A Tabular Minimization Procedure for Ternary Switching Functions

ROY D. MERRILL, JR., MEMBER, IEEE

Abstract—**A tabular minimization procedure for ternary switching functions is developed. The theory is analogous to that used in the McCluskey simplification method for Boolean functions. Using the ternary function truth table, the procedure provides a systematic method of applying a limited set of reduction rules in a converging process for obtaining a minimal irredundant form of the function. It is shown how the procedure can be used to derive simplified expressions for arbitrary ternary functions in terms of a particularly attractive system of threshold gating functions. The procedure has more general applications in providing a simple method of finding, for any given function, all binary variables of the function and all variables of which the function is independent.**

I. Introduction

ALTHOUGH the machine states as well as the number representation of virtually every digital computer are implemented with binary logic, there are other systems such as the ternary and quinary that also could be used. With the advancing electronic technology, there are indications that ternary logic, particularly, can be reliably and economically implemented. Moreover, there have been several instances reported recently where ternary logic and switching networks have found practical application. For example, the SET'UN, a Russian stored-program computer, incorporates the ternary number system in its arithmetic unit to take advantage of the relative ease of representing negative and positive numbers and the simplicity of performing round-off in the ternary representation [1], [2]. Also many proposed content addressable memories utilize tristable switching and memory elements to accomplish masking and associated operations [3]. Ternary switching theory has been proposed as a useful means of designing hazard-free binary gating, and series-parallel contact networks [4–5]. NASA has actively supported the development of tristable fluid logic devices for hydraulic switching systems [6]. Still another area is the use of ternary logic redundancy in binary networks to improve operating reliability [7].

This paper develops a procedure for simplifying ternary switching functions represented in truth table form. The procedure is shown to be a simple systematic method of determining for the given function and all its subfunctions (made up of subsets of the rows of the original truth table) which have simplifications, all binary variables of the functions, and all variables of which the functions are independent. The results obtained from the procedure are generally useful for any system of ternary gating functions which might be considered. This is illustrated with a system composed of one set of functionally complete threshold gating functions.

Several authors have considered the problem of simplification with respect to some chosen system of gating functions [8–10, and 14]. The procedure developed in this paper considers switching function simplification from a somewhat more general viewpoint so that the results should be applicable in any logic system used for design.

II. Definitions

Let the vector space V^n be the set of n-tuples $V_i=(v_{i1}, v_{i2}, \cdots, v_{in})$, $i=1, 2, \cdots, 3^n$, representing all possible assignments of the truth values -1, 0, and $+1$ to the n variables $x_1, x_2, \cdots, x_n$. A *ternary switching function* $f(x_1, x_2, \cdots, x_n)$, or simply, $f(X)$, defines a partition of V^n into the three truth sets f^-, f^0, and f^+ which are mapped by f into -1, 0, and $+1$, respectively. Any two of these three sets uniquely represent f. Such a representation is the truth table form of f where each vector of the two sets chosen constitutes a row of the table. The form used in this discussion will be

$$\begin{array}{c} x_1x_2\cdots x_n \quad f(X) \\ \left[\begin{array}{c:c} & +1 \\ & +1 \\ f^+ & \cdot \\ & \cdot \\ & \cdot \\ & +1 \\ \hdashline & -1 \\ & \cdot \\ f^- & \cdot \\ & \cdot \\ & -1 \end{array}\right]. \end{array}$$

Later developments in this paper make extensive use of the 1-variable ternary switching functions. These functions, called transforms and denoted by $\psi_i(x)$, $i=0, \pm 1, \cdots, \pm 13$, are given in Table I.

III. The Simplification Problem

Regardless of the system of gating functions which might be used in logical design, it is desirable to first

Manuscript received September 15, 1965; revised May 2, 1966. The research reported in this paper was sponsored in part by the Air Force, under Contract AF 33(657)-8777, and the Lockheed Independent Research Program.

The author is with the Electronic Sciences Laboratory, Lockheed Missiles and Space Company, Palo Alto, Calif.

TABLE I
TRANSFORMS OF x

	$\psi_i(x)$																										
x \ i	−13	−12	−11	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	0	1	2	3	4	5	6	7	8	9	10	11	12	13
−	−	0	+	−	0	+	−	0	+	−	0	+	−	0	+	−	0	+	−	0	+	−	0	+	−	0	+
0	−	−	−	0	0	0	+	+	+	−	−	−	0	0	0	+	+	+	−	−	−	0	0	0	+	+	+
+	−	−	−	−	−	−	−	−	−	0	0	0	0	0	0	0	0	0	+	+	+	+	+	+	+	+	+

reduce the switching function to its simplest form. In this development the criteria for simplicity are the following: consider all decompositions of the switching function whose − and + truth sets are included in the − and + truth sets, respectively, of the original function.[1] (These subfunctions are generated, in effect, by considering all combinations of the rows of the truth table for the function.) For each decomposition, or subfunction, determine all input variables for which the function is independent. Further, determine all binary input variables of the subfunction; that is, variables for which two of the three truth states are indistinguishable by the function.

The following example will illustrate the general approach taken in the simplification procedure. Consider the function in Table II(a). Note that f is independent of x_1 [denoted by the asterisks in Table II(b) and (c)] and dependent only on x_2 being 0, and +1 or −1 [denoted by the grouping of +1 and −1 in parentheses in Table II(c)].

In order to reduce confusion where there may be many subfunctions with simplifications, we express the elements of the rows in the truth table for f in terms of the transforms $\psi_i(x)$ such that for every $V_i \in f^+$ or f^- form a *functional* (an n-tuple of transforms):

$$g_i(X) = [g_{i1}(x_1), g_{i2}(x_2), \cdots, g_{in}(x_n)]$$

where, for $V_i \in f^+$,

$$g_{ij}(x_j) = \begin{cases} \psi_1(x_j) \Leftrightarrow v_{ij} = -1 \\ \psi_3(x_j) \Leftrightarrow v_{ij} = \quad 0 \\ \psi_9(x_j) \Leftrightarrow v_{ij} = +1 \end{cases}, \text{ all } j \tag{1}$$

and where, for $V_i \in f^-$,

$$g_{ij}(x_j) = \begin{cases} \psi_{-1}(x_j) \Leftrightarrow v_{ij} = -1 \\ \psi_{-3}(x_j) \Leftrightarrow v_{ij} = \quad 0 \\ \psi_{-9}(x_j) \Leftrightarrow v_{ij} = +1 \end{cases}, \text{ all } j. \tag{2}$$

It may be noted that transforms $\psi_i(x)$, $i = \pm 1, \pm 3, \pm 9$, are two-valued.

