that either $f_6$ has a five variable alternating 1-tree or $h_5 = g_5$, in which case $f_6$ is independent of $x_1$, a contradiction.

b) Since $f_7$ has a six-variable subfunction, it must have an alternating 1-tree of at least five variables by a). The following three functions have maximum alternating 1-trees of five, six, and seven variables, respectively: $f_7^1 = x_1x_2x_5x_6x_7 + x_3x_4x_5'x_6'x_7'$; $f_7^2 = x_7' \cdot f_6^1$; $f_7^3 = x_1 \cdots x_7$.

*Corollary 3:* For $n \geq 6$, a realization of $f_n$ can be tested for terminal stuck-faults with no more than $2n-4$ input combinations.

*Proof:* If $n=6, f_n$ has an alternating 1-tree of at least five variables, and for $n>6, f_n$ has a subfunction of six variables, from Lemma 2, so that an alternating 1-tree of at least five variables always exists. Therefore, at least five inputs can be tested for stuck faults, using only six input combinations, the remaining $(n-5)$ requiring at most two combinations each for a maximum of $6+2(n-5)=2n-4$.

The problem of determining for $n \geq 8$ how small the maximum alternating 1-tree of some function of $n$ variables can be has proven to be quite difficult. The following theorem establishes that there can be as few as approximately $\frac{1}{2}$ of the variables appearing in any one alternating 1-tree for some functions.

*Theorem 3:* For all $n$ there exist functions $f_n$ whose maximum alternating 1-tree contains $n-[(n-3)/2]$ variables (where $[i]$ denotes the greatest integer $\leq i$, and is 0 for $i<1$).

*Proof:* This has already been proven for $n \leq 7$. For $n \geq 5$ (when the formula gives values less than $n$), the following function satisfies the theorem

$$f_n = x_{k+1} \cdots x_{2k}x_{2k+1} \cdots x_{2k+h} + x_1 \cdots x_k x'_{2k+1} \cdots x'_{2k+h}$$

where for $n$ odd, $h=3$, $k=n-3/2$; for $n$ even, $h=4$, $k=(n-4)/2$.

## V. Conclusions

Although the maximum size alternating 1-tree is useful in establishing an upper bound on the length of a terminal stuck-fault test, it cannot be expected to provide information for a least upper bound in all cases because tests based on alternating $k$-trees are not the most general possible. For the functions exhibited in Theorem 3, a test based on alternating $k$-trees would require $n+2$ input combinations, although a test of length $n-1$ (for $n$ odd) or $n-2$ (for $n$ even) can be constructed by other reasoning. It may well be that $n+1$ is a least upper bound on the test length for arbitrary $f_n$, as has been proven to be the case for $n \leq 5$. This is an open question.

The best upper bound we have been able to establish is $2n-4$ for $n \geq 6$. This is based on the existence of an alternating 1-tree of five variables in every nondegenerate function of six or more variables. If one could establish a better lower bound on the minimum size (over $n$ variable functions) of the maximum alternating 1-tree that appears in $f_n$, this bound could be improved. We believe the minimum size is $n-[(n-3)/2]$, a bound that we have proven for $n \leq 7$. For $n>7$, Theorem 3 only asserts that there exist functions $f_n$ with this as the maximum size alternating 1-tree, but all systematic efforts to find or prove the existence of a nondegenerate $n$ variable function with a smaller maximum alternating 1-tree have failed. If this lower bound could be established, a better upper bound on the test length, $n+[(n-1)/2]$, would follow by the arguments of Section II. However, this must remain a conjecture.

Another problem is to develop an efficient algorithm for finding minimum length terminal stuck-fault tests. The method described following Lemma 1 will yield a test of length $n+1$ rather easily for $n \leq 5$ because a single 1-tree set of $n$ elements is guaranteed to exist (with one exceptional case). For all $n$, one may combine input combinations from 1-tree sets with those in one or more "input-set tests" (see [5]) to find complete tests that may be shorter than $n+1$. The constants required in input-set tests are obtained by using input variables tested by the 1-tree sets.

