Isomorph-Redundancy in Sequential Circuits

Debesh K. Das
Dept. of Comp. Sc. & Engg.
Jadavpur University
Calcutta - 700 032, India
dgd@jadav.ernet.in

Uttam K. Bhattacharya
Cadence Design Systems
SDF B8, NEPZ
Noida - 201 305, India
uttam@cadence.com

Bhargab B. Bhattacharya
Electronics Unit
Indian Statistical Institute
Calcutta - 700 035, India
bhargab@isical.ernet.in

Abstract

An isomorph fault in a sequential circuit makes the state diagram of the faulty machine identical to that of the fault-free machine, under the renaming of states. However, no example of a reduced sequential machine whose circuit realization is combinationally irredundant but isomorph-redundant, is yet known. This paper shows that an infinite family of such circuits can be constructed with isomorph-redundancy. Isomorph faults are then classified into various types. Their properties reveal new insight and understanding of redundancy in sequential circuits.

1. Introduction

Test generation in sequential circuits is a very complex task and the presence of redundancy makes it worse. While scan designs simplify test generation, they pay penalty in terms of additional hardware and longer test application time. Consequently, much efforts have lately been made to design fully testable non-scan sequential circuits via optimal logic synthesis [1], [2], [8], [16]. Testability of such circuits is interlaced with the presence of redundancy and therefore, their understanding is fundamental to synthesis techniques and testing [1], [2], [4], [7-9], [12-16]. In this paper, we focus on a particular type of redundancy in non-scan synchronous sequential circuits known as isomorph-redundancy.

In combinational and scan-based sequential circuits, undetectable and redundant faults are synonymous [14]. In the presence of hardware reset facility, redundancies in non-scan sequential circuits, are classified into two categories [1]. A fault in a sequential circuit is said to be a *combinationally redundant* if it is undetectable even under full scan. If a fault is not combinationally redundant, but changes the state diagram in such a manner that no input sequence starting from the reset state, can detect the change, then the fault is a *sequentially redundant fault* (SRF). The SRF's may be of three types : (i) invalid (ii) equivalent and (iii) isomorph [1]. Identification and removal of redundancies in sequential circuits were reported in [12-16].

The most comprehensive classification of faults in a general non-scan sequential circuit appears in [10], where a distinction is drawn between *undetectable* and *redundant* faults.There are some faults which are *undetectable but irredundant*, as they affect the circuit output under certain initial conditions. Therefore, a SRF as defined in [1] is undetectable, but not necessarily redundant. Further studies in undetectable and redundant faults appear in [11], [12].

2. Main Results

An isomorph fault in a sequential machine is a redundant fault that makes the fault-free and faulty machines identical, excepting the renaming of states. However, not a single instance of isomorph-redundancy was observed while analyzing the benchmark circuits [1], [2], [8], and hence, its importance was ignored by the designers. The concept of isomorph-redundancy seems to be too abstract as it is hard to design a circuit of a reduced sequential machine that is combinationally irredundant, but isomorph-redundant. In this paper, we show that there exists an infinite family of such circuits where isomorph-redundancy may occur and consequently degrade their testability. We further identified three types of isomorph faults and explored their properties. The state table description of a machine often implies non-existence of certain types of isomorph-redundancy irrespective of its circuit structure. We believe that these results will deepen our understanding of redundancy in sequential circuits, and possibly lead to new techniques of testable design.

3. Preliminaries

We assume the classical *stuck-at fault* model; both single and multiple faults are considered. A gate-level combinational circuit is said to be *irredundant* if all faults, single or multiple, are detectable by input-output experiments. The general model [6] of a synchronous sequential machine M is considered. The circuit has l input terminals fed with binary variables $x_1, x_2, \ldots x_l$. and m output terminals defining the set $\{Z_1, Z_2, \ldots Z_m\}$ of output functions. The outputs $y_1, y_2, \ldots y_k$ of k memory elements define the *present state* of the machine.