[1] Any two of the three truth sets can be considered as the truth table representation for a function. Throughout this discussion the − and + truth sets are arbitrarily chosen as this representation. It should be clear in the development to follow that the results obtained are equally applicable to any of the three truth table forms representing either of the functions in each of the pairs: $[f, \psi_{-8}(f)]$, $[\psi_6(f), \psi_{-6}(f)]$, and $[\psi_2(f), \psi_{-2}(f)]$.

Clearly, every switching function has a unique representation in the set of functionals created using (1) and (2) and containing one functional for each row in the truth table form of the function. By using this representation we perform reduction operations similar to the Quine-McCluskey reduction technique used for binary functions [11].

The functional representation of f of Table II is given in Table III(a). For convenience only, the transform subscripts are used in representing the functionals. Further, since the transforms of a functional uniquely identify the truth set to which it is associated, we may dispense with the truth table column designating f and, if desired, the dotted line separating the truth sets.

The corresponding representation for Table II(b) is given in Table III(b) where, since f is independent of x_1, the transforms in the functionals for f are constant transforms. The fact that f does not distinguish between x_2 having the truth values −1 and +1 is reflected by $\psi_{10}(x_2)$ which maps ±1 into +1 and 0 into 0.

Still another form of simplification is possible as suggested in Table III(d). Specifically, it will be noted that the functional

$$[\psi_0(v_{i1}), \psi_7(v_{i2}), \psi_{-8}(v_{i3})] \cdot \begin{cases} = (0, +1, +1) \text{ for all } V_i \in f^+ \\ = (0, -1, -1) \text{ for all } V_i \in f^- \\ \neq (0, +1, +1) \text{ nor } (0, -1, -1) \quad \text{otherwise.} \end{cases} \tag{3}$$

Thus, this single functional uniquely represents the original switching function. It will be shown later that, in one system of gating functions at least, this simplified form can result in a most economical network realization. The important point to note, however, is the conciseness of representing the simplified form.

It is easy to verify that, in general, a functional $g_i(X)$ uniquely defines a partition of V^n into the three sets g_i^+, g_i^-, and g_i^0. Each $V_k \in g_i^+$ and g_i^- is mapped by g_i into an n-tuple, all of whose transforms *not* identically zero are +1 and −1, respectively. For each $V_k \in g_i^0$, either at least two transforms of $g_i(V_k)$ not identically zero will be opposite in sign, or at least, one transform of $g_i(V_k)$ not identically zero will take the truth value 0. Equation (3) illustrates this functional partitioning property for the example in Table III. There are several additional properties of both functionals and switching functions which require consideration.

TABLE II
SIMPLIFICATION OF f

$$\left[\begin{array}{ccc|c} 0 & + & - & + \\ + & + & - & + \\ - & + & - & + \\ - & - & - & + \\ 0 & - & - & + \\ + & - & - & + \\ \hline 0 & 0 & + & - \\ - & 0 & + & - \\ + & 0 & + & - \end{array}\right]$$
(a)

$$\left[\begin{array}{ccc|c} * & + & - & + \\ * & - & - & + \\ \hline * & 0 & + & - \end{array}\right]$$
(b)

$$\left[\begin{array}{ccc|c} * & (\pm) & - & + \\ \hline * & 0 & + & - \end{array}\right]$$
(c)

TABLE III
SIMPLIFIED FUNCTIONAL REPRESENTATION FOR f

$$\left[\begin{array}{ccc} 3 & 9 & 1 \\ 9 & 9 & 1 \\ 1 & 9 & 1 \\ 1 & 1 & 1 \\ 3 & 1 & 1 \\ 9 & 1 & 1 \\ \hline -3 & -3 & -9 \\ -1 & -3 & -9 \\ -9 & -3 & -9 \end{array}\right]$$
(a)

$$\left[\begin{array}{ccc} 13 & 9 & 1 \\ 13 & 1 & 1 \\ \hline -13 & -3 & -9 \end{array}\right]$$
(b)

$$\left[\begin{array}{ccc} 13 & 10 & 1 \\ \hline -13 & -3 & -9 \end{array}\right]$$
(c)

$$[0 \quad 7 \quad -8]$$
(d)

IV. COVERING RELATION AND REDUCTION THEOREMS

Let g, h, and l each be ternary functionals or switching functions with $-$, $+$, and 0 truth sets: g^-, h^-, l^-; g^+, h^+, l^+; and g^0, h^0, l^0, respectively. If for all $V_i \in V^n$, $h^+ \supset g^+$ and $h^- \supset g^-$, then h is said to *cover* g, denoted by $h > g$. This covering relation is illustrated in Table III where, for example,

$$[\psi_{13}(x_1), \psi_{10}(x_2), \psi_1(x_3)] > [\psi_3(x_1), \psi_1(x_2), \psi_1(x_3)]$$

Clearly, by this definition, if $l > g$ and $l > h$, then $g^+ \cap h^- = \phi$, the null set. Also, $g^- \cap h^+ = \phi$. The covering relation is transitive, that is, if $l > h$ and $h > g$, then $l > g$. Moreover, it is easily shown that if $h > g$ and $g > h$, then $h^+ = g^+$, $h^- = g^-$, and $h^0 = g^0$. This property is written $h \sim g$. In Table III, note that $f \sim [\psi_0(x_1), \psi_7(x_2), \psi_{-8}(x_3)]$.

It is now possible to formally present the reduction rules and the simplification procedure illustrated in Table III.

Theorem 1

Let $g(X) = [g_1(x_2), g_2(x_2), \cdots, g_n(x_n)]$ and $h(X) = [h_1(x_1), h_2(x_2), \cdots, h_n(x_n)]$ be two functionals such that $g_i = h_i$ for every i except $i = j$. There exists a functional $l(X)$ such that $l^+ = g^+ \cup h^+$ and $l^- = g^- \cup h^-$ if and only if (g_j, h_j) is one of the pairs in Table IV.