## References

[1] J. F. Poage, "Derivation of optimal tests to detect faults in combinational circuits," in *Mathematical Theory of Automata.* Brooklyn, N. Y.: Polytechnic Press, 1962.
[2] W. H. Kautz, "Fault testing and diagnosis in combinational digital circuits," *IEEE Trans. Comput.*, vol. C-17, pp. 352–366, Apr. 1968.
[3] R. E. Schwartz, "Existence and uniqueness properties of subfunctions of Boolean functions," *SIAM J. Appl. Math.*, vol. 18, pp. 454–461, Mar. 1970.
[4] M. A. Harrison, *Introduction to Switching and Automata Theory.* New York: McGraw-Hill, 1965.
[5] C. D. Weiss, "Cell input-set systems and their use in fault-detection tests for combinational networks," in review.

# A Decomposition Method of Determining Maximum Compatibles

P. K. SINHA ROY, MEMBER, IEEE, AND C. L. SHENG, SENIOR MEMBER, IEEE

**Abstract—A direct method of determining the maximum compatibility sets of an incompletely specified flow table of a sequential machine is presented. Subsets of pairwise incompatibles are utilized to decompose the set of all states in a step-by-step process into the maximum compatibles in a few steps. The method is simpler and faster than previously reported tabular, algebraic, and graphical techniques.**

**Index Terms—Decomposition tree, flow table, incompletely specified sequential machines, pairwise incompatibles.**

## I. Introduction

The problem of state minimization of incompletely specified sequential machines has been dealt with by a number of workers, the main contributors being Ginsberg [1], Paull and Unger [2], Grasselli and Luccio [3], Meisel [4], and Kella [5]. In most of these works a knowledge of the maximum compatibility sets of states is required to arrive at the minimal machine. Systematic methods are presented by Paull and Unger [2], Marcus [6], Das and Sheng [7], and Choudhury *et al.* [8] for determining the maximum compatibles.

Manuscript received June 5, 1971; revised October 12, 1971. This work was supported in part by the National Research Council of Canada under Grant A-1690.

The authors are with the Department of Electrical Engineering, Faculty of Pure and Applied Science, University of Ottawa, Ottawa, Ont., Canada.

In Paull and Unger [2], the compatibility table showing the pairwise incompatibility relations between states by "crosses" and pairwise compatibility relations by no entries is first formed from the given flow table of the machine. In one method the compatibility table is scanned from the rightmost column and compatible sets are formed and augmented by one state if all the states compatible with the state being scanned occur in a previously formed compatible. The process is continued until all the columns are scanned. The method becomes lengthy as the size of the table increases. In a second method, the incompatibility relations between states are used in successively forming smaller subsets of states that do not contain pairwise incompatibles. This, as well as the previous method, requires $(n-1)$ steps to determine the set of maximum compatibles, where $n$ is the number of states in the machine. Paull and Unger have also suggested a process of decomposition of the set of all states, successively using each of the incompatibility relations for determining the set of maximum compatibles. This would require $k$ steps, $0 \leq k \leq \binom{n}{2}$, to find the solution, where $k$ is the number of pairwise incompatibles and $\binom{n}{2} = {}^nC_2$ is the binomial coefficient. Thus $k$ may greatly exceed $n$. The present method utilizes this idea of decomposing the set of all states successively, but this is done not by individual incompatible state pairs, but by groups of them and, as a result, the decomposition process terminates quickly. In fact, the number of steps required is always less than $k$ whenever there is a single pair of pairwise incompatibles that shares a common state between them, and when none of the pairwise incompatibles has a state in common, the number of steps required is $k$, but then $k \leq n/2$.

The algebraic method of Marcus [6] and Choudhury *et al.* [8] first obtains a sum-of-products expression with each product term corresponding to an incompatible state pair. Considering the states as switching variables, the dual function is obtained. For each of the terms in the dual function the complementary subset with respect to the set of all states is formed. The set of all these subsets corresponds to the maximum compatibles. The process thus involves three operations: forming the sum-of-products expression, finding its dual, and finally complementation of each term, and, consequently, takes more time in determining the maximum compatibles.

In the graphical approach, an incompatibility graph is formed by connecting all incompatible state pairs, each state being represented as a vertex. Each isolated vertex is compatible with all other vertices and appears in all the maximum compatible sets. The maximum compatibles are formed by inspecting the connected portion of the incompatibility graph and partitioning the same successively in disjoint subsets, using the concept of a "modified cut set." The process thus involves a graphical representation and depends much on inspection.

