算子の回路構成について述べる。いま，真理値の0，1，2を各々電位の低，中，高に対応させると $H_1(x)$，$H_2(x)$，H_3 は fig-(1) に示すように比較的簡単な回路で実現することができる。

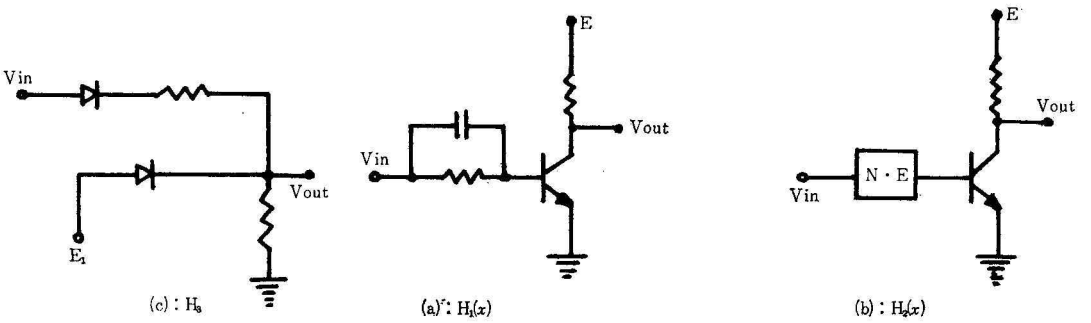

Fig-(1) $H_1(x)$, $H_2(x)$, and H_3 operator Circuits.

$H_1(x)$ の回路は通常のインバータ回路で表現することができる。$H_2(x)$ の回路はインバータ回路の入力側に負性抵抗素子* を接続し，それの電流特性を利用してトランジスタ（以後 Tr で示す）を制御している。すなわち，入力端子に真理値0に相当する低電位が加わると負性抵抗素子にかかる電圧が低いため，それを通って Tr のベースに流れる電流が動作電流以下のためしゃ断となり，出力電位は真理値2に相

*：$H_2(x)$ の構成回路図中ではNE (nonlinear element) にて示してある。

当する高電位となる。入力信号を増して真理値1に相当する中電位にすると負性抵抗素子を通ってベースへ流れる電流が増して Tr は飽和し，出力電位は低電位となる。さらに入力信号を増して真理値2に相当する高電位にすると，負性抵抗素子が負性抵抗領域に入るためベース電流が急激に減り Tr は再びしゃ断状態となる。したがって，この回路は真理値（0，1，2）に対応する入力信号に対して（2，0，2）なる出力信号を作り出す。負性抵抗素子の特性曲線と $H_2(x)$ の出力ー入力特性曲線を Fig－(2)，Fig－(3) に示してある。

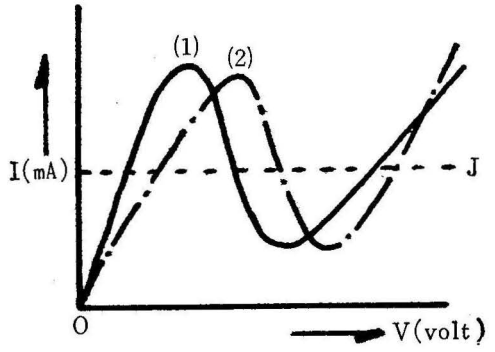

Fig-(2)　NE characteristic curves.

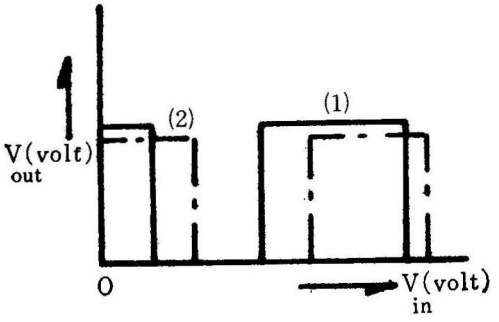

Fig-(3)　Relation between Vin and Vout for $H_2(x)$.

