2-1977

DUAL-MODE COMBINATIONAL LOGIC FOR FUNCTION-INDEPENDENT FAULT TESTING

Sumit DasGupta
Syracuse University

2-77

DUAL-MODE COMBINATIONAL LOGIC FOR FUNCTION-INDEPENDENT FAULT TESTING

Sumit DasGupta

Carlos R. P. Hartmann

Luther D. Rudolph

February 1977

SCHOOL OF COMPUTER
AND INFORMATION SCIENCE
SYRACUSE UNIVERSITY

DUAL-MODE COMBINATIONAL LOGIC FOR FUNCTION-INDEPENDENT FAULT TESTING

Sumit DasGupta

Carlos R. P. Hartmann

Luther D. Rudolph

School of Computer and Information Science

Syracuse University

Syracuse, New York 13210

Tele. (315)423-2368

This work was supported in part by the Air Force Systems Command's Rome Air Development Center, Griffiss, AFB, New York and by IBM under the Resident Study Program.

ABSTRACT

This paper presents a method of using hardware redundancy to ease the problem of fault testing in combinational logic networks. Combinational logic networks are constructed using dual-mode logic gates. Initially, it is shown that these networks can be tested for all single stuck-at-faults using just two function-independent tests. This method is then extended to detect a large class of multiple faults with the same two function-independent tests.

I. INTRODUCTION

Traditionally, logic circuits have been designed with little regard for the problem of fault testing. Designers have been concerned with minimizing the complexity of the network and this was certainly understandable in the period prior to the development of LSI. When a logic network is small, there are a number of standard techniques for deriving tests to detect faults [1,2]. As the size of the network grows, however, the number of test required may increase very rapidly. More importantly, the complexity of finding the test set may become prohibitive. As logic component costs decrease and the difficulty of test generation increases, a point is reached where logic designers should begin to consider the possibility of introducing hardware redundancy to simplify testing.

In this paper, we present one such approach based on the concept of dual-mode logic. The basic idea is to use redundancy to (1) reduce the number of required tests, and (2) reduce the complexity of deriving the tests. From a testing standpoint, the ultimate solution would be a method of designing logic networks that could be tested using a function-independent test set of minimum possible size, i.e., a test set that is independent of the logic function and internal layout of the network. Although this may appear to be an unrealistic goal, we will show that such a solution is possible not only for the problem of detecting all single stuck-at-faults (s-a-faults) but also for the problem of detecting large classes of multiple faults in any combinational network. This is achieved through the use of a family of dual-mode gates which perform the required logic functions in one of the modes while it is tested in the other mode.

In section II we develop concurrently the theory of the dual-mode gates and the combinational network built with them. We initially consider the problem of detecting all single s-a-faults with a function-independent test set and later extend it to detecting a larger class of s-a-faults. Section III contains the conclusions and discussion of the results.

II. THE DUAL-MODE COMBINATIONAL LOGIC NETWORK

In this section we will develop the dual-mode combinational (DMC) network. We will require that this network have the following properties:

Property 1: Any DMC network can be tested for all single s-a-faults with the minimum number of tests.

Property 2: The tests are function-independent, i.e., independent of the logic function and internal layout of the network.

Since there are two types of faults, i.e., stuck-at-1 (s-a-1) and stuck-at-0 (s-a-0), a minimum of two tests are required to test any network. Therefore, we will require that the DMC network be tested for all single s-a-faults with two tests. This requires that each test must test all the gates of the network simultaneously for exactly half of the faults associated with each gate.

In a general combinational network containing reconvergent fan-out a single fault may affect all the data inputs of some gate. This phenomenon and the constraint that the DMC network be tested for all single s-a-faults with function-independent tests dictates that the logic gate with which the DMC network is built must be tested for all possible data input faults with two tests. Obviously, the existing logic gates, i.e., AND, OR, NAND, NOR and Exclusive-OR, do not satisfy the above requirement. Hence, we need to define a new gate. This gate must also perform the necessary logic functions so that any general network can be built with it. To define this new gate we need the following lemma:

Lemma 1: The necessary and sufficient conditions for the existence of an n-input, 1-output logic gate which can be tested for all patterns of p or fewer s-a-faults with two tests are:

(i) there exists at least one n-input vector $\underset{\sim}{S}$ for which the output is the complement of the output for the complement input $\bar{\underset{\sim}{S}}$,

(ii) the output for any input n-tuple which is Hamming distance [3] one through p away from $\underset{\sim}{S}$ is the complement of the output for $\underset{\sim}{S}$,

(iii) the output for any input n-tuple which is Hamming distance one through p away from $\bar{\underset{\sim}{S}}$ is the complement of the output for $\bar{\underset{\sim}{S}}$.

Then, the set $\{\underset{\sim}{S},\bar{\underset{\sim}{S}}\}$ detects all patterns of p or fewer s-a-faults in the inputs and output of the gate.

Proof: This follows immediately from the fact that if any pattern of p or fewer faults occurs in the input of the gate, then application of $\underset{\sim}{S}$ or $\bar{\underset{\sim}{S}}$ will have the effect of applying an input n-tuple Hamming distance one through p away from $\underset{\sim}{S}$ or $\bar{\underset{\sim}{S}}$. Thus, the output of the faulty gate will be the complement of the expected output. Since the output for $\underset{\sim}{S}$ is different from that for $\bar{\underset{\sim}{S}}$, a fault in the output will also be detected. Conversely, conditions (i), (ii) and (iii) must hold in order that $\underset{\sim}{S}$ and $\bar{\underset{\sim}{S}}$ form a complete test set for all patterns of p or fewer s-a-faults in the inputs and output of the gate.