0-8186-7304-4/96 $05.00 © 1996 IEEE

The inputs $Y_1, Y_2, \ldots Y_k$ to the memory elements define the *next state* of M. If the next state and outputs are defined for every possible transition from every state, then M is said to be *completely specified.* Two states S_i and S_j are said to be *equivalent,* if all possible input sequences produce the same output response, when the machine is initially in either of the two states. A machine M is said to be *reduced* if no two states of M are equivalent. A machine is said to be *strongly connected* if every state is reachable from every other on some input sequence.

4. Isomorph-Redundancy

The behavior of isomorph fault poses a mystery to a test engineer. Till date, such a fault has been observed *only in theory* [2]. However, several synthesis techniques targeted to eliminate them have been reported [1], [2], [16].

Definition : A fault f in a reduced, completely specified, sequential machine, is an *isomorph fault* if the state table of the faulty machine is identical to that of the fault-free machine under renaming (i.e., some permutation) of the states.

Example 1: Consider a machine specified by the state table of Table 1a. If a fault changes it to that of Table 1b, then the fault is an isomorph fault as it causes renaming of S_0 by S_1 and vice-versa.

Table 1a

(fault-free machine)

x	0	1
y_1y_2		
$S_0(00)$	$S_1/0$	$S_2/0$
$S_1(01)$	$S_0/0$	$S_3/0$
$S_2(10)$	$S_3/0$	$S_3/0$
$S_3(11)$	$S_2/1$	$S_1/1$

Table 1b

(faulty machine)

x	0	1
y_1y_2		
$S_1(01)$	$S_0/0$	$S_2/0$
$S_0(00)$	$S_1/0$	$S_3/0$
$S_2(10)$	$S_3/0$	$S_3/0$
$S_3(11)$	$S_2/1$	$S_0/1$

If the fault is of isomorph type, then for every state S_i in the fault-free machine, there exists a unique state S_j^f in the faulty machine such that S_i is equivalent to S_j^f, and vice-versa. Therefore, an isomorph fault is sequentially *undetectable as well as redundant* according to the definition of [10-12]. Henceforth, a circuit that has an isomorph fault will be called *isomorph-redundant.*

Definition [12]: A fault is said to be *strongly redundant* if every pair of corresponding states (states having the same encoding) of the fault free and faulty machines are equivalent.

Lemma 1 : If the circuit realization is combinationally irredundant, then an isomorph fault in a reduced machinne cannot be strongly redundant.

Proof : Clear.

We will consider a *reduced* machine, and *combinationally irredundant* circuit realization. We will also assume that the machine is under the *free mode* [10] of operation, i. e., it does not use power-up sequences, synchronization sequences or hardware reset signals. Reset lines if any, will be treated as regular inputs. Thus, the machine may start operation from any unknown state.

4.1 Isomorphism using a swapping pair of functions

Example 2: Consider a sequential circuit with two flip-flops and a single output line. Let an isomorph fault in the circuit cause an interchange of two states say, $S_1(01)$ and $S_2(10)$. One way to achieve this is to identify two next-state functions F_1,F_2 and their suitable circuit realizations, such that $F_1(F_2)$ changes to $F_2(F_1)$ under the fault. We also need a reduced machine and a combinationally irredundant circuit realization. To design such a network, we recall the concept of swapping functions [3]. Isormorphism can also be caused in many other ways as we will show later.

4.1.1 Swapping of next-state functions

Definition: Let $N_1(N_2)$ denote an irredundant combinational circuit realizing a function F_1 (F_2). If there exists a stuck-at fault under the effect of which the network $N_1(N_2)$ realizes F_2 (F_1), (i.e., F_1 changes to F_2, and F_2 changes to F_1), then F_1 and F_2 are said to form a swapping pair [3].

The function realized by the network N_1 (Fig. 1) is given by:

$$F_1 = x_1x_2 + x_2x_3 + x_3x_4 + x_2x_4 + x_1x_4 .$$

The above example is taken from [4] . This is an interesting circuit that shows that a *unate function* can be realized by a single-output, *irredundant binate* combinational circuit (a circuit that contains reconvergent fanouts with unequal inversion-parity paths).