## II. The Decomposition Method

Let $S = \{s_1, s_2, \cdots, s_n\}$, be the set of all the states, and $X = \{x_1, x_2, \cdots, x_u\}$, and $Z = \{z_1, z_2, \cdots, z_m\}$, be the set of input and output alphabets, respectively. The next-state and output mappings $\delta$ and $\omega$, respectively, are given by

$$\delta\colon X \times S \rightarrow S$$

$$\omega\colon X \times S \rightarrow Z.$$

Two states $s_i$ and $s_j$ are said to be compatible ($s_i \sim s_j$) if and only if the following hold.

1) For any input $x_p$ for which both $\omega(x_p, s_i)$ and $\omega(x_p, s_j)$ are specified, $\omega(x_p, s_i) = \omega(x_p, s_j)$.

2) For any input $x_p$ such that both $\delta(x_p, s_i)$ and $\delta(x_p, s_j)$ are specified, $\delta(x_p, s_i) \sim \delta(x_p, s_j)$.

Two states $s_i$ and $s_j$ are incompatible ($s_i \nsim s_j$) if they are not compatible.

A compatibility set $C$ is a set of states such that all the elements of $C$ are pairwise compatible.

A maximum compatibility set $MC$ is a compatibility set $C$ such that it does not form a proper subset of any other compatibility set.

Let a state $s_i$ be incompatible with all the states $s_{j1}, s_{j2}, \cdots, s_{jr}$, forming a subset $S_j$ of $S$, $(S_j \subset S,\ s_i \notin S_j)$. This is represented as $s_i \nsim S_j$. Thus a compatible set of states $C$ can either contain $s_i$ or any subset $\hat{S}_j$ of $S_j (\hat{S}_j \subseteq S_j)$, but not both.

Consider a set of states $\hat{C}$. The relation $s_i \nsim S_j$ decomposes the set $\hat{C}$ according to the following rules.

*Rule 1:* $\hat{C} \rightarrow \hat{C}$ if $s_i \notin \hat{C}$, or $\hat{S}_j \not\subset \hat{C}$, or both.

*Rule 2:*

$$\begin{tikzcd}[column sep=small, row sep=small]
 & & \hat{C}_1 \\
 & s_i \arrow[ur] & \\
\hat{C} \arrow[ur, dash] \arrow[dr, dash] & & \\
 & \hat{S}_j \arrow[dr] & \\
 & & \hat{C}_2
\end{tikzcd}$$

$$\text{if} \quad s_i \in \hat{C}, \quad \text{and} \quad \hat{S}_j \subset \hat{C},$$

where

$$\hat{C}_1 = \hat{C} - \{s_i\} \quad \text{and} \quad \hat{C}_2 = \hat{C} - \hat{S}_j.$$

The subset of states $\hat{C}_1$ and $\hat{C}_2$ will be called residue subsets of $\hat{C}$ with respect to $s_i$ and $\hat{S}_j$, respectively, and will be represented as

$$\hat{C}_1 = R(s_i) \quad \text{and} \quad \hat{C}_2 = R(\hat{S}_j).$$

Decomposition rules 1 and 2 provide us with a simple technique to split a set of states into two residue subsets that do not contain any pair of states $s_i s_k$ such that $s_k \in S_j$ when the decomposing relation is $s_i \nsim S_j$.

If we select the set of all states $S$ as the starting node of a decomposition tree in level 0, the first two residue subsets $R(s_{i1})$ and $R(S_{j1})$ with respect to the first incompatibility relation $(s_i \nsim S_j)_1$ form two nodes in level 1. Now, $R(s_{i1})$ and $R(S_{j1})$ would be maximum compatibles if there were only $r_1$ pairwise incompatibles of the form $s_i s_{j1}, s_i s_{j2}, \cdots, s_i s_{jr_1}$, $r_1$ being the number of states in $S_{j1}$. With the next incompatibility relation $(s_i \nsim S_j)_2$ selected, each of the nodes of the decomposition tree in level 1 branches into residue subsets in level 2. Residue subsets $R_1$ and $R_2$ may appear such that $R_2 \supseteq R_1$. In that case, $R_2$ is retained and $R_1$ is terminated since $R_2$ and $R_1$ are both compatible for the relations tested and $R_2$ is the maximal compatible. All the residue subsets in a level $t$, $1 \le t \le p$, that are not terminated are thus maximum compatibles with respect to the $t$ incompatibility relations tested. The level $p$ corresponds to the end of the decomposition process. An examination of the $k$ pairwise incompatibles will reveal that a number of incompatibility relations of the form $(s_i \not\sim S_j)$ can be formed for every state $s_i \in S$. The total number of such relations is obviously $q \le n$. The inequality $q < n$ holds, if for some $i$, $s_i \not\sim S_j$ and $S_j = \phi$, the null set.