Q.E.D.

Observe that a gate that satisfies Lemma 1 must have at least 2p+1 data inputs [3]. In the following theorem we give the requirement for the existence of a gate in which all patterns of faults in the data inputs or output of the gate is detected by only two tests.

<u>Theorem 1</u>: Any gate must have at least one extra "control" input to enable all patterns of faults in the data inputs and output to be detected by two tests.

<u>Proof</u>: Consider any gate with n data inputs only. It cannot satisfy statements (ii) and (iii) of Lemma 1 simultaneously when $p = n$.

Q.E.D.

Theorem 1 essentially implies that it is not enough to have only data inputs in this new logic gate. Rather, it is necessary to expand the truth table of this logic gate using at least one extra "control" input. To guarantee that not all inputs of the gate are affected by a single fault in a network built with this gate, data and control inputs of a gate cannot be intermixed or fed from a common source. Furthermore, since a DMC network must have control inputs, to minimize the number of extra "control" inputs in the DMC network we impose the following property:

<u>Property 3</u>: All identical control inputs of gates in a DMC network must be driven from a common network input.

In accordance with the requirements of Lemma 1, the truth table of this logic gate will be as shown in Table I, where $\{\underset{\sim}{S},\bar{\underset{\sim}{S}}\}$ is one of the possible test sets which detect all patterns of faults in the data inputs, $d_1,d_2,\ldots,d_n$. However, a fault in the control input of this gate will not be detected by the test set $\{\underset{\sim}{S},\bar{\underset{\sim}{S}}\}$ since the output for $\underset{\sim}{S}$ or $\bar{\underset{\sim}{S}}$ is unchanged when a fault occurs in the control input, k_1. Hence, the gate described by Table I cannot be used to build a DMC network.

TABLE I

k_1	d_1	...	d_{n-1}	d_n	$f(k_1,d_1,\ldots,d_n)$
0	0	...	0	0	
0	0	...	0	1	$\bar{b}$
0	0	...	1	0	
		⋮			
		$\underset{\approx}{S}$			b
		⋮			$\bar{b}$
0	1	...	1	1	
1	0	...	0	0	
1	0	...	0	1	b
1	0	...	1	0	
		⋮			
		$\bar{\underset{\approx}{S}}$			$\bar{b}$
		⋮			b
1	1	...	1	1	

So, Table I has to be expanded by introducing a second control input, k_2, as shown in Table II, where $\underset{\approx}{S} = (a_1,a_2,c_1,\ldots,c_n)$ and $\bar{\underset{\approx}{S}} = (\bar{a}_1,\bar{a}_2,\bar{c}_1,\ldots,\bar{c}_n)$ are the two tests for the logic gate. It is

TABLE II

k_1	k_2	d_1	...	d_{n-1}	d_n	f
a_1	a_2	0	...	0	0	
		0	...	0	1	
		0	...	1	0	$\bar{b}$
			⋮			
a_1	a_2	c_1	...	c_{n-1}	c_n	b
			⋮			$\bar{b}$
a_1	a_2	1	...	1	1	
a_1	$\bar{a}_2$	0	...	0	0	
		0	...	0	1	
		0	...	1	0	
			⋮			
a_1	$\bar{a}_2$	c_1	...	c_{n-1}	c_n	$\bar{b}$
			⋮			
a_1	$\bar{a}_2$	$\bar{c}_1$	...	$\bar{c}_{n-1}$	$\bar{c}_n$	b
			⋮			
a_1	$\bar{a}_2$	1	...	1	1	
$\bar{a}_1$	a_2	0	...	0	0	
		0	...	0	1	
		0	...	1	0	
			⋮			
$\bar{a}_1$	a_2	c_1	...	c_{n-1}	c_n	$\bar{b}$
			⋮			
$\bar{a}_1$	a_2	$\bar{c}_1$	...	$\bar{c}_{n-1}$	$\bar{c}_n$	b
			⋮			
$\bar{a}_1$	a_2	1	...	1	1	
$\bar{a}_1$	$\bar{a}_2$	0	...	0	0	
		0	...	0	1	b
		0	...	1	0	
			⋮			
$\bar{a}_1$	$\bar{a}_2$	$\bar{c}_1$	...	$\bar{c}_{n-1}$	$\bar{c}_n$	$\bar{b}$
			⋮			
$\bar{a}_1$	$\bar{a}_2$	1	...	1	1	b

easily seen that $\{\underset{\sim}{S},\underset{\sim}{\bar{S}}\}$ detects all possible patterns of faults in the data inputs, $d_1,\ldots,d_n$ and output of the gate defined by Table II, assuming that the controls, k_1 and k_2 are fault-free. Moreover, it detects single faults in the control inputs assuming that the data inputs are fault-free. The above results can be formalized in the following theorem:

Theorem 2: Two control inputs are necessary and sufficient for any gate such that:

(i) all patterns of faults in the data inputs and output of the gate can be detected with two tests assuming that the control inputs are fault-free and

(ii) all single faults in the control inputs can also be detected by the same two tests assuming that all the data inputs are fault free.

The general class of gates, defined by Table II, will henceforth be referred to as a 2T(2) gate where the two in the parentheses refers to the number of control inputs in the gate. We will now prove some basic properties of the 2T(2) gate to demonstrate that it is not very suitable for building logic networks.

Lemma 2: In the truth table shown in Table II, there exists at most one test pattern per segment, as defined by k_1 and k_2 for gates with 2 or more data inputs.