A single stuck-at zero (s-a-0) fault at line l (denoted by l^0), now changes F_1 to F_2 where

$$F_2 = x_1x_3 + x_2x_3 + x_3x_4 + x_2x_4 + x_1x_4 .$$

F_1 and F_2 are P-equivalent [4], because by permuting x_2 and x_3 in F_1, we get F_2. The function F_2 can be realized by a network N_2 (which is same as N_1 with inputs x_2 and x_3 swapped), where a similar stuck-at fault changes F_2 to F_1. Thus the functions F_1 and F_2 form a swapping pair.

The next-state functions Y_1 and Y_2, willbe designed using the networks N_1 and N_2, such that a fault changes Y_1 to Y_2, and Y_2 to Y_1.

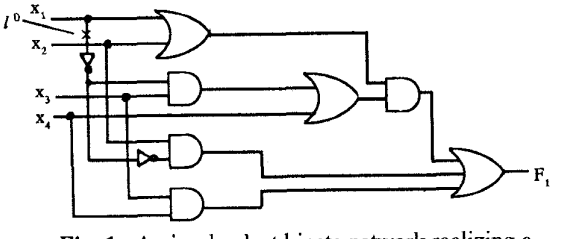

Fig. 1 : An irredundant binate network realizing a unate function F_1

4.1.2 Designing the output

The output logic is to be so designed that the fault induces an isomorphism with respect to interchange of states S_1 and S_2. Thus for each input I_j, one should have :

$Z(S_1, I_j) = Z^f(S_2, I_j)$, and $Z(S_2, I_j) = Z^f(S_1, I_j)$

$Z(S_0, I_j) = Z^f(S_0, I_j)$, and $Z(S_3, I_j) = Z^f(S_3, I_j)$

4.1.3 The overall circuit

We choose $Y_1 = F_1$ and $Y_2 = F_2$ which are realized by the circuits shown in Fig. 1. The output function is chosen as :

$Z = (F_1 \oplus y_1) + (F_2 \oplus y_2)$.

Fig. 2 shows the complete sequential circuit, whose state table (Table 2a), is **reduced.** Moreover, the circuit is **combinationally irredundant.** The state table of the faulty machine (Table 2b) in the presence of the fault l^0, is isomorphic to that of the fault-free machine with interchange of states S_1 and S_2. Thus, the above stuck-at fault is an **isomorph fault.**

4.2 Isomorphism keeping the outputs invariant

Example 3: The next circuit consists of two PI's fed by x_1, x_2, one output line and two memory elements encoding four states. The next-state functions are : $Y_1 = x_2 \oplus y_1$ and $Y_2 = x_1 \oplus y_2$. We use a curious circuit structure (Fig. 3a) taken from Hayes [5]. This two-input, two-output circuit implements trivial functions $F_1(x_1, x_2) = x_1$ and $F_2(x_1, x_2) = x_2$ and yet it is irredundant. More interestingly, the fault l^0 swaps the