However, each pairwise incompatibility relation is contained in the set of $q$ incompatibility relations twice, and thus not all of these relations are essential and some are redundant.

A method will now be presented for selection of the proper incompatibility relations so that starting from the set of all states we can find the maximum compatibles in fewer number of decompositions.

Let $(s_i \not\sim S_j)_1$ be one of the $q$ incompatibility relations such that $r_1 \ge r_t$, $t \in \{2, \cdots, q\}$. Selecting $(s_i \not\sim S_j)_1$ as the first decomposing relation, let us delete all $s_{i1}$ from the remaining relations that have $s_{i1} \in S_{jt}$, $t \in \{2, 3, \cdots, q\}$. We now get a modified set of $(q-1)$ relations. From this set we choose the next relation using the same principle of selection. Whenever all the states in a particular set $S_{jt}$ of a relation $t$ are deleted, this relation becomes redundant. Finally, we are left with only $p$, $p < q$ relations as the decomposing relations.

The number of elements $r_t$ in the set $S_{jt}$ satisfy

$$r_1 \ge r_2 \ge \cdots \ge r_p \ge 1, \qquad 1 \le t \le p \tag{1}$$

$$\sum_{t=1}^{p} r_t = k. \tag{2}$$

In general, $r_1 > 1$, then $p \le k-1$; also, $p \le n-1$. Therefore, the upper bound of $p$ is min $(k-1, n-1)$. As a special case, when $r_1 = 1$, $p = k \le n/2$.

For obtaining the decomposition relations it is not necessary to write first all the $q$ incompatibility relations. They can be directly obtained by using a table called the distribution table, showing the distribution of states among the pairwise incompatibles. Here all the states of the machine appear as column designators. The pairwise incompatibles are scanned and the number of times each state appears among them is recorded below each state in the distribution table. The state with the highest number of appearances is selected as $s_{i1}$ (if more than one state has the highest appearance, any one of them can be selected). This is indicated by encircling the entry under $s_{i1}$. All the states that are incompatible with $s_{i1}$ are collected as $S_{j1}$. The first decomposition relations between one state and a set of states is thus obtained. A second row of the table is next formed by entering a "0" in the column of $s_{i1}$ and the number of appearances of each $s_j \in S_{j1}$ is reduced by unity. This step corresponds to deleting some states from the incompatibility relations, as discussed earlier. For all other states the entries remain unchanged, as the previous row. This process is continued until we have "0's" for all entries in the last row and a set of relations of the form $(s_i \not\sim S_j)_t$, $t = 1, 2, \cdots, p$. This is the complete set of decomposition relations.

TABLE I

FLOW TABLE OF MACHINE $M$

| $S$ \ $X$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $s_1$ | $s_2$, – | $s_4$, – | — | $s_3$, – |
| $s_2$ | $s_6$, – | $s_9$, – | — | — |
| $s_3$ | — | — | $s_7$, – | $s_8$, – |
| $s_4$ | $s_2$, – | $s_1$, – | $s_6$, – | $s_5$, – |
| $s_5$ | — | — | — | $s_6$, – |
| $s_6$ | $s_1$, 0 | — | $s_2$, – | –, 1 |
| $s_7$ | $s_5$, 1 | $s_2$, – | — | — |
| $s_8$ | $s_5$, – | — | — | $s_1$, 0 |
| $s_9$ | $s_5$, – | $s_3$, – | — | — |

TABLE II

DISTRIBUTION TABLE

| States | | | | | | | | | Decomposition Relations |
|---|---|---|---|---|---|---|---|---|---|
| $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ | $s_8$ | $s_9$ | |
| 3 | 1 | 2 | 3 | 1 | 2 | 2 | 1 | 1 | $s_1 \not\sim \{s_4, s_7, s_9\}$ |
| 0 | 1 | 2 | 2 | 1 | 2 | 1 | 1 | 0 | $s_3 \not\sim \{s_4, s_5\}$ |
| 0 | 1 | 0 | 1 | 0 | 2 | 1 | 1 | 0 | $s_6 \not\sim \{s_7, s_8\}$ |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | $s_2 \not\sim \{s_4\}$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

An algorithm can now be given for determining the maximum compatibles.