Proof: Without loss of generality, consider the segment $(k_1,k_2) = (a_1,a_2)$ in which the test is $(a_1,a_2,c_1,\ldots,c_n) = \underset{\sim}{V}_1$ with expected output b. We will prove by contradiction that it is the only test in the segment. To do so, assume there exists yet another test $(a_1,a_2,\hat{c}_1,\dots,\hat{c}_n) = \underset{\sim}{V}_2$ in the same segment. We thus have to consider 2 cases:

(i) $\underset{\sim}{V}_2$ has output b

(ii) $\underset{\sim}{V}_2$ has output $\bar{b}$.

In case (i), $\underset{\sim}{V}_2$ having output b contradicts with the requirement that all patterns Hamming distance one through n from $\underset{\sim}{V}_1$ and in the same segment of the truth table as $\underset{\sim}{V}_1$ must have output $\bar{b}$. In case (ii), since there are at least 2 data inputs, there exists patterns other than $\underset{\sim}{V}_1$ and $\underset{\sim}{V}_2$ in the same segment that are required to have output $\bar{b}$ because of $\underset{\sim}{V}_1$ and output b because of $\underset{\sim}{V}_2$. Hence a contradiction.

Q.E.D.

Lemma 2 implies that a 2T(2) gate has at most four tests.

<u>Lemma 3</u>: The knowledge of any of the four tests for a 2T(2) gate determines the other three.

<u>Proof</u>: Let $(a_1,a_2,c_1,\dots,c_n)$ with output b be one of the tests. By Lemma 1, $(\bar{a}_1,\bar{a}_2,\bar{c}_1,\dots,\bar{c}_n)$ with output $\bar{b}$ is the other member of the same test set. To detect single s-a-faults in the control inputs, $(a_1,\bar{a}_2,c_1,\dots,c_n)$ and $(\bar{a}_1,a_2,c_1,\dots,c_n)$ must have output $\bar{b}$ while $(a_1,\bar{a}_2,\bar{c}_1,\dots,\bar{c}_n)$ and $(\bar{a}_1,a_2,\bar{c}_1,\dots,\bar{c}_n)$ must have output b. We now show by contradiction that the two remaining tests are either $\{(a_1,\bar{a}_2,c_1,\dots,c_n),\ (\bar{a}_1,a_2,\bar{c}_1,\dots,\bar{c}_n)\}$ or $\{(\bar{a}_1,a_2,c_1,\dots,c_n),$ $(a_1,\bar{a}_2,\bar{c}_1,\dots,\bar{c}_n)\}$. Suppose that one of the remaining tests is $(a_1,\bar{a}_2,e_1,\dots,e_n) = \underset{\sim}{V}_1$. If $\underset{\sim}{V}_1$ has output b, then considering only the data inputs, $(a_1,\bar{a}_2,\bar{c}_1,\dots,\bar{c}_n)$ having output b contradicts

Lemma 1 for p = n. Similarly, for the case when $\underset{\sim}{V}_1$ has output $\bar{b}$.

Q.E.D.

Lemma 3 implies that a 2T(2) gate has at most two test sets.

We now show that if a network is constructed with 2T(2) gates, then it is not always possible to detect all classes of single s-a-faults.

Theorem 3: In a network constructed with 2T(2) gates any single s-a-fault that affects a control input of more than one gate is not always detectable by any function-independent test set of minimum size.

Proof: Consider the network of Fig. 1 where both gates

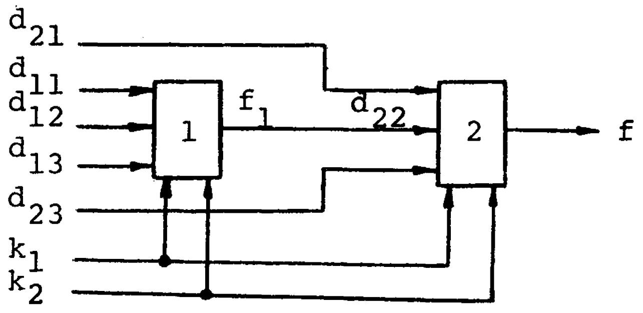

Fig. 1 Network with undetectable control faults.

have common controls. Using Lemma 3, let the two test sets for Gate 1 be $\{\underset{\sim}{S}_{11},\bar{\underset{\sim}{S}}_{11}\}$ and $\{\underset{\sim}{S}_{12},\bar{\underset{\sim}{S}}_{12}\}$, where

$$\underset{\sim}{S}_{11} = (k_1,k_2,d_{11},d_{12},d_{13}) = (a_1,a_2,c_{11},c_{12},c_{13})$$

with output b_1, and

$$\underset{\sim}{S}_{12} = (k_1,k_2,d_{11},d_{12},d_{13}) = (a_1,\bar{a}_2,c_{11},c_{12},c_{13})$$

with output $\bar{b}_1$. Similarly, let the two test sets for gate 2 be $\{\underset{\sim}{S}_{21}, \bar{\underset{\sim}{S}}_{21}\}$ and $\{\underset{\sim}{S}_{22}, \bar{\underset{\sim}{S}}_{22}\}$, where

$$\underset{\sim}{S}_{21} = (k_1, k_2, d_{21}, d_{22}, d_{23}) = (a_1, a_2, c_{21}, c_{22}, c_{23})$$

with output b_2, and

$$\underset{\sim}{S}_{22} = (k_1, k_2, d_{21}, d_{22}, d_{23}) = (a_1, \bar{a}_2, c_{21}, c_{22}, c_{23})$$

with output $\bar{b}_2$.