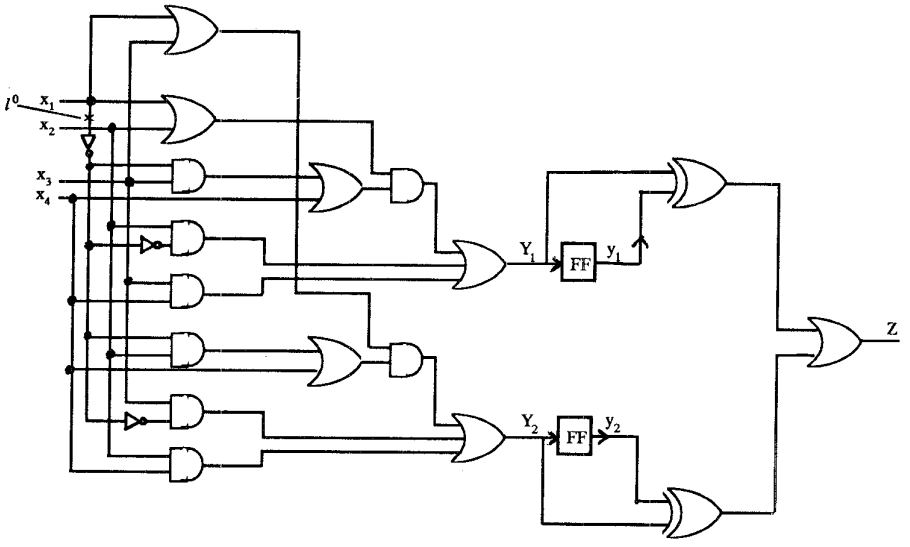

Fig.2: A combinationally irredundant sequential circuit with an isomorph-redundant fault (l^0) of type-A

Table 2a: State table of the fault-free machine of Fig. 2

Decimal equivalents of inputs : $x_1x_2x_3x_4$

y_1y_2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S_0(00)$	$S_0/0$	$S_0/0$	$S_0/0$	$S_3/1$	$S_0/0$	$S_3/1$	$S_3/1$	$S_3/1$	$S_0/0$	$S_3/1$	$S_1/1$	$S_3/1$	$S_2/1$	$S_3/1$	$S_3/1$	$S_3/1$
$S_1(01)$	$S_0/1$	$S_0/1$	$S_0/1$	$S_3/1$	$S_0/1$	$S_3/1$	$S_3/1$	$S_3/1$	$S_0/1$	$S_3/1$	$S_1/0$	$S_3/1$	$S_2/1$	$S_3/1$	$S_3/1$	$S_3/1$
$S_2(10)$	$S_0/1$	$S_0/1$	$S_0/1$	$S_3/1$	$S_0/1$	$S_3/1$	$S_3/1$	$S_3/1$	$S_0/1$	$S_3/1$	$S_1/1$	$S_3/1$	$S_2/0$	$S_3/1$	$S_3/1$	$S_3/1$
$S_3(11)$	$S_0/1$	$S_0/1$	$S_0/1$	$S_3/0$	$S_0/1$	$S_3/0$	$S_3/0$	$S_3/0$	$S_0/1$	$S_3/0$	$S_1/1$	$S_3/0$	$S_2/1$	$S_3/0$	$S_3/0$	$S_3/0$

Table 2b: State table of the machine of Fig. 2 in the presence of the fault l s-a-0

y_1y_2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S_0(00)$	$S_0/0$	$S_0/0$	$S_0/0$	$S_3/1$	$S_0/0$	$S_3/1$	$S_3/1$	$S_3/1$	$S_0/0$	$S_3/1$	$S_2/1$	$S_3/1$	$S_1/1$	$S_3/1$	$S_3/1$	$S_3/1$
$S_2(10)$	$S_0/1$	$S_0/1$	$S_0/1$	$S_3/1$	$S_0/1$	$S_3/1$	$S_3/1$	$S_3/1$	$S_0/1$	$S_3/1$	$S_2/0$	$S_3/1$	$S_1/1$	$S_3/1$	$S_3/1$	$S_3/1$
$S_1(01)$	$S_0/1$	$S_0/1$	$S_0/1$	$S_3/1$	$S_0/1$	$S_3/1$	$S_3/1$	$S_3/1$	$S_0/1$	$S_3/1$	$S_2/1$	$S_3/1$	$S_1/0$	$S_3/1$	$S_3/1$	$S_3/1$
$S_3(11)$	$S_0/1$	$S_0/1$	$S_0/1$	$S_3/0$	$S_0/1$	$S_3/0$	$S_3/0$	$S_3/0$	$S_0/1$	$S_3/0$	$S_2/1$	$S_3/0$	$S_1/1$	$S_3/0$	$S_3/0$	$S_3/0$

functionality of these two outputs.

The output function is chosen as $Z = y_1 y_2$.