The proof of Theorem 1 in the Appendix provides the following method of finding $l(X)$: select $l_i = g_i$ for every $i \neq j$ and select l_j as a function of (g_j, h_j) from Table IV. It will be clear later that only the combinations of Table IV(a) need be considered in carrying out the simplification procedure. The simplifications derived from Table IV(b) will always be realized by two iterative applications using Table IV(a), together with Theorem 2. Also, all combinations in Table IV(c) are degenerate in that every case either $g < h$ or $h < g$.

Theorem 2

Let $g(X)$ and $h(X)$ be two functionals such that $g_i \neq h_i$ for all i. There exists a functional $l(X)$ such that $l^+ = g^+ \cup h^+$ and $l^- = g^- \cup h^-$ if and only if each (g_i, h_i) is one of the pairs of transforms in Table V.

The Theorem 2 proof in the Appendix provides the following method of finding $l(X)$: select each l_i as a function of (g_i, h_i), for all i, from Table V.

V. SIMPLIFICATION PROCEDURE

Using Theorems 1 and 2, a simplification procedure for ternary switching functions can be systematized in a manner analogous to the Quine-McCluskey procedure [11]. First, a set of functionals, one for each $V_i \in f^+$ and f^-, is formed according to (1) and (2). These are called *original functionals*.

Next, the original functionals are grouped according to their corresponding truth sets and compared by pairs for reductions possible under Theorem 1. The new covering functionals generated by this process are again grouped according to the truth set being covered and Theorem 1 is again applied. This procedure is repeated iteratively until no new functionals can be created. This step is greatly simplified by noting from the Theorem 1 proof that only the first six combinations in Table IV(a) arise for $+$ truth set original functionals and their subsequently created covering functionals. Similarly, only the second group of six combinations in Table IV(a) arise

TABLE IV

THEOREM 1 REDUCTION RULES

	(g_j, h_j) or (h_j, g_j)		l_j
#1	ψ_1	ψ_3	ψ_4
	ψ_1	ψ_9	ψ_{10}
	ψ_3	ψ_9	ψ_{12}
	ψ_4	ψ_{10}	ψ_{13}
	ψ_4	ψ_{12}	ψ_{13}
	ψ_{10}	ψ_{12}	ψ_{13}
#2	ψ_{-1}	ψ_{-3}	ψ_{-4}
	ψ_{-1}	ψ_{-9}	ψ_{-10}
	ψ_{-3}	ψ_{-9}	ψ_{-12}
	ψ_{-4}	ψ_{-10}	ψ_{-13}
	ψ_{-4}	ψ_{-12}	ψ_{-13}
	ψ_{-10}	ψ_{-12}	ψ_{-13}

	(g_j, h_j) or (h_j, g_j)		l_j
#3	ψ_2	ψ_{-6}	ψ_{-7}
	ψ_2	ψ_8	ψ_{11}
	ψ_6	ψ_8	ψ_5
	ψ_6	ψ_{-2}	ψ_7
	ψ_{-6}	ψ_{-8}	ψ_{-5}
	ψ_{-8}	ψ_{-2}	ψ_{-11}
#4	ψ_5	ψ_{-5}	ψ_0
	ψ_7	ψ_{-7}	ψ_0
	ψ_{11}	ψ_{-11}	ψ_0
	ψ_{13}	ψ_{-13}	ψ_0

(a)

(g_j, h_j) or (h_j, g_j)		l_j
ψ_1	ψ_6	ψ_7
ψ_1	ψ_{12}	ψ_{13}
ψ_1	ψ_{-3}	ψ_{-2}
ψ_1	ψ_{-6}	ψ_{-5}
ψ_1	ψ_{-9}	ψ_{-8}
ψ_1	ψ_{-12}	ψ_{-11}
ψ_2	ψ_{-9}	ψ_{-7}
ψ_2	ψ_9	ψ_{11}
ψ_3	ψ_8	ψ_{11}
ψ_3	ψ_{10}	ψ_{13}

(g_j, h_j) or (h_j, g_j)		l_j
ψ_3	ψ_{-1}	ψ_2
ψ_3	ψ_{-8}	ψ_{-5}
ψ_3	ψ_{-9}	ψ_{-6}
ψ_3	ψ_{-10}	ψ_{-7}
ψ_4	ψ_{-9}	ψ_{-5}
ψ_{-1}	ψ_{-6}	ψ_{-7}
ψ_{-1}	ψ_{-12}	ψ_{-13}
ψ_{-1}	ψ_3	ψ_2
ψ_{-1}	ψ_6	ψ_5
ψ_{-1}	ψ_9	ψ_8

(g_j, h_j) or (h_j, g_j)		l_j
ψ_{-1}	ψ_{12}	ψ_{11}
ψ_{-2}	ψ_9	ψ_7
ψ_{-2}	ψ_{-9}	ψ_{-11}
ψ_{-3}	ψ_{-8}	ψ_{-11}
ψ_{-3}	ψ_{-10}	ψ_{-13}
ψ_{-3}	ψ_1	ψ_{-2}
ψ_{-3}	ψ_8	ψ_5
ψ_{-3}	ψ_9	ψ_6
ψ_{-3}	ψ_{10}	ψ_7
ψ_{-4}	ψ_9	ψ_5

(b)

(g_j, h_j) or (h_j, g_j)		l_j
ψ_0	ψ_{-13}	ψ_0
ψ_0	ψ_{-12}	ψ_0
.	.	.
.	.	.
.	.	.
ψ_0	ψ_{+13}	ψ_0
ψ_1	ψ_4	ψ_4
ψ_1	ψ_7	ψ_7
ψ_1	ψ_{10}	ψ_{10}
ψ_1	ψ_{13}	ψ_{13}
ψ_1	ψ_{-2}	ψ_{-2}
ψ_1	ψ_{-5}	ψ_{-5}
ψ_1	ψ_{-8}	ψ_{-8}
ψ_1	ψ_{-11}	ψ_{-11}
ψ_3	ψ_{11}	ψ_{11}
ψ_3	ψ_{12}	ψ_{12}
ψ_3	ψ_{13}	ψ_{13}
ψ_3	ψ_{-5}	ψ_{-5}
ψ_3	ψ_{-6}	ψ_{-6}
ψ_3	ψ_{-7}	ψ_{-7}
ψ_4	ψ_{13}	ψ_{13}
ψ_4	ψ_{-5}	ψ_{-5}
ψ_6	ψ_5	ψ_5
ψ_6	ψ_7	ψ_7