Since this network must be tested for all single s-a-faults with two tests, both gates must be tested simultaneously by each test. Suppose a single s-a-fault occurs in k_2 which affects both gates and one of the tests that can be applied is $\underset{\sim}{S}_{11}$ in gate 1 and $\underset{\sim}{S}_{21}$ in gate 2, i.e., the pattern applied to the network is $(k_1, k_2, d_{11}, d_{12}, d_{13}, d_{21}, d_{23}) = (a_1, a_2, c_{11}, c_{12}, c_{13}, c_{21}, c_{23})$. This implies that the expected output b_1 of gate 1 equals the required logic value c_{22} in input d_{22} of gate 2. Then, expected network output is b_2. The fault in k_2 makes the output of gate 1 $\bar{b}_1$. Thus, the pattern applied to gate 2 is $(a_1, \bar{a}_2, c_{21}, \bar{c}_{22}, c_{23})$. Since, this pattern is Hamming distance one way from S_{22}, by Lemma 1, the output of the network would still be b_2. Hence, this fault will not be detected. Similarly, we can show that this fault is undetectable even when the alternative set of tests, i.e., $\bar{\underset{\sim}{S}}_{12}$ for gate 1 and $\bar{\underset{\sim}{S}}_{22}$ for gate 2, is used. Lemma 3 shows that there exists a second choice for test sets $\{\underset{\sim}{S}_{12}, \bar{\underset{\sim}{S}}_{12}\}$ and $\{\underset{\sim}{S}_{22}, \bar{\underset{\sim}{S}}_{22}\}$. However, by a similar argument it can be shown that this second choice of tests will not detect at least a single s-a-fault in k_1 if it affects both gates.

Q.E.D.

Evidently, a single fault in a control that affects two or more gates can always be detected if the output of the gate in the presence of the above fault is the complement of the expected output, independent of the data inputs. Thus, in relation to test $\underset{\sim}{S}_1 = (2_1, a_2, c_1, \ldots, c_n)$ for the 2T(2) gate in Table II, a single s-a-$\bar{a}_1$ fault in k_1 or a single s-a-$\bar{a}_2$ fault in k_2 that affects more than one gate in a network would be detected if for all rows in segments of the truth table in Table II defined by $(k_1,k_2) = (\bar{a}_1, a_2)$ and $(k_1,k_2) = (a_1,\bar{a}_2)$, the output of the 2T(2) gate is $\bar{b}$. But a similar condition arising out of a single s-a-a_1 fault in k_1 or a single s-a-a_2 fault in k_2 requires all rows in segments $(\bar{a}_1, a_2)$ and $(a_1, \bar{a}_2)$ to have output b so that these faults can be detected by the test $\bar{\underset{\sim}{S}}_1 = (\bar{a}_1, \bar{a}_2, c_1, \ldots, c_n)$. This conflict in requirement not only shows that it is physically impossible to detect all single control input faults that affect more than one gate in any general network but more importantly, it shows that the test requirements for any pair of tests for a 2T(2) gate tend to encroach on each other, i.e., there is not enough Hamming distance between the two tests.

Consider then the gate with three controls, k_1, k_2 and k_3. Suppose $(k_1,k_2,k_3,d_1,\ldots,d_n) = (a_1,a_2,a_3,c_1,\ldots,c_n) = \underset{\sim}{S}$ with output b is one test and $(k_1,k_2,k_3,d_1,\ldots,d_n) = (\bar{a}_1,\bar{a}_2,\bar{a}_3,\bar{c}_1,\ldots,\bar{c}_n) = \bar{\underset{\sim}{S}}$ with output $\bar{b}$ is the other test from the same test set. Then, if all rows of segments $(\bar{a}_1,a_2,a_3)$, $(a_1,\bar{a}_2,a_3)$ and $(a_1,a_2,\bar{a}_3)$ have output $\bar{b}$ and if all rows of segments $(a_1,\bar{a}_2,\bar{a}_3)$, $(\bar{a}_1,a_2,\bar{a}_3)$ and $(\bar{a}_1,\bar{a}_2,a_3)$ have output b, as shown in Table III, then all single faults in any control which affect one or more gates in a network built with 2T(3) gates will be detected. Furthermore, by Lemma 1, all patterns of data input and output faults in the gate specified by Table III can be detected with the test set $\{\underset{\sim}{S}, \bar{\underset{\sim}{S}}\}$.

TABLE III

k_1	k_2	k_3	d_1	...	d_n		f
a_1	a_2	a_3	0	...	0	}	
				⋮			$\bar{b}$
a_1	a_2	a_3	c_1	...	c_n		b
				⋮		}	$\bar{b}$
a_1	a_2	a_3	1	...	1		
a_1	a_2	$\bar{a}_3$	0	...	0		$\bar{b}$
				⋮			⋮
a_1	a_2	$\bar{a}_3$	1	...	1		$\bar{b}$
a_1	$\bar{a}_2$	a_3	0	...	0		$\bar{b}$
				⋮			⋮
a_1	$\bar{a}_2$	a_3	1	...	1		$\bar{b}$
$\bar{a}_1$	a_2	a_3	0	...	0		$\bar{b}$
				⋮			⋮
$\bar{a}_1$	a_2	a_3	1	...	1		$\bar{b}$
a_1	$\bar{a}_2$	$\bar{a}_3$	0	...	0		b
				⋮			⋮
a_1	$\bar{a}_2$	$\bar{a}_3$	1	...	1		b
$\bar{a}_1$	a_2	$\bar{a}_3$	0	...	0		b
				⋮			⋮
$\bar{a}_1$	a_2	$\bar{a}_3$	1	...	1		b
$\bar{a}_1$	$\bar{a}_2$	a_3	0	...	0		b
				⋮			⋮
$\bar{a}_1$	$\bar{a}_2$	a_3	1	...	1		b
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	0	...	0	}	
				⋮			b
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{c}_1$	...	$\bar{c}_n$		$\bar{b}$
				⋮		}	b
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	1		1		

We will show that in order to test with two function-independent tests for all single s-a-faults in any network built with 2T(3) gates, the tests $\underset{\sim}{S}$ and $\underset{\sim}{\bar{S}}$ must be completely specified in the data inputs and outputs.