基本演算子 H_3 の回路は普通のＯＲゲート回路で，一つの入力端子には真理値1に相当する中電位が加えてあり他端子より入力信号を印加する。この端子にはダイオードと直列に抵抗が接続してあり，入力信号が真理値2に相当する高電位になっても出力端子には常に真理値1に相当する中電位があらわれるようにしてある。しかるに，この回路は真理値（0，1，2）に相当する入力信号に対して（1，1，1）なる信号を出力として与える。

ところで，$H_2(x)$ の構成のときに用いる負性抵抗素子はトンネルダイオードを用いてもよいが，トンネルダイオードは入力信号振巾や素子値の偏差に対して制限が加わるため雑音や電源電圧の変動により誤動作をおこす危険性がある。本研究では Fig－(4) に示す回路で負性抵抗特性素子を作り出した。

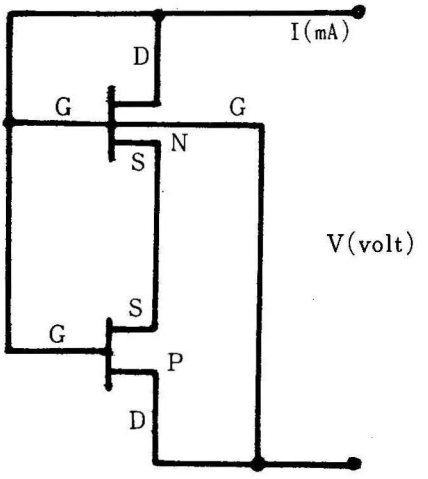

Fig-(4)　NE Circuit.

2-2：基本演算子の性質と任意関数の構成

はじめに，任意関数を実現していくうえで必要となる事項について述べる。

いま，変数 x のとりうる値を 0～P-1 までのP個とし，i をそのうちの1つの値としたとき

$$Li(x)=(x-0)(x-1)\cdots(x-i+1)(x-i-1)\cdots(x-p+1)/(i-1)\cdot(i-1)\cdots(i-i+1)(i-i-1)\cdots(i-p+1)$$

なる関数を一般にラグランジュ (Lagrangue) の関数，または一致関数といい多値論理関数を表現するとき便利な関数である。

この関数の性質としてはつぎのようなものがある。

i)：$\sum_{i=0}^{P-1} Li(x) = 1$

ii)：$Li(x) \wedge Lj(x) = 1 \quad (i=j)$

$\qquad\qquad\qquad\quad = 0 \quad (i \neq j)$

iii)：$\sum_{i=0}^{P-1} i \wedge Li(x) = x$

Mod-3 方式を用いた場合のラグランジュ関数を基本演算子 $H_1(x)$，$H_2(x)$，H_3 で構成するとつぎのようになる。

$$\begin{cases} L_0(x) = \mathrm{Min}(H_1(x),\ H_3) \\ L_1(x) = \mathrm{Min}(H_1(H_2(x)),\ H_3) \\ L_2(x) = \mathrm{Min}(x,\ H_2(x),\ H_3) \end{cases}$$

ところで，二変数関数を表現するときに変数の各組合せに対して関数が2をとる場合があるので，それに必要な表現式を Table-(4) に示してある。

		基本系による展開式
$f(x,y)=2$	$2(L_0(x)\cdot L_0(y))$	$H_1(x)\cdot H_1(y)$
	$2(L_1(x)\cdot L_0(y))$	$H_1(H_2(x))\cdot H_1(y)$
	$2(L_0(x)\cdot L_1(y))$	$H_1(x)\cdot H_1(H_2(y))$
	$2(L_2(x)\cdot L_0(y))$	$x\cdot H_2(x)\cdot H_1(y)$
	$2(L_1(x)\cdot L_1(y))$	$H_1(H_2(x))\cdot H_1(H_2(y))$
	$2(L_0(x)\cdot L_2(y))$	$H_1(x)\cdot y\cdot H_2(y)$
	$2(L_2(x)\cdot L_1(y))$	$x\cdot H_2(x)\cdot H_1(H_2(y))$
	$2(L_1(x)\cdot L_2(y))$	$H_1(H_2(x))\cdot y\cdot H_2(y)$
	$2(L_2(x)\cdot L_2(y))$	$x\cdot H_2(x)\cdot y\cdot H_2(y)$

Table-(4) : Three valued two variable functions.