*Step 1:* The pairwise incompatibles of a given flow table are formed. To do this the pairwise output incompatibles are first obtained. Each of these pairs is then used to find new pairs that are implied by them. This process is continued until no new pairs are generated [1], [5].

*Step 2:* The distribution table is formed and all the decomposition relations are established.

*Step 3:* The decomposition tree is developed using each relation successively. The final level residue subsets that are not terminated correspond to the desired maximum compatibles (residue subsets that are terminated are underlined).

*Example (from Unger [9, p. 41]):* The incompletely specified machine $M$ is represented by Table I.

*Step 1:* The pairwise incompatibles are

$$\{s_6, s_7\}, \{s_6, s_8\}, \{s_3, s_4\}, \{s_3, s_5\}, \{s_1, s_9\}, \{s_1, s_4\}, \{s_2, s_4\}, \{s_1, s_7\}.$$

*Step 2:* The distribution table and the decomposition relations are given in Table II.

*Step 3:* The decomposition tree is developed as shown in Fig. 1.

The maximum compatibility sets are

$$\{s_4, s_5, s_7, s_8, s_9\}, \{s_2, s_5, s_7, s_8, s_9\}, \{s_4, s_5, s_6, s_9\}, \{s_2, s_5, s_6, s_9\},$$

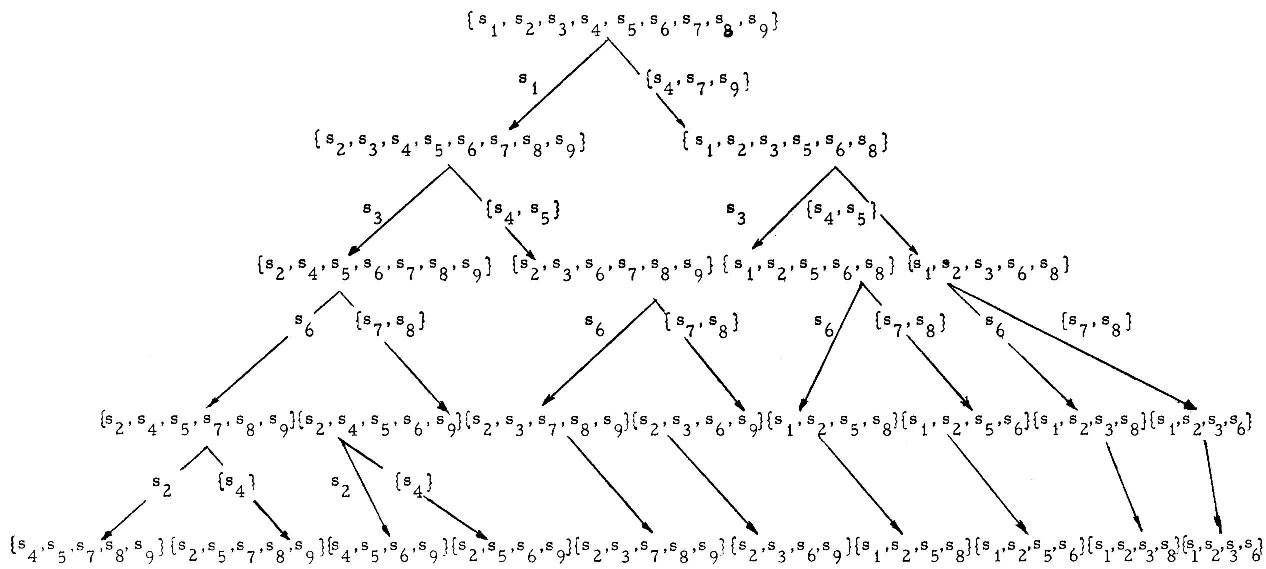


Fig. 1. Decomposition tree of example.

$$\{s_2, s_3, s_7, s_8, s_9\}, \{s_2, s_3, s_6, s_9\}, \{s_1, s_2, s_5, s_8\}, \{s_1, s_2, s_5, s_6\},$$
$$\{s_1, s_2, s_3, s_8\}, \{s_1, s_2, s_3, s_6\}.$$

Here, the number of states $n=9$, the number of pairwise incompatibles is $k=8$, and the number of levels of the decomposition tree is $p=4$.