Theorem 4: For any network built with 2T(3) gates to be tested for all single s-a-faults with two function-independent tests, one test for the 2T(3) gate must have $(d_1,\ldots,d_n) = (0,\ldots,0)$ with output 0 and the other $(d_1,\ldots,d_n) = (1,\ldots,1)$ with output 1.

Proof: This proof uses the fact that since a gate is the most elementary network, function-independent testing implies that all gates with the same number of inputs must have the same test sets. Now, consider the network of Fig. 2 built with 2T(3) gates.

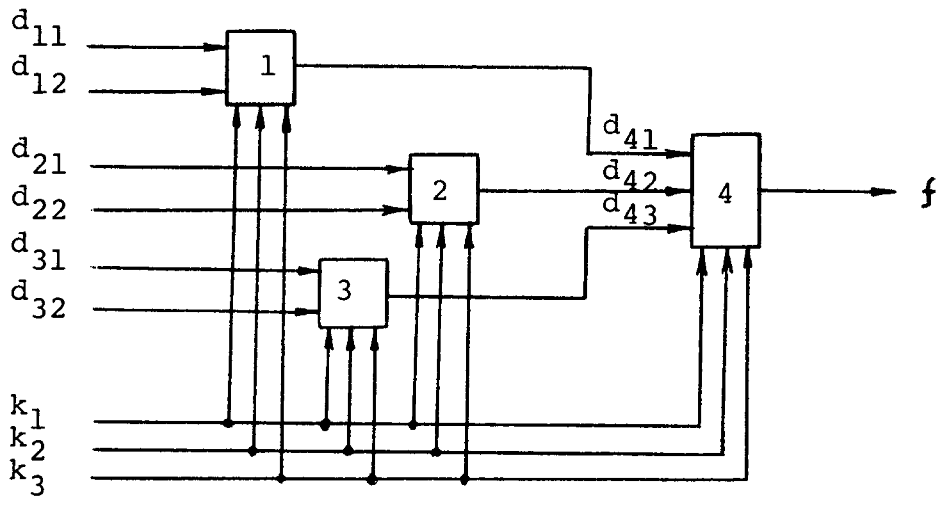

Fig. 2 Network 1.

Suppose that one of the tests for gate 1 is

$\underset{\sim}{S}_1 = (k_1,k_2,k_3,d_{11},d_{12}) = (a_1,a_2,a_3,0,1)$. Then since all the gates have common controls and have to be tested simulataneously, the corresponding required tests for gates 2 and 3 are, respectively:

$$\underset{\sim}{S}_2 = (k_1,k_2,k_3,d_{21},d_{22}) = (a_1,a_2,a_3,0,1)$$

and

$$\underset{\sim}{S}_3 = (k_1,k_2,k_3,d_{31},d_{32}) = (a_1,a_2,a_3,0,1) .$$

Thus, the test applied to network 1 is,

$$\underset{\sim}{T} = (k_1,k_2,k_3,d_{11},d_{12},d_{21},d_{22},d_{31},d_{32}) = (a_1,a_2,a_3,0,1,0,1,0,1)$$

Since this test must be function-independent, it must be able to test the network of Fig. 3. But since $\underset{\sim}{T}$ has an odd number of 0's and 1's, gates 1 and 2 of Fig. 3 cannot have identical patterns applied to their respective inputs. But since both gates have common controls, both must have the same pattern applied to their inputs

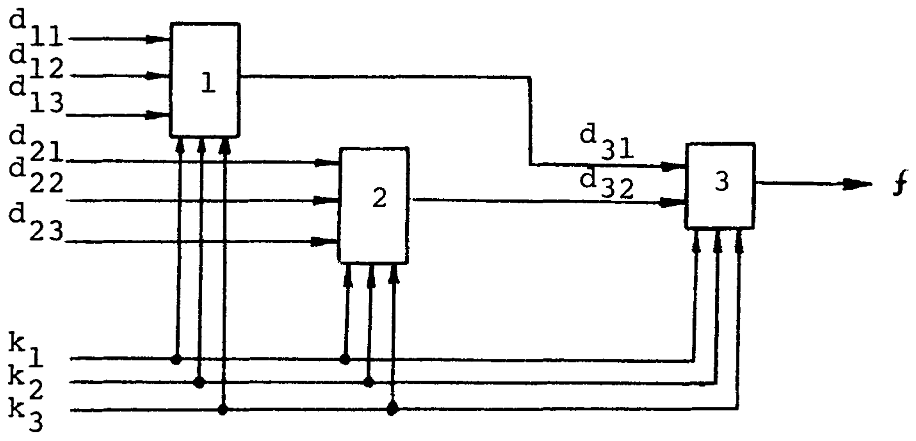

Fig. 3 Network 2.

to be tested simultaneously. Hence, $\underset{\sim}{T}$ cannot test both gates 1 and 2 together thus contradicting the assumption that the network can be tested with two tests. Hence, the tests for the 2T(3) gates must have $(d_1,\ldots,d_n) = (0,\ldots,0)$ or $(d_1,\ldots,d_n) = (1,\ldots,1)$.