(g_j, h_j) or (h_j, g_j)		l_j
ψ_8	ψ_5	ψ_5
ψ_8	ψ_{11}	ψ_{11}
ψ_9	ψ_5	ψ_5
ψ_9	ψ_6	ψ_6
ψ_9	ψ_7	ψ_7
ψ_9	ψ_8	ψ_8
ψ_9	ψ_9	ψ_9
ψ_9	ψ_{10}	ψ_{10}
ψ_9	ψ_{11}	ψ_{11}
ψ_9	ψ_{12}	ψ_{12}
ψ_9	ψ_{13}	ψ_{13}
ψ_{10}	ψ_7	ψ_7
ψ_{10}	ψ_{13}	ψ_{13}
ψ_{12}	ψ_{11}	ψ_{11}
ψ_{12}	ψ_{13}	ψ_{13}
ψ_{-1}	ψ_{-4}	ψ_{-4}
ψ_{-1}	ψ_{-7}	ψ_{-7}
ψ_{-1}	ψ_{-10}	ψ_{-10}
ψ_{-1}	ψ_{-13}	ψ_{-13}
ψ_{-1}	ψ_2	ψ_2
ψ_{-1}	ψ_5	ψ_5
ψ_{-1}	ψ_8	ψ_8
ψ_{-1}	ψ_{11}	ψ_{11}
ψ_{-3}	ψ_{-11}	ψ_{-11}

(g_j, h_j) or (h_j, g_j)		l_j
ψ_{-3}	ψ_{-12}	ψ_{-12}
ψ_{-3}	ψ_{-13}	ψ_{-13}
ψ_{-3}	ψ_5	ψ_5
ψ_{-3}	ψ_6	ψ_6
ψ_{-3}	ψ_7	ψ_7
ψ_{-4}	ψ_{-13}	ψ_{-13}
ψ_{-4}	ψ_5	ψ_5
ψ_{-6}	ψ_{-5}	ψ_{-5}
ψ_{-6}	ψ_{-7}	ψ_{-7}
ψ_{-8}	ψ_{-5}	ψ_{-5}
ψ_{-8}	ψ_{-11}	ψ_{-11}
ψ_{-9}	ψ_{-5}	ψ_{-5}
ψ_{-9}	ψ_{-6}	ψ_{-6}
ψ_{-9}	ψ_{-7}	ψ_{-7}
ψ_{-9}	ψ_{-8}	ψ_{-8}
ψ_{-9}	ψ_{-9}	ψ_{-9}
ψ_{-9}	ψ_{-10}	ψ_{-10}
ψ_{-9}	ψ_{-11}	ψ_{-11}
ψ_{-9}	ψ_{-12}	ψ_{-12}
ψ_{-9}	ψ_{-13}	ψ_{-13}
ψ_{-10}	ψ_{-7}	ψ_{-7}
ψ_{-10}	ψ_{-13}	ψ_{-13}
ψ_{-12}	ψ_{-11}	ψ_{-11}
ψ_{-12}	ψ_{-13}	ψ_{-13}

(c)

TABLE V

THEOREM 2 REDUCTION RULE

(g_i, h_i)		l_i
ψ_1	ψ_{-3}	ψ_{-2}
ψ_1	ψ_{-9}	ψ_{-8}
ψ_1	ψ_{-12}	ψ_{-11}
ψ_3	ψ_{-1}	ψ_2

(g_i, h_i)		l_i
ψ_3	ψ_{-9}	ψ_{-6}
ψ_3	ψ_{-10}	ψ_{-7}
ψ_9	ψ_{-1}	ψ_8
ψ_9	ψ_{-3}	ψ_6

(g_i, h_i)		l_i
ψ_9	ψ_{-4}	ψ_5
ψ_4	ψ_{-9}	ψ_{-5}
ψ_{10}	ψ_{-3}	ψ_7
ψ_{12}	ψ_{-1}	ψ_{11}
ψ_{13}	ψ_{-13}	ψ_0

for $-$ truth set original functionals and their subsequently created covering functionals. The example in Table VI illustrates this step. Column 1 contains the original functionals and columns 2, 3, and 4 contain covering functionals created using Theorem 1, one for each iteration. (In the simplification tableau, e.g., Table VI, covered functionals are checked and redundant functionals are struck-out.)

In the third step, each functional of the $+$ truth set is compared with every functional of the $-$ truth set for reductions possible under Theorem 2. The functionals created under this step are grouped separately from those of Steps 1 and 2 in carrying out the next step. Columns 5, 6, 7, and 8 of Table VI contain this grouping for the example as created from columns 1, 2, 3, and 4, respectively.

In the fourth step, all functionals created in Step 3 are compared by pairs using Theorem 1. This step is greatly simplified by noting that in the first iteration only transforms in the third group of six combinations of Table IV(a) arise. In the second and subsequent iteration in this step only transforms in the third and fourth, or final, group of combinations in Table IV(a) arise. It is easily verified by enumeration that at the completion of Step 4, the set of all unchecked functionals are exactly those obtained if comparisons are made which require using Table IV(b). However, sometimes the coverings given in Table IV(c) will greatly simplify the reduction procedure. For example, in Table VI, each functional with one or more ψ_0 covers all functionals which have the same corresponding transforms except in the position of the ψ_0. Hence, we designate these latter functionals with small crosses as in columns 5, 6, and 7, since they need not be considered further as a group. Individually, they must be compared with the remaining uncrossed functionals, however, since additional simplification may be possible under Theorem 1.