ここで，任意関数として三値論理和関数をとりあげ，それをラグランジュ関数にて表現してみる。

$x \backslash y$	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

Table-(5) : Truth table for $fx\otimes y$.

三値論理和関数の真理値表 (Table-(5)) より関数値が2をとる x，y の組合せは，

$\{(x,\ y)\mid fx\otimes y=2\}=\{(2,\ 0),\ (0,\ 2),\ (1,\ 1)\}$ で，関数値が1をとる x，y の組合せは，

$\{(x,\ y)\mid fx\otimes y=1\}=\{(2,\ 2),\ (1,\ 0),\ (0,\ 1)\}$ となる。

したがって，この関数はつぎのように表現できる。

$$\begin{aligned} fx\otimes y = {} & L_2(x)\cdot(y)\ \vee\ L_1(x)\cdot L_0(y)\ \vee \\ & L_0(x)\cdot L_1(y)\ \vee\ 2(L_2(x)\cdot L_1(y))\ \vee\ 2(L_1(x)\cdot L_1(y))\ \vee\ 2(L_1(x)\cdot L_2(y)) \end{aligned}$$

ただし，$A\cdot B=\mathrm{Min}(A,\ B)$，$A\vee B=\mathrm{Max}(A,\ B)$ を表わす。このままの状態で回路を構成すると素子数が膨大になる。そこで，次の節においては基本関数の性質を使った任意関数の簡単化について述べる。

2-3：任意関数の簡単化

簡単化にはいるまえに基本演算子の諸性質を列記しておく。

2-3-1：基本演子の諸性質

1). $H_i(H_3(x))=0$
2). $H_1(H_i(x))=1$
3). $H_2(H_i(x))=2$
4). $H_i(H_i(x))\cdot H_i(x)=0$
5). $H_1(x\cdot y)=H_1(x)\vee H_1(y)$
6). $H_1(x\vee y)=H_1(x)\cdot H_1(y)$
7). $(x\cdot y)\cdot H_2(x\cdot y)=(x\cdot y)\cdot(H_2(x)\cdot H_2(y))$
8). $(x\cdot y)\vee H_2(x\cdot y)=(x\cdot y)\vee(H_1(x)\vee H_1(y))$
9). $(x\vee y)\cdot H_2(x\vee y)=(x\vee y)\vee(H_1(x)\cdot H_1(y))$
10). $(x\vee y)\vee H_2(x\vee y)=(x\vee y)\vee(H_2(x)\cdot H_2(y))$
11). $H_1(x\vee y)\cdot H_2(x\cdot y)=H_1(x\vee y)$
12). $H_2(x\vee y)\cdot H_1(x\cdot y)=(H_1(x)\cdot H_2(y)\vee(H_1(y)\cdot H_2(x))$
13). $H_2(x\vee y)\vee H_1(x\cdot y)=H_2(x)\vee H_2(y)$
14). $H_1(H_2(x\vee y))\cdot H_2(x\cdot y)=H_1(H_2(x))\cdot H_1(y)\vee H_1(x)\cdot H_1(H_2(y))$
15). $H_1(H_2(x\cdot y))\cdot H_2(x\vee y)=x\cdot H_2(x)\cdot H_1(H_2(y))\vee y\cdot H_2(y)\cdot H_1(H_2(x))$

2-3-2：簡単化手法

2変数関数の変数 x，y に換って $\mathrm{Min}(x,\ y)$，$\mathrm{Max}(x,\ y)$ をあらたな変数と考えると，与えられた関数が変数値（x，y に対する）の配列こそちがえその値が同じで，しかも関数値も等しいものは1つで代表させることができるので，取扱う変数の組合せが少なくてすむ。まず，例を通して簡単化のアルゴリズ

ムを記述する。なお，ここで取扱う関数は前節で示した三値論理和関数である。

アルゴリズム

i)：与えられた真理値表（Table-(5)）の中で関数値の等しい変数の組をひとまとめにする。

Table-(6) にまとめたものが示されている。

	I			II			III		
x	0	2	1	0	1	2	1	0	2
y	0	1	2	1	0	2	1	2	0
$fx \otimes y$	0	0	0	1	1	1	2	2	2