In order to show that the test in which $(d_1,\ldots,d_n) = (0,\ldots,0)$ must have output 0 and the test in which $(d_1,\ldots,d_n) = (1,\ldots,1)$ must have output 1 we have to show that for any network constructed with 2T(3) gates, the data inputs have to be either all -0's or all -1's during a test. Consider the network of Fig. 2 again. Suppose one of the tests for network 3 is

$(k_1,k_2,k_3,d_{11},d_{12},d_{21},d_{22},d_{31},d_{32}) = (a_1,a_2,a_3,0,0,1,1,0,0)$.

Then, since this test must be function-independent, it must also test network 2 of Fig. 3. However that implies that either gate 1 or gate 2 of network 2 would have a pattern of 0's and 1's which contradicts the first part of this theorem. Finally, consider network 3 of Fig. 4. Suppose $(k_1,k_2,k_3,d_{11},d_{12},d_{22}) = (a_1,a_2,a_3,0,0,0)$ is one of the tests. Then, this test must test both gates

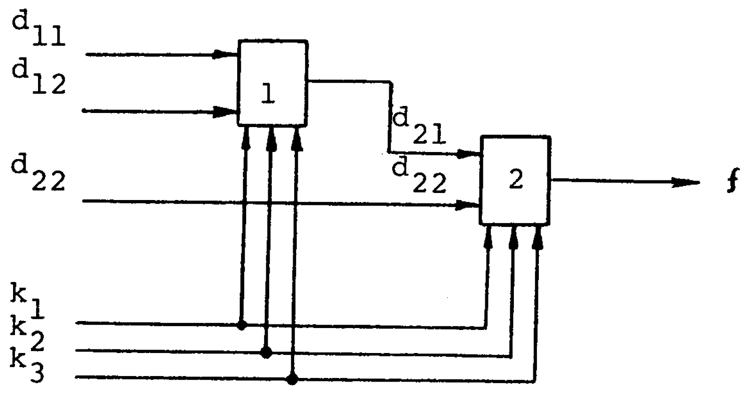

Fig. 4 Network 3

simultaneously but if gate 1 of network 3 has output 1, then gate 2 will not be tested. Hence, this network cannot be tested with two tests which is a contradiction.

Q.E.D.

As a consequence of the above theorem, the truth table for a 2T(3) gate is converted to the form in Table IV. Evidently, there are only two segments, i.e., $(k_1,k_2,k_3) = (a_1,a_2,a_3)$ which is an

TABLE IV

k_1	k_2	k_3	d_1	...	d_{n-1}	d_n	f
a_1	a_2	a_3	0	...	0	0	0
a_1	a_2	a_3	0	...	0	1	1
				⋮			⋮
a_1	a_2	a_3	1	...	1	1	1
a_1	a_2	$\bar{a}_3$	0	...	0	0	1
				⋮			⋮
a_1	a_2	$\bar{a}_3$	1	...	1	1	1
a_1	$\bar{a}_2$	a_3	0	...	0	0	1
				⋮			⋮
a_1	$\bar{a}_2$	a_3	1	...	1	1	1
$\bar{a}_1$	a_2	a_3	0	...	0	0	1
				⋮			⋮
$\bar{a}_1$	a_2	a_3	1	...	1	1	1
a_1	$\bar{a}_2$	$\bar{a}_3$	0	...	0	0	0
				⋮			⋮
a_1	$\bar{a}_2$	$\bar{a}_3$	1	...	1	1	0
$\bar{a}_1$	a_2	$\bar{a}_3$	0	...	0	0	0
				⋮			⋮
$\bar{a}_1$	a_2	$\bar{a}_3$	1	...	1	1	0
$\bar{a}_1$	$\bar{a}_2$	a_3	0	...	0	0	0
				⋮			⋮
$\bar{a}_1$	$\bar{a}_2$	a_3	1	...	1	1	0
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	0	...	0	0	0
				⋮			⋮
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	1	...	1	0	0
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	1	...	1	1	1

OR function and $(k_1,k_2,k_3) = (\bar{a}_1,\bar{a}_2,\bar{a}_3)$ which is an AND function, that can be used to build some network. But since neither is a functionally complete set, the 2T(3) gate cannot be used to build any general network. Hence, we expand Table IV using another control k_4 such that the resulting truth table, shown in Table V, contains all the features of Table IV and has one or more segments to accommodate different logic functions. Note that Theorem 4 can be extended to show that the test patterns needed to test a 2T(4) gate, as described in Table V, must be $(k_1,k_2,k_3,k_4,d_1,\ldots,d_4) = (a_1,a_2,a_3,a_4,0,\ldots,0)$ with output 0 and $(k_1,k_2,k_3,k_4,d_1,\ldots,d_n) = (\bar{a}_1,\bar{a}_2,\bar{a}_3,\bar{a}_4,1,\ldots,1)$ with output 1. Table V shows that the 2T(4) gate has two modes of operation, i.e., test mode for testing and normal mode for some logic function.