Following the exhaustive application of Theorem 1 in Step 4, it should be clear that the unchecked and uncrossed functionals are *prime functionals;* that is, functionals which are covered only by the original switching function itself. Further, the set of prime functionals together cover the original functionals created using (1) and (2). In Table VI, [(13, 3, 1), (-2, 5, -5), and (7, 0, -8)] is the set of all prime func-

TABLE VI
SIMPLIFICATION PROCEDURE EXAMPLE

1

√	9	3	1
√	9	1	1
√	9	9	1
√	1	9	1
√	1	3	1
√	1	1	1
√	1	9	3
√	3	3	1
√	−3	−3	−9
√	−3	−1	−9
√	−3	−9	−9

2

√	9	4	1
√	9	12	1
√	10	3	1
√	12	3	1
√	9	10	1
√	10	1	1
√	10	9	1
√	1	12	1
√	1	10	1
√	4	3	1
√	1	9	4
√	1	4	1
√	−3	−4	−9
√	−3	−12	−9
√	−3	−10	−9

3

√	9	13	1
	~~9~~	~~13~~	~~1~~
√	10	4	1
	~~9~~	~~13~~	~~1~~
√	10	12	1
	~~10~~	~~4~~	~~1~~
	~~10~~	~~12~~	~~1~~
	13	3	1
	~~13~~	~~3~~	~~1~~
	~~13~~	~~3~~	~~1~~
√	10	10	1
	~~10~~	~~10~~	~~1~~
√	1	13	1
	~~1~~	~~13~~	~~1~~
√	−3	−13	−9
	~~−3~~	~~−13~~	~~−9~~

4

√	10	13	1
	~~10~~	~~13~~	~~1~~
	~~10~~	~~13~~	~~1~~
	~~10~~	~~13~~	~~1~~

5

×	6	2	−8
×	6	−6	−8
×	6	−7	−8
×	6	−2	−8
×	6	−8	−8
×	6	−11	−8
×	6	6	−8
×	6	8	−8
×	6	5	−8
×	−2	6	−8
×	−2	8	−8
×	−2	5	−8
×	−2	2	−8
×	−2	−6	−8
×	−2	−7	−8
×	−2	−2	−8
×	−2	−8	−8
√	−2	6	−6
√	−2	8	−6
√	−2	5	−6

6

×	6	−5	−8
×	6	11	−8
×	7	2	−8
×	7	−6	−8
×	7	−7	−8
×	6	7	−8
×	7	−2	−8
×	7	−8	−8
×	7	6	−8
×	7	8	−8
×	−2	11	−8
×	−2	7	−8
√	−2	6	−5
√	−2	8	−5
	−2	5	−5
×	−2	−5	−8

7

√	6	0	−8
×	7	−5	−8
×	7	11	−8
×	7	7	−8
√	−2	0	−8

8

7	0	−8

tionals. Moreover, all three are *essential;* that is, for each of the prime functionals there exists at least one original functional which is not covered by any other prime functional. This can be seen in carrying out the fifth, and final, step of the simplification procedure: construction of the prime functional covering table as in Table VII. Generally, all prime functionals will not be essential, hence the covering table is used to select an *irredundant set of prime functionals.* Such a set has the property that removal of any one of the functionals from the set results in at least one of the original functionals not being covered.

Since Theorems 1 and 2 provide all possible ways of generating a new covering functional given two functionals, the simplification procedure produces all possible functionals each of which covers a subset of the original functionals. Moreover, by checking a functional only if at least one other functional covers it, we are assured that by exhaustive application of Theorems 1 and 2, the set of unchecked functionals will collectively cover all the original functionals. It follows that by the procedure given, one obtains the set of all prime functionals. Using the covering table then provides the means of selecting any of the possible irredundant sets of prime functionals.

TABLE VII
PRIME FUNCTIONAL COVERING TABLE FOR EXAMPLE OF TABLE VI

Original Functionals			Prime Functionals: 13 3 1	−2 5 −5	7 0 −8
9	3	1	x		x
9	1	1			x
9	9	1			x
1	9	1		x	x
1	3	1	x		x
1	1	1			x
1	9	3		x	
3	3	1	x		
−3	−3	−9		x	x
−3	−1	−9		x	x
−3	−9	−9			x

It should also be recognized that one must consider the three combinations of the three truth sets taken two at a time and apply the simplification procedure to each before it is possible to select the irredundant set(s) having a minimal number of prime functionals.[1]

The irredundant set chosen to represent the switching function in logical design will depend on the group of gating functions used. In the next section, a particular group of gating functions is considered and it is shown how the truth table simplification procedure can be utilized in obtaining efficient realizations.

VI. Realizing Ternary Switching Functions with Threshold Logic

Threshold switching functions appear promising as a source of gating functions for logical design because of their algebraic and switching theoretic properties, as well as the relative simplicity of their circuit mechanizations for reasonable numbers of inputs [9], [12], and [13]. The group of threshold gating functions[2]

$$T_{1,1,\cdots,1;1/-1}^{y_1,y_2,\cdots,y_n}, \qquad T_{1,1,\cdots,1;n/-n}^{y_1,y_2,\cdots,y_n}$$

and the transform[3] $\psi_{-8}(y)$ will be used as one example for the application of the truth table simplification procedure.

Every ternary function $f(X)$ can be expressed in terms of the chosen gating functions as follows: Let $f_1, f_2, \cdots, f_r$ be the original functionals of f. From the definitions of (1) and (2), it is easily verified that the truth sets of f_i are identically the truth sets of

$$F_i(X) \triangleq T_{1,1,\cdots,1;n/-n}^{f_{i1}(x_1),f_{i2}(x_2),\cdots,f_{in}(x_n)} \tag{4}$$

where the transforms $f_{ij}(x_j)$ have the realizations of Table VIII which are expressed in terms of the chosen gating functions.

$f(X)$ then has the following *canonical* representation:

$$f(X) = T_{1,1,\cdots,1;1/-1}^{F_1(X),F_2(X),\cdots,F_r(X)}. \tag{5}$$

It is easily verified that the canonical expression (5) is unique and the selected group of gating functions is functionally complete. Hanson has also shown functional completeness by deriving a general expansion theorem for the group [9]. The general representation for $f(X)$ developed by Hanson [9] is equivalent to (5).

It is important to note that the set of functions obtained by (4) for all original functions of $f(X)$ has the basic property that if for some $V_k \in V^n$

$$F_i(V_k) \neq 0, \qquad 1 \leq i \leq r.$$

Then, either

$$F_j(V_k) = F_i(V_k) \quad \text{or} \quad F_j(V_k) = 0, \qquad j = 1, 2, \cdots, r.$$

Further, this property holds for any set of functionals generated for $f(X)$ in the simplification procedure. To see this, note that each functional covers some subset of the original functionals. But, from the definition of covering, if the functionals d, g, and h have the relationship $d<h$ and $g<h$, then $V_i \in d^+$ (or g^+) implies $V_i \in h^+$ and $V_i \in d^-$ (or g^-) implies $V_i \in h^-$. Hence, if $V_i \in h^+$, then either $V_i \in d^+$ (or g^+) or $V_i \in d^0$ (or g^0). The analogous situation also holds for $V_i \in h^-$. Thus, in (4) the functions for d and g, $D(X)$, and $G(X)$ have the above property; specifically, for some $V_k \in V^n$ if $D(V_k) \neq 0$, then either $G(V_k) = D(V_k)$ or $G(V_k) = 0$.