Table-(6)：Classificational table.

ii)：同一の組の中で変数値の組合せ方が同じであるものに対しては1つで代表させる。

Table-(7) に縮小した真理値表が示してある。

	I′		II′		III′	
x	0	1	0	2	1	0
y	0	2	1	2	1	2
$fx \otimes y$	0	0	1	1	2	2

Table-(7)：Simplified table.

iii)：残った変数の組に対して Min（x，y），Max（x，y）をとり，それらをあらたな変数として H_1 (Min (x, y)), H_2 (Min (x, y)), H_1 (Max (x, y)), H_2 (Max (x, y)), H_3 をつくる。

iV)：$x = \mathrm{Min}(x, y)$，$y = \mathrm{Max}(x, y)$ とおいて与えられた関数を展開する。

Table-(8) より

$$fx \otimes y = L_0(x) \cdot L_1(y) \vee L_2(x) \cdot L_2(y) \vee 2(L_1(x) \cdot L_1(y)) \vee 2(L_0(x) \cdot L_2(y) = H_3 \cdot \{x \cdot y \cdot H_2(x) \cdot H_2(y) \vee H_1(x) \cdot H_1(H_2(y))\} \vee H_1(H_2(x)) \cdot H_1(H_2(y) \vee y \cdot H_2(y) \cdot H_1(x)$$

となる。

$fx \otimes y$	x	y	$Y = \mathrm{Max}(x, y)$	$X = \mathrm{Min}(x, y)$	$H_1(x)$	$H_2(x)$	$H_1(y)$	$H_2(y)$	H_3
0	0	0	0	0	2	2	2	2	1
1	0	1	1	0	0	0	2	2	1
2	1	1	1	1	0	0	0	0	1
2	0	2	2	0	0	2	2	2	1
0	1	2	2	1	0	2	0	0	1
1	2	2	2	2	0	2	0	2	1

Table-(8)：Operational table for H_1 (x), H_2 (x), H_1 (y) H_2 (y), and H_3.

V)：基本演算子の諸性質を使って iV) で与えられた式を簡単化する。

$$f_{x \oplus y} = \{(y \vee H_1(H_2(x))) \cdot (H_1(x) \vee H_1(H_2(y)))\} \vee (H_3 \cdot x \cdot H_2(x))$$

VI)：$x \cdot y = x = x \cdot y$，$x \vee y = y = x \vee y$ により，V) で得られた式をもとの変数 x，y の表現式にもどしてやる。

$$f_{x \oplus y} = \{((x \vee y) \vee H_1(H_2(x \cdot y))) \cdot (H_1(x \cdot y) \vee H_1(H_2(x \vee y)))\} \vee (H_3 \cdot x \cdot y \cdot H_2(x \cdot y))$$

したがって，前節で示された三値論理和関数は最終的に基本演算子7個で実現できることになる。つぎに，上記のアルゴリズムの有効性を証明するために2，3の例題を示す。

例題1：三値論理積（mod-3）関数 $f_{x \otimes y}$ を簡単化してみる。

関数 $f_{x \oplus y}$ の真理値表を Table-(9) に示してある。

$x \backslash y$	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

Table-(9)：Truth table for $f_{x \otimes y}$

いま，この関数をラグランジュの関数を用いて展開すると，

$$f_{x \otimes y} = L_1(x) \cdot L_1(y) \vee L_2(x) \cdot L_2(y) \vee 2(L_1(x) \cdot L_2(y) \vee 2(L_2(x) \cdot L_1(y))$$

となる。

ここで，Min $(x . y) = x$，Max$(y, x) = y$ としてアルゴリズムのIVまでを適用すると

$$f_{x \otimes y} = L_1(x) \cdot L_1(y) \vee L_2(x) \cdot L_2(y) \vee 2(L_1(x) \cdot L_2(y))$$

となり，上式にアルゴリズムの V)，VI) を適用すると最終的に

$$f_{x \otimes y} = (X \cdot H_2(x) \cdot H_3) \vee (Y \cdot H_1(H_2(x))) = (x \cdot y \cdot H_2(x \cdot y) \cdot H_3) \vee (x \cdot y \cdot H_1(H_2(x \cdot y)))$$

となる。

いま取扱った $f_{x \otimes y}$ の関数ではあらたな変数X，Yの組に対して1価関数となっているのでX，Yのみで展開していっても別段問題はないが，多価関数となるような場合にはもとの変数 x，y をも考慮しなければならない。

例題2：Table-(10) に示される様な関数を簡単化してみる。

$x \backslash y$	0	1	2
0	2	0	2
1	1	1	1
2	2	1	1

Table-(10)：Truth table for $f_{x, y}$.