Evidently, any network built with 2T(4) gates will also have two modes of operation. Henceforth, these networks will be referred as DMC networks. Table V represents a family of logic gates in normal mode and hence any network can be converted to a DMC network by a one-to-one replacement of a conventional logic gate with an equivalent 2T(4) gate such that all gates in the network can share the same control inputs. Since all the gates in a DMC network must share the same controls and all 2T(4) gates must be tested simultaneously, then the tests for an m-data input DMC network must be:

$$\underset{\sim}{T}_0 = (k_1,k_2,k_3,k_4,d_1,\ldots,d_m) = (a_1,a_2,a_3,a_4,0,\ldots,0)$$

and

$$\bar{\underset{\sim}{T}}_0 = (k_1,k_2,k_3,k_4,d_1,\ldots,d_m) = (\bar{a}_1,\bar{a}_2,\bar{a}_3,\bar{a}_4,1,\ldots,1) .$$

TABLE V

k_1	k_2	k_3	k_4	d_1	...	d_{n-1}	d_n	f
a_1	a_2	a_3	a_4	0	...	0	0	0
a_1	a_2	a_3	a_4	0	...	0	1	1
					$\vdots$			
a_1	a_2	a_3	a_4	1	...	1	1	1
a_1	a_2	a_3	$\bar{a}_4$	x	...	x	x	1
a_1	a_2	$\bar{a}_3$	a_4	x	...	x	x	1
a_1	$\bar{a}_2$	a_3	a_4	x	...	x	x	1
$\bar{a}_1$	a_2	a_3	a_4	x	...	x	x	1
a_1	a_2	$\bar{a}_3$	$\bar{a}_4$					
a_1	$\bar{a}_2$	a_3	$\bar{a}_4$					
$\bar{a}_1$	a_2	a_3	$\bar{a}_4$		One or More			
a_1	$\bar{a}_2$	$\bar{a}_3$	a_4		Logic			
$\bar{a}_1$	a_2	$\bar{a}_3$	a_4		Functions			
$\bar{a}_1$	$\bar{a}_2$	a_3	a_4					
a_1	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	x	...	x	x	0
$\bar{a}_1$	a_2	$\bar{a}_3$	$\bar{a}_4$	x	...	x	x	0
$\bar{a}_1$	$\bar{a}_2$	a_3	$\bar{a}_4$	x	...	x	x	0
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	a_4	x	...	x	x	0
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	0	...	0	0	0
					$\vdots$			
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	1	...	1	0	0
$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	1	...	1	1	1

x $\equiv$ Don't Care.

Note that for a fault-free DMC network the expected logic value at any network output when $\underset{\sim}{T}_0$ is applied is 0 and 1 when $\underset{\sim}{\bar{T}}_0$ is applied.

<u>Theorem 5</u>: In any DMC network built with 2T(4) gates, the test set $\{\underset{\sim}{T}_0, \underset{\sim}{\bar{T}}_0\}$ will detect any pattern of s-a-faults provided that faults in the control lines affect at most one control input of any 2T(4) gate.

<u>Proof</u>: Any fault in a network output is detectable by the test set $\{\underset{\sim}{T}_0, \underset{\sim}{\bar{T}}_0\}$ since the expected output for $\underset{\sim}{T}_0$ is the complement of that for $\underset{\sim}{\bar{T}}_0$. So without loss of generality, we will consider that the outputs of the network are fault-free. Moreover, we will concern ourselves only with faults appearing at the inputs of 2T(4) gates since a fault that appears in the output of a gate is indistinguishable from the corresponding faults at the inputs to the gates fed by that output. Also faults in control lines that affect more than one gate can be considered as distributed faults in control inputs of the same gates.

So, consider any pattern of s-a-faults in the network provided that faults in the control lines affect at most one control input of any 2T(4) gate. There exists at least one faulty gate with at least one fault in its inputs such that all gates in the path from this faultygate to a network output are fault free. Since there can be at most one fault in the control inputs of this faulty gate, then as Table V shows, $\underset{\sim}{T}_0$ or $\underset{\sim}{\bar{T}}_0$ will force an erroneous logic value at the output of this gate which will be propagated to the network output via the fault-free path.

Q.E.D.

As a consequence of Theorem 5 we have the following corollary:

<u>Corollary 1</u>: In any DMC network built with 2T(4) gates, the test set $\{\underset{\sim}{T}_0, \bar{\underset{\sim}{T}}_0\}$ will detect all single s-a-faults.

Observe that there is at least one pair of control line faults that can transfer any DMC network or parts of it from test mode to normal mode during testing. When that happens, the output of the network for inputs $\underset{\sim}{T}_0$ or $\bar{\underset{\sim}{T}}_0$ becomes dependent on the network function and layout. Hence, the output of this faulty network is not always the complement of its expected output for the test patterns. The following example illustrates the above argument. For this example we assume $a_1 = a_2 = a_3 = a_4 = 0$ and that segment $(k_1, k_2, k_3, k_4) = (1,1,0,0)$ realizes a NAND gate. Then a s-a-1 fault at points α and β of Network 4 in Fig. 5 cannot be detected by test set $\{\underset{\sim}{T}_0, \bar{\underset{\sim}{T}}_0\}$.