It follows, therefore, that if $(g_1, g_2, \cdots g_s)$ is a set of irredundant prime functionals for $f(X)$, then

$$f(X) = T_{1,1,\cdots,1;1/-1}^{G_1(X),G_2(X),\cdots,G_s(X)} \tag{6}$$

where the G_i are obtained from (4) given the g_i. For example, the function of Table II which is covered by $g = [\psi_0(x_1), \psi_7(x_2), \psi_{-8}(x_3)]$, has the threshold realization

$$T_{1;1/-1}^{G} = G = T_{1,-1;2/-2}^{\psi_7(x_2),x_3}.$$

Similarly, the function in Table VI has the realization

$$T_{1,1,1;1/-1}^{G_1,G_2,G_3}$$

where

$$G_1 = T_{1,1;2/-2}^{\psi_3(x_2),\psi_1(x_3)}$$
$$G_2 = T_{1,1,1;3/-3}^{\psi_{-2}(x_1),\psi_5(x_2),\psi_{-5}(x_3)}$$

and

$$G_3 = T_{1,1;2/-2}^{\psi_7(x_1),x_3}.$$

For the examples of Tables II and VI, the prime functionals are all *essential.* In general, this will not be the case. The function of Table IX illustrates the more general situation where not all prime functionals are essential: columns of the covering table with circled entries correspond to essential prime functionals.[4] There are five distinct irredundant prime functional sets in Table IX. The set $\{(3, 9, 10), (2, 6, -5), (-2, -6, 7),$

[2] The author's definition for threshold functions [13] is used: $f(X)$ is a threshold function if there exists weights $(w_1, w_2, \cdots, w_n)$ and thresholds (t_+, t_-) such that

$$f(X) = \begin{cases} +1 \Leftrightarrow \sum_{i=1}^{n} x_i w_i \geq t_+ \\ -1 \Leftrightarrow \sum_{i=1}^{n} x_i w_i \leq t_- \\ 0 \quad \text{otherwise} \end{cases}$$

which is written

$$f(X) \triangleq T_{w_1,w_2,\cdots,w_n;t_+/t_-}^{x_1,x_2,\cdots,x_n}$$

[3] In most threshold function mechanizations $\psi_{-8}(x)$ can be realized directly, for example,

$$T_{1,1,\cdots,1;t_+/t_-}^{\psi_{-8}(y_1),y_2,\cdots,y_n} = T_{-1,1,\cdots,1;t_+/t_-}^{y_1,y_2,\cdots,y_n}$$

and

$$\psi_{-8}(T_{1,1,\cdots,1;t_+/t_-}^{y_1,y_2,\cdots,y_n}) = T_{-1,-1,\cdots,-1;-t_-/-t_+}^{y_1,y_2,\cdots,y_n}$$

[4] The simplification procedure tableau for this example can be found on p. B-227 of [12].

TABLE VIII

NONTRIVIAL TRANSFORM THRESHOLD REALIZATIONS*

$\psi_1(x) = \psi_{-8}[T_{1,1;2/-2}^{x,-1}]$	$\psi_7(x) = T_{1,2,4;1/-1}^{-1,x,\psi_1(x)}$
$\psi_2(x) = \psi_{-8}[T_{1,1,3;1/-1}^{-1,x,\psi_1(x)}]$	$\psi_8(x) = x$
$\psi_3(x) = T_{1,1;1/-1}^{x,\psi_{-5}(x)}$	$\psi_9(x) = \psi_{-8}[T_{1,1;2/-2}^{\psi^{-8}(x),-1}]$
$\psi_4(x) = T_{1,1;1/-1}^{1,\psi_{-8}(x)}$	$\psi_{10}(x) = T_{1,2;1/-1}^{x,\psi_1(x)}$
$\psi_5(x) = T_{1,2;1/-1}^{-1,x}$	$\psi_{11}(x) = T_{1,2;1/-1}^{1,x}$
$\psi_6(x) = T_{1,1,3;1/-1}^{-1,\psi_{-8}(x),\psi_9(x)}$	$\psi_{12}(x) = T_{1,1;1/-1}^{1,x}$

* In general, $\psi_i(x) = \psi_{-8}[\psi_{-i}(x)]$; hence only one-half the nontrivial 1-place functions need be given.

TABLE IX

COVERING TABLE FOR THE THIRD EXAMPLE

Original Functionals	Prime Functionals: 7, 7, −11	−5, −5, −11	7, −11, −2	−11, −5, −2	7, −5, −2	−2, −6, 7	2, 6, −5	13, 13, 1	3, 9, 10
1, 1, 1	$\boxtimes$	$\otimes$	×	×	×			$\boxtimes$	
3, 1, 1		$\otimes$						$\boxtimes$	
9, 1, 1	$\boxtimes$		×		×			$\boxtimes$	
1, 3, 1		$\otimes$		×	×	$\otimes$		$\boxtimes$	
3, 3, 1		$\otimes$						$\boxtimes$	
9, 3, 1					×			$\boxtimes$	
1, 9, 1	$\boxtimes$						$\otimes$	$\boxtimes$	
3, 9, 1								$\boxtimes$	$\otimes$
9, 9, 1	$\boxtimes$							$\boxtimes$	
1, 9, 3							$\otimes$		
1, 3, 9						$\otimes$			
3, 9, 9									$\otimes$
−3, −3, −3	$\boxtimes$		×						
−3, −9, −3			×	×	×	$\otimes$			
−9, −9, −3		$\otimes$		×					
−3, −3, −9	$\boxtimes$						$\otimes$		
−9, −9, −9		$\otimes$							

$(-5, -5, -11)$, $(13, 13, 1)$, and $(7, 7, -11)\}$ has the minimum cost function: the number of nonconstant arguments in the functionals plus the number of functionals with more than one argument. This cost function represents the number of threshold gate inputs in (6) for a switching function without considering the transform realizations. If the latter realizations merit inclusion, still other cost functions might be used.