真理値表よりラグランジュ関数を用いて展開すると

$$f_{x, y} = L_1(x) \cdot L_0(y) \vee L_1(x) \cdot L_1(y) \vee L_1(x) \cdot L_2(y) \vee L_2(x) \cdot L_1(y) \vee L_2(x) \cdot L_2(y) \vee 2(L_0(x) \cdot L_0(y)) \vee 2(L_2(x) \cdot L_0(y)) \vee 2(L_0(x) \cdot L_2(y))$$

となる。

そこで簡単化のアルゴリズムを適用していくわけであるが，この関数の場合には $(x, y) = (0, 1)$ に対して2価関数となるのでもとの変数 x をも用いて展開するとつぎのようになる。

$$f_{x, y} = x \cdot L_0(x) \cdot L_1(y) \vee L_1(x) \cdot L_1(y) \vee L_1(x) \cdot L_2(y)$$

上式にアルゴリズムのV)，VI)を適用すると，

$$f_{x, y} = x \cdot H_1(H_2(x \vee y)) \vee x \cdot y \cdot H_2(x \vee y) \cdot H_3 \vee H_1(x \cdot y) \cdot H_2(x \vee y)$$

となる。

以上により，ここで与えた簡単化手法がかなり有効なものであるということが実証されたと思う。

なお，2変数の応用演算子として3を法とする含意，および対等の真理値は Table-(11) のようになり，基本演算子 $H_1(x)$，$H_2(x)$，H_3 を用いて表わすとつぎのようになる。

$$f_{x \to y} = (y \vee H_1(x)) \vee (H_1(H_1(y)) \cdot H_1(H_2(x))) \vee H_3$$

$$f_{x \rightleftarrows y} = H_1(H_1(x \cdot y)) \cdot H_1(H_1(x \vee y)) \cdot H_3 \vee H_1(x \vee y) \vee x \cdot y$$

$x \backslash y$	$x \to y$			$x \rightleftarrows y$		
	0	1	2	0	1	2
0	2	2	2	2	1	0
1	1	1	2	1	1	1
2	0	1	2	0	1	2

Table-(11)：Truth table for $x \to y$, $x \rightleftarrows y$.

$x \backslash y$	Sum			Carry		
	0	1	2	0	1	2
0	0	1	2	0	0	0
1	1	2	0	0	0	1
2	2	0	1	0	1	1

Table-(12)：Operational table for Ternary half adder circuit.

2-4：三値半および全加算器の関数表現とその回路構成

本節では三値論理演算回路の中で重要とおもわれる2つの回路について述べることにする。

通常，加減算回路は並列型にしろ直列型にしろ半加算器および全加算器で構成されている。演算部の最小桁（並列型の場合）を半加算器が，それ以外の桁は全加算器がうけもつことになる。したがって，三値論理演算部で主体となる半加算器と全加算器の回路を簡単化することは三値情報処理機械を構成していくうえで重要なことである。

このような観点からここでは半加算器および全加算器の機能を簡単な関数で表現するとともに，それにもとずく回路構成について述べる。

2-4-1：三値半加算器の構成

三値半加算器の動作を表わす真理値表を Table-(12) に示す。Sum を表現する関数は前に記した $fx \oplus y$ の関数に相当するので結果をそのまま使うことにする。

$$fx \oplus y = \{(x \vee y \vee H_1(H_2(x \cdot y))) \cdot (H_1(x \cdot y) \vee H_1(H_2(x \vee y)))\} \vee H_3 \cdot x \cdot y \cdot H_2(x \cdot y))$$