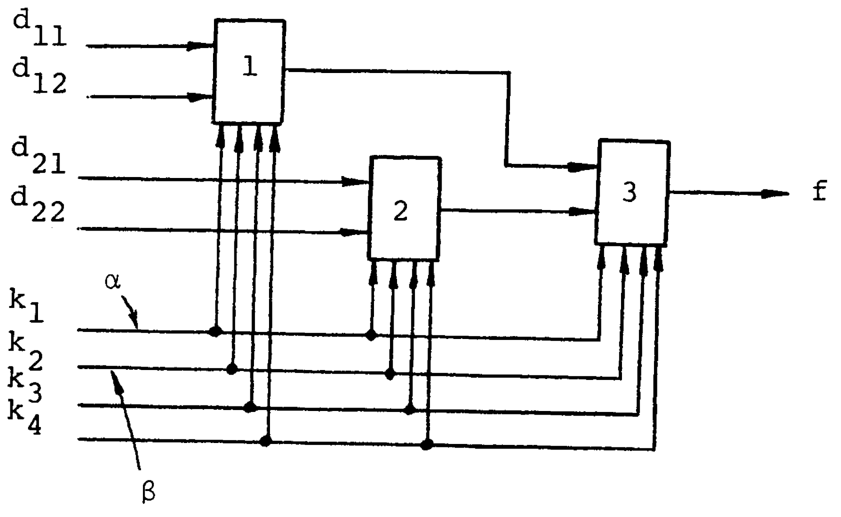

Fig. 5 Network 4

A different way to explain the above phenomenon is to note that, given any class of control line faults, the Hamming distance between the test modes and normal mode, as defined by their control values, must be large enough such that the above class of faults can never transfer the network from test mode to normal mode during testing. Two conclusions can be drawn from this fact. First, at least four control inputs are required to detect all single s-a-faults in any network. Reduction of the number of controls cannot be achieved by enlarging the test set since for any network with 3 controls or less, there exists at least one single fault that will transfer the network, or parts of it, to normal mode during testing. Any additional tests to detect such a fault would depend on the function and layout of the network and hence cannot be function-independent.

The above discussion can be formalized in the following theorem:

Theorem 6: The necessary and sufficient conditon for a network so that all single s-a-faults can be detected with a function-independent test set is that the network must have at least four controls.

Second, the concept of Hamming distance between test modes and normal mode, as defined by their control values, implies that if this distance is increased, more control faults can be detected. Ideally, normal mode should be equidistant from either test mode since fault detection capability is always expressed in terms of the smallest number of faults, all classes of which can be detected. Thus, if there are r control inputs in each gate of a network, then the optimum distance between normal mode and one of the test modes should be $\lfloor r/2 \rfloor$ which is also the minimum number of faults that will transfer a network from test mode to normal mode. Hence, the maximum number of faults occurring at the control inputs of any 2T(r) gate in any network that can be detected is,

$$t = \lfloor r/2 \rfloor - 1 = \lfloor (r-2)/2 \rfloor \quad .$$

Now, generalizing the results from a network with four controls to one with r controls, we obtain the following theorem:

<u>Theorem 7</u>:

(i) The necessary and sufficient condition for a network so that all patterns of t or fewer s-a-faults can be detected with a function-independent test set is that the network must have at least $r = 2t+2$ controls

(ii) In any m-data input DMC network built with 2T(r) gates in which the Hamming distance between normal mode and any test mode, as defined by their control values, is at least $\lfloor r/2 \rfloor$, the test set $\{\underset{\sim}{T}_0, \bar{\underset{\sim}{T}}_0\}$ where $\underset{\sim}{T}_0 = (k_1,\ldots,k_r,d_1,\ldots,d_m) = (a_1,\ldots,a_r,0,\ldots,0)$, will detect any pattern of faults provided that faults in control lines affect at most $t = \lfloor (r-2)/2 \rfloor$ control inputs of any 2T(r) gate.

(iii) In any m-data input DMC network, the test set $\{\underset{\sim}{T}_0, \underset{\sim}{\bar{T}}_0\}$ will detect all patterns of t or fewer s-a-faults.

The above theorem can be proven in a manner analogous to that for $t = 1$.

III. CONCLUSIONS AND DISCUSSION

In this paper we have presented a family of 2T(r) gates which has r control inputs and which has two modes of operation- test mode for testing and normal mode for logic operation. We have shown that we can construct DMC networks with 2T(r) gates, which can be tested with two function-independent tests for all patterns of s-a-faults as long as no gate has more than $t = \lfloor (r-2)/2 \rfloor$ control faults, provided no control inputs and data inputs are driven from a common source. We have also shown that $r = 2t+2$ is the minimum number of controls that a network must have in order to be tested for all pattern of t or fewer s-a-faults with function-independent tests.

Note that the segment of the truth table for the 2T(r) gate which contains the test with the all-0 pattern in the data inputs represents an OR function of the data inputs. The segment that contains the test with the all-1 pattern in the data inputs represents an AND function of the data inputs. Thus, during test mode, any DMC network is transformed into a network of only OR gates in terms of the data inputs of the network when $\underset{\sim}{T}_0$ is applied. The same network is transformed into a network of only AND gates in terms of the data inputs of the network when $\bar{\underset{\sim}{T}}_0$ is applied. This fact explains why logical redundancies which may exist in any DMC network in normal mode and which leave untestable faults in an equivalent conventional network, become invisible and do not affect testing of the DMC network for all specified faults.

One must note that we have considered only faults that occur at the inputs or output of a gate. We have not investigated whether there is any fault internal to the 2T(r) gate which does not manifest itself as a s-a-fault at an input or output of the gate since the effect of such a fault is dependent on the specific realization of the gate on a LSI chip. In defense of our approach, it can be said that if any fault does exist that does not appear outside the gate, it may be possible either to interconnect the internal elements of the gates in such a way that if an undetectable fault occurs, it also causes a detectable fault to occur, or to use redundancy in interconnections to reduce the probability of such an undetectable failure.

References

[1] A. D. Friedman and P. R. Memon, Fault Detection in Digital Circuits, Prentice-Hall, 1971.

[2] H. Y. Chang, E. Manning and G. Metze, Fault Diagnosis of Digital Systems, Wiley-Interscience, 1970.

[3] W. W. Peterson and E. J. Weldon, Jr., Error-Correcting Codes, 2nd ed. Cambridge, Mass: M.I.T. Press 1972.