An Additional Consideration. It is known that through the simplification procedure alone, there is no assurance of obtaining a threshold realization with a minimal cost function of the above form. For example, in the simplification procedure rather than limit the original functionals to those generated from the truth table by (1) and (2), suppose we also consider functionals generated from vectors in the 0 truth set. The example in Table X

TABLE X

AN ADDITIONAL SIMPLIFICATION FOR THRESHOLD REALIZATIONS

	Irredundant Sets of Functionals: without (+1, 0, −1)			with (+1, 0, −1)	
Original Functionals	3 9 13	9 10 1	−9 −3 −12	9 13 1	−6 6 0
3 9 1	x				x
3 9 3	x				x
3 9 9	x				x
9 9 1		x		x	
9 1 1		x		x	
(9 3 1)				x	
−9 −3 −9			x		x
−9 −3 −3			x		x
(−9 −3 −1)					x

demonstrates one approach of doing this to simplify the representation (6) for the switching function. Note that by utilizing the 0 truth set vector $(+1, 0, -1)$, it is necessary to form two functionals; one by (1) and the other by (2). The simplification procedure then is carried out as before, however, in using the covering table to obtain the best irredundant set of functionals for forming (6), it is necessary to include as many functionals covering $(9, 3, 1)$ as there are covering $(-9, -3, -1)$. In this way we are assured that (6) evaluated for $X = (+1, 0, -1)$ will be 0. However, further research will be required before a unified method can be developed.

VII. CONCLUSIONS

In this paper, a truth table simplification procedure was developed by which irredundant sets of prime functionals can be obtained having minimal numbers of prime functionals. Further, it was shown how the procedure can be used to derive simplified expressions for an arbitrary ternary switching function in terms of a particular group of threshold gating functions. The procedure has more general application, also, in providing a simple, systematic method of finding all binary variables in the given function and all subfunctions (which can be found using subsets of the rows of the original truth table) as well as all variables of which these functions are individually independent.

APPENDIX

THEOREM PROOFS

Proof of Theorem 1

To prove an l exists if (g_j, h_j) occurs in Table IV we consider two cases.

Case 1: If either g_j or h_j equals ψ_0, then g or h, respectively, is independent of x_j. Suppose $g_j = \psi_0$, then it follows that $g > h$. Hence by selecting $l = g$ the theorem is satisfied. Similarly, if $h > g$ select $l = h$.

Case 2: If neither g_j nor h_j equals ψ_0, then from Table IV it is known that every V_k for which $g_j(v_{kj}) \neq 0$ implies that either $g_j(v_{kj}) = h_j(v_{kj})$ or $h_j(v_{kj}) = 0$. If $g_j(v_{kj}) = 0$, then there are no restrictions on $h_j(v_{kj})$. Therefore, $V_k \notin g^0$ implies either $g(V_k) = h(V_k)$ or $V_k \in h^0$. Further, $V_k \in g^0$ implies either $g(V_k) = h(V_k)$ or $V_k \notin h^0$. Hence by selecting $l_i = g_i$ for all $i \neq j$, and selecting l_j such that $g_j(v_{kj}) \neq 0$ implies $l_j(v_{kj}) = g_j(v_{kj})$ and $g_j(v_{kj}) = 0$ implies $l_j(v_{kj}) = h_j(v_{kj})$, the resulting l satisfies the "if" part of the theorem. The (g_j, h_j) enumerated in Table IV covers only combinations satisfying conditions in each of these cases.

To prove an l exists only if (g_j, h_j) occurs in Table IV, we first note that the existence of an $l(X)$ such that $l^+ = g^+ \cup h^+$ and $l^- = g^- \cup h^-$ implies $g^+ \cap h^- = \phi$ and $g^- \cap h^+ = \phi$. Hence the criterion under which two transforms may appear as g_j and h_j is that either $g_j(x) = h_j(x)$, or $g_j(x) = 0$, or $h_j(x) = 0$ for all truth values of x. The following itemized lists exhaust all possibilities under this criterion as can be verified from Table I.

1) ψ_1 and $\psi_{-12}, \psi_{-11}, \psi_{-9}, \psi_{-8}, \psi_{-6}, \psi_{-5}, \psi_{-3}, \psi_{-2}, \psi_0, \psi_3, \psi_4, \psi_6, \psi_7, \psi_9, \psi_{10}, \psi_{12}$, or ψ_{13}.
2) ψ_2 and $\psi_{-10}, \psi_{-9}, \psi_{-6}, \psi_{-1}, \psi_0, \psi_3, \psi_8, \psi_9, \psi_{11}$, or ψ_{12}.
3) ψ_3 and $\psi_{-10}, \psi_{-9}, \psi_{-8}, \psi_{-7}, \psi_{-6}, \psi_{-5}, \psi_{-1}, \psi_0, \psi_1, \psi_2, \psi_4, \psi_8, \psi_9, \psi_{10}, \psi_{11}, \psi_{12}$, or ψ_{13}.
4) ψ_4 and $\psi_{-9}, \psi_{-8}, \psi_{-6}, \psi_{-5}, \psi_0, \psi_1, \psi_3, \psi_9, \psi_{10}, \psi_{12}$, or ψ_{13}.
5) ψ_5 and $\psi_{-4}, \psi_{-3}, \psi_{-1}, \psi_0, \psi_6, \psi_8$, or ψ_9.
6) ψ_6 and $\psi_{-4}, \psi_{-3}, \psi_{-1}, \psi_0, \psi_1, \psi_5, \psi_7, \psi_8$, or ψ_9.
7) ψ_7 and $\psi_{-3}, \psi_{-2}, \psi_0, \psi_1, \psi_6, \psi_9$, or ψ_{10}.
8) ψ_8 and $\psi_{-4}, \psi_{-3}, \psi_{-1}, \psi_0, \psi_2, \psi_3, \psi_5, \psi_6, \psi_9, \psi_{11}$, or ψ_{12}.
9) ψ_9 and $\psi_{-4}, \psi_{-3}, \psi_{-2}, \psi_{-1}, \psi_0, \psi_1, \psi_2, \psi_3, \psi_4, \psi_5, \psi_6, \psi_7, \psi_8, \psi_{10}, \psi_{11}, \psi_{12}$, or ψ_{13}.
10) ψ_{10}, and $\psi_{-3}, \psi_{-2}, \psi_0, \psi_1, \psi_3, \psi_4, \psi_6, \psi_7, \psi_9, \psi_{10}, \psi_{12}$, or ψ_{13}.
11) ψ_{11} and $\psi_{-1}, \psi_0, \psi_2, \psi_3, \psi_8, \psi_9$, or ψ_{12}.
12) ψ_{12} and $\psi_{-1}, \psi_0, \psi_1, \psi_2, \psi_3, \psi_4, \psi_8, \psi_9, \psi_{10}, \psi_{11}$, or ψ_{13}.