一方，Carry を表わす関数は Table-(12) より

$$fx \oplus y,\ \mathrm{Carry} = L_1(x) \cdot L_2(y) \vee L_2(x) \cdot L_1(y) \vee L_2(x) \cdot L_2(y)$$

となり簡単化手法を用いて書換えるとつぎのようになる。

$$fx \oplus y.\ \mathrm{Carry} = x \cdot H_2(y) \cdot H_3 = x \cdot y \cdot H_2(x \vee y) \cdot H_3$$

したがって，三値半加算器の動作を記述するものとしては，$fx \oplus y$ と $fx \oplus y$, Carryの2つの関数が合成されたものとなる。両関数を使って回路を構成すると Fig-(5) のようになる。

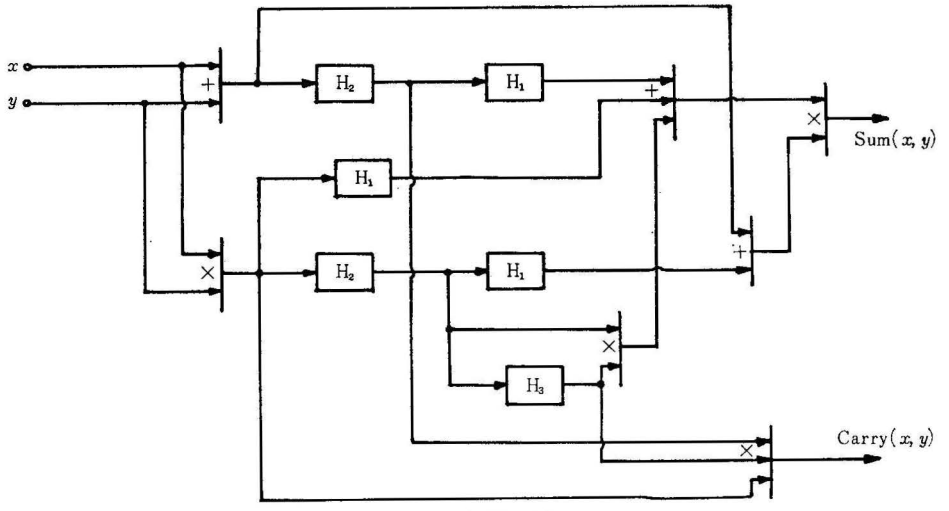

Fig-(5) Ternary half adder circuit.

2-4-2：三値全加算器の構成

三値全加算器の動作を表わす真理値表が Table-(13) に示されている。ただし，Zは前段の加算器から桁上げをあらわすものである。なお，Z＝2の場合は一般の加算器ではおこりえないのでここでは記載してない。

	Sum								
Z	0			1			2		
x \ y	0	1	2	0	1	2	0	1	2
0	0	1	2	1	2	0	※	※	※
1	1	2	0	2	0	1	※	※	※
2	2	0	1	0	1	2	※	※	※
0	0	0	0	0	0	1	※	※	※
1	0	0	1	0	1	0	※	※	※
2	0	1	1	1	0	0	※	※	※
	Carry								

Table-(13)：:Operational table for Ternary full adder circuit.

表からもわかるように，全加算器の場合はx，y，Zと三変数関数となるので簡単化はなかなか面倒である。ここでは全加算器の Sum を半加算器の Sum を表わす関数 $fx\oplus y$ と桁上げ信号Zとの2変数関数と考えて簡単化していった。その結果，Sum を表わす関数は，

$$fx\oplus y\oplus z=(H_1(H_2(Z\cdot fx\oplus y))\vee Z\vee fx\oplus y)\cdot(H_1(Z\cdot fx\oplus y)\vee H_1(H_2(Z\vee fx\oplus y))).$$

となる。

一方，全加算器の Carry は半加算器の動作を記述する関数 $fx\oplus y$，$fx\oplus y$，Carry と桁上げ信号Zとの3変数関数として簡単化していった。その結果，Carry を表わす関数は，

$$fx\oplus y\oplus z,\ \mathrm{Carry}=(Z\vee fx\oplus y,\ \mathrm{Carry})\cdot(fx\oplus y,\ \mathrm{Carry}\vee H_2(Z\vee fx\oplus y))$$