## III. Comments

If at any stage of forming the decomposing relations the incompatibility relations or successively modified sets of them show the same number of states in $S^{i}_{j1}$ and $S^{i'}_{j1}$, the sets of states incompatible with $s_i$ and $s_{i'}$, respectively, $s_i \neq s_{i'}$, a selection process is involved, and this choice affects the value of $p$, as can be shown using the example solved here. Otherwise, the formation of the decomposing relations does not involve any selection and $p$ is unique. Once the decomposing relations are formed, the order of decomposition does not affect the final result.

## References


[1] S. Ginsburg, "On the reduction of superfluous states in a sequential machine," *J. Ass. Comput. Mach.*, vol. 6, pp. 252–282, 1959.

[2] M. C. Paull and S. H. Unger, "Minimizing the number of states in incompletely specified sequential switching functions," *IRE Trans. Electron. Comput.*, vol. EC-8, pp. 356–367, Sept. 1959.

[3] A. Grasselli and F. Luccio, "A method for minimizing the number of internal states in incompletely specified sequential networks," *IEEE Trans. Electron. Comput.*, vol. EC-14, pp. 350–359, June 1965.

[4] W. S. Meisel, "A note on internal state minimization in incompletely specified sequential networks," *IEEE Trans. Electron. Comput.* (Short Notes), vol. EC-16, pp. 508–509, Aug. 1967.

[5] J. Kella, "State minimization of incompletely specified sequential machines," *IEEE Trans. Comput.*, vol. C-19, pp. 342–348, Apr. 1970.

[6] M. P. Marcus, "Derivation of maximal compatibles using Boolean algebra," *IBM J. Res. Develop.*, vol. 8, pp. 537–538, Nov. 1964.

[7] S. R. Das and C. L. Sheng, "On finding maximum compatibles," *Proc. IEEE* (Lett.), vol. 57, pp. 694–695, Apr. 1969.

[8] A. K. Choudhury, A. K. Basu, and S. C. DeSarkar, "On the determination of the maximum compatibility classes," *IEEE Trans. Comput.* (Corresp.), vol. C-18, p. 665, July 1969.

[9] S. H. Unger, *Asynchronous Sequential Switching Circuits.* New York: Wiley, 1969, p. 41.


# On Multivalued Symmetric Functions

SAMUEL C. LEE, MEMBER, IEEE, AND EDWARD T. LEE, MEMBER, IEEE,

**Abstract—This note describes an algorithm for identifying multivalued symmetric switching functions using parallel processing. Some general properties of multivalued symmetric functions have been investigated. The mixed multivalued symmetric switching function is defined and an algorithm for identifying it is also presented.**

**Index Terms—Function identification, mixed symmetric functions, multivalued logic, switching functions, symmetric functions.**

## I. Introduction

The binary number system has been used throughout the entire development of computer technology. The growth of computation systems and the need to process increasing volumes of data faster has resulted in the development of large-scale integrated logic circuitry. However, the volume of data continues to increase while the circuit components and memory devices approach their practical limit in size and speed.

The design of computation systems in other number systems seems to be a logical solution to the continued increase in volume of data to be processed by digital computers. Interest in the ternary computer has been evidenced in the past decade by an increasing number of papers on various phases of three-valued algebras and circuits. The Soviet Union has been operating a small experimental ternary computer since 1959 [1]. The staff of the Computation Laboratory of Harvard University has advocated the use of the ternary system in preference to the binary system as early as 1951 [2]. Recently there has been a growing interest in the ternary number system as a means of processing greater volumes of data per unit of time, and a number of authors have been

Manuscript received March 29, 1970; revised February 12, 1971, and June 16, 1971.

S. C. Lee is with the Department of Electrical Engineering, University of Houston, Houston, Tex.

E. T. Lee is with Systems Control, Inc., Palo Alto, Calif.