Lists for transforms with negative subscripts can be obtained from the above tabulations by multiplying all transform subscripts by -1.

A comparison between Table IV and the above lists will verify that all allowable combinations of g_j and h_j have been included. Again, l is selected as specified in Cases 1 and 2. Since this proves the "only if" part of the theorem, the proof is complete.

Proof of Theorem 2

The existence of an $l(X)$ such that $l^+ = g^+ \cup h^+$ and $l^- = g^- \cup h^-$ implies that $g^+ \cap h^- = \phi$ and $g^- \cap h^+ = \phi$. Further, $\{\psi_1, \psi_3, \psi_4, \psi_9, \psi_{10}, \psi_{13}\}$ is the set of all transforms for which $\psi_i(x) \geq 0$ for all truth values of x, and $\{\psi_{-1}, \psi_{-3}, \psi_{-4}, \psi_{-9}, \psi_{-10}, \psi_{-12}, \psi_{-13}\}$ is the set of all transforms for which $\psi_i(x) \leq 0$ for all truth values of x. Hence by the definition of g and h given in Theorem 2, one and only one of the following three cases can arise for each $V_k \in V^n$.

Case	$V_k \in g^+$	$V_k \in g^-$	$V_k \in h^+$	$V_k \in h^-$
a	1	0	0	0
b	0	0	0	0
c	0	0	0	1

where $1 \triangleq$ yes and $0 \triangleq$ no.

For case (a), $V_k \in g^+$ and $V_k \in h^0$. Thus $g_i(v_{ki}) = +1$ for all i. But by the choice of l_i given in Table V, $V_k \in g^+$ implies that $l_i(v_{ki}) = +1$ for all i ecxept where $g_i = \psi_{13}$ in which case $l_i(v_{ki}) \equiv 0$. Hence, $V_k \in l^+$ for those V_k giving rise to case (b), and conversely.

For case (b), $V_k \in g^0$, h^0. This implies there exist a g_p and an h_q such that $g_p(v_{kp}) \neq +1$ and $h_q(v_{kq}) \neq -1$. By the choice of l_i, $l_p(v_{kp}) \neq +1$ and $l_q(v_{kq}) \neq -1$. Hence, $V_k \in l^0$ for all V_k of case (b). Conversely, when $V_k \in l^0$ there exist at least two transforms l_p and l_q not identically zero such that $-2 < l_p(v_{kp}) + l_q(v_{kq}) < +2$. Therefore, by the choice of the l_i, $V_k \in g_0$, h_0.

For case (c), $V_k \in g_0$ and $V_k \in h^-$ which implies $h_i(v_{ki}) = -1$ for all i. Hence, it is clearly that $V_k \in l^-$ for those $V_k \in g^0$ and h^-. Similarly, if $V_k \in l^-$ under the conditions of this case $h_i(v_{ki}) = -1$ and there exist at least two transforms g_p and g_q such that $-2 < g_p(v_{kp}) + g_q(v_{kq}) < +2$. Hence, $V_k \in l^-$ implies $V_k \in h^-$ and $g.^0$ Since these three cases are exhaustive on V^n, the proof is complete.

Acknowledgment

The author gratefully acknowledges the assistance of Dr. C. E. Duncan, Senior Member of the Lockheed Missiles and Space Company, Electronic Sciences Laboratory, and the helpful advice and consultation of O. Firschein.

References

[1] Y. A. Zhogolev, "The order code and an interpretative system for the SET'UN computer," *USSR Comp. Math. and Math. Phys.*, vol. 3, Oxford: Pergamon, 1962, pp. 563–578.
[2] R. D. Merrill, "Ternary logic in digital computers," presented at the 1965 SHARE Design Automation Workshop, Atlantic City, N. J., to be published.
[3] M. H. Lewin, "Retrieval of ordered lists from a content-addressable memory," *RCA Rev.*, vol. 23, pp. 215–229, June 1962.
[4] M. Yoeli and S. Rinen, "Application of ternary algebra to the study of static hazards," *J. ACM*, vol. 11, pp. 84–97, January 1964.
[5] E. B. Eichelberger, "Hazard detection in combinational and sequential switching circuits," *IBM J. Res. and Dev.*, vol. 9, pp. 90–99, March 1965.
[6] T. D. Reader and R. J. Quigley, "Research and development in fluid logic elements," Univac Monthly Progress Repts., nos. 1–8, NAS8-11021, July 1, 1963-February 29, 1964.
[7] V. I. Varshavskii, "Ternary majority logic," *Avtomatika i Telemekhanika*, vol. 25, pp. 673–684, May 1964.
[8] J. Santos and H. Arange, "On the analysis and synthesis of three-valued digital systems," *Proc. 1964 AFIPS Conf.*, vol. 25, pp. 463–475.
[9] W. H. Hanson, "Ternary threshold logic," *IEEE Trans. on Electronic Computers*, vol. EC-12, pp. 191–197, June 1963.
[10] M. Yoeli and G. Rosenfeld, "Logical design of ternary switching circuits," *IEEE Trans. on Electronic Computers*, vol. EC-14, pp. 19–29, February 1965.
[11] E. J. McCluskey, "Minimization of Boolean functions," *Bell Sys. Tech. J.*, pp. 1417–1443, November 1956.
[12] Y. A. Keir, R. D. Merrill, and C. L. Thornton, "Ternary logic," in *Research on Automatic Computer Electronics*, vol. 2, Lockheed Missiles and Space Co., Palo Alto, Calif., RTD-TDR-63-4173, pp. B-151–B-231, October 31, 1963.
[13] R. D. Merrill, Jr., "Some properties of ternary threshold logic," *IEEE Trans. on Electronic Computers* (*Short Notes*), pp. 632–635, October 1964.
[14] J. Santos, H. Arango, and F. Lorenzo, "Threshold synthesis of ternary digital systems," *IEEE Trans. on Electronic Computers* (*Short Notes*), vol. EC-15, pp. 105–107, February 1966.