となる。

以上2つの式より三値全加算器の回路は Fig-(6) のようになる。

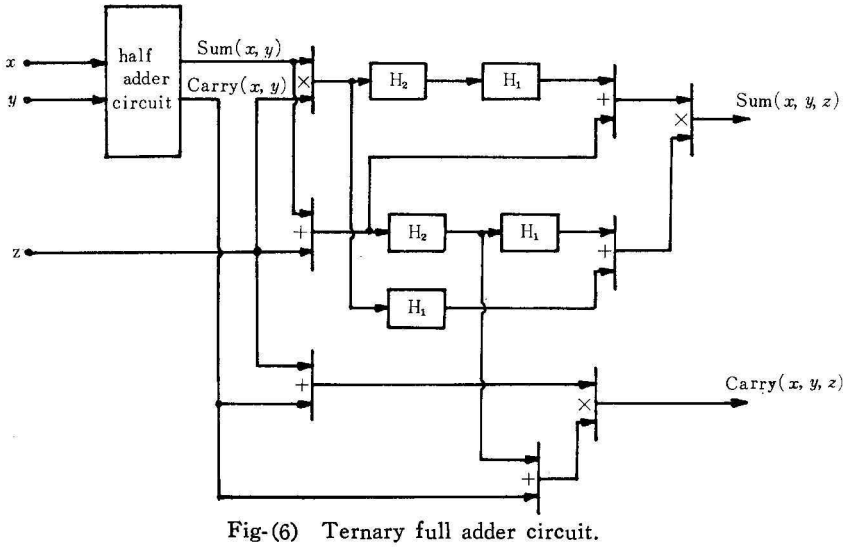

Fig-(6) Ternary full adder circuit.

3. あとがき

本論文では，

Ⅰ）新しい基本演算子の定義。

Ⅱ）それらの具体的な回路構成。

Ⅲ）新しい基本演算子を用いた三値論理関数の表現，および簡単化。

Ⅳ）三値半加算器および全加算器の回路構成。

等について述べた。

本論文で定義した基本演算子はその数も少なく，しかも回路構成が容易である，といった利点を有している。また，任意に与えられた関数を基本演算子で表現したあと簡単な表現式にもってゆくことができるように演算子の性質を列記した。したがって，与えられた関数の真理値表を作成し，それを基本演算子で示したあと演算子の性質を用いて簡単化してゆけばよい。

本論文の最後の節では三値半加算器および全加算器を基本演算子でもって構成した。その結果，回路に要する素子数は従来発表されている回路に比べて少なく，安定性の面でも保証できるものと考える。

それゆえ，筆者の定義した基本演算子はかなり有効なものであるといえる。

なお，今後の課題としては，

Ⅰ）基本演算子 $H_2(x)$ に用いられている負性抵抗素子を通常のトランジスタを用いて構成する。

Ⅱ）三値情報処理系の構成。

等があげられる。

4. 謝　辞

本研究をすすめるにあたり日ごろ御指導，御助言をいただいている東北大学・木村正行教授，ならびに丸岡章助教授に深く感謝の意を表する。

文　献

(1) E. L. Post："Introduction to a general theory of elementary propositions" Amerjcan Journal of Mathematics. XL Ⅲ (1921)。

(2) 今西，村中："三値論理の一手法"，信学論 Vo 155-D (№-7)。

(3) 三根，長谷川，他："三値論理回路の一構成" 信学論，Vol-51-C (№-12)。

(4) 今西，村中，他："三値論理回路の簡単化" 信学論，Vol. 56-D (№-7)。

(5) 安宅：ブール代数（共立出版）p 207～

(6) 田中，田原："三値論理関数の完全性と Polypheck" 信学論，Vol. 53-C (№-2)。

(7) M. Miyakawa："Functional Completeness and Structure of three-Valued Logic I, — Classification of P_3 —" Reserches of the Electrotecnical Laboratory, №-717. June (1971)。

(8) 石塚："エサキダイオード・トランジスタ多しきい値論理回路の一設計法" 信学論，Vol. 54-C (№-8)。