The complete sequential circuit is now shown in Fig. 3b. Its state table (Table 3a) is verified to be **reduced.** The portion of the circuit shown within dotted lines is irredundant [5]. It is easy to verify that the above sequential circuit is **combinationally irredundant.**

Table 3a
(fault-free machine)

x_1x_2 / y_1y_2	00	01	10	11
$S_0(00)$	$S_0/0$	$S_2/0$	$S_1/0$	$S_3/0$
$S_1(01)$	$S_1/0$	$S_3/0$	$S_0/0$	$S_2/0$
$S_2(10)$	$S_2/0$	$S_0/0$	$S_3/0$	$S_1/0$
$S_3(11)$	$S_3/1$	$S_1/1$	$S_2/1$	$S_0/1$

Table 3b
(faulty machine)

x_1x_2 / y_1y_2	00	01	10	11
$S_0(00)$	$S_0/0$	$S_1/0$	$S_2/0$	$S_3/0$
$S_2(10)$	$S_2/0$	$S_3/0$	$S_0/0$	$S_1/0$
$S_1(01)$	$S_1/0$	$S_0/0$	$S_3/0$	$S_2/0$
$S_3(11)$	$S_3/1$	$S_2/1$	$S_1/1$	$S_0/1$

The faulty machine under the fault l^0 is shown in Table 3b.

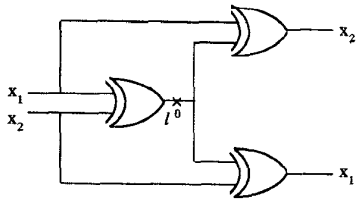

Fig. 3a : An irredundant combinational circuit

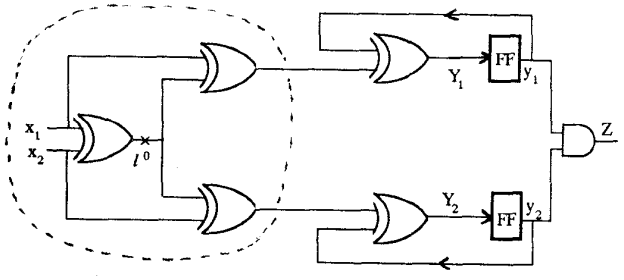

Fig. 3b : An isomorph-redundant fault (l^0)

Clearly, the fault-free and the faulty machines are **isomorphic** under relabeling of states S_1 and S_2. The next-state and the output functions, in the presence of fault are: $Y_1^f = x_1 \oplus y_1$; $Y_2^f = x_2 \oplus y_2$ and $Z^f = y_1 y_2$.

4.3 Isomorphism by logical swapping

In order to portray isomorph-redundancy, the encoding of at least two states of the faulty machine must be different from those of the respective equivalent states of the fault-free machine (by Lemma 1). Therefore, it is natural to think that the fault should affect the next-state functions. In both the examples presented earlier, the isomorph-fault changes the next-state functions. Now we show an interesting example: the fault changes none of the next-state functions, yet the faulty machine is isomorphic to the original one. Thus, swapping of two states is being experienced *logically* rather than changing the next-state functions *physically.* We believe that this shows a new kind of isomorph-redundancy that makes its understanding more complex than thought before.

Example 4 : A sequential circuit is shown in Fig. 4 and its state table is given in Table 4a. Combinational irredundancy of this circuit can easily be checked. The machine is reduced, and the next-state and output functions are given by

$$Y = x_1 \oplus x_2 \oplus y;\ Z = \bar{x}_1 y + \bar{x}_2 \bar{y}.$$

The fault l^0 now changes the state table as shown in Table 4b. Notice that the fault does not affect the next-state logic, but the faulty machine is isomorphic to the original one with respect of interchange of states S_0 and S_1. The fault occurs in the output circuit and swaps the states logically.

Table 4a
(fault-free machine)

x_1x_2 / y	00	01	10	11
$S_0(0)$	$S_0/1$	$S_1/0$	$S_1/1$	$S_0/0$
$S_1(1)$	$S_1/1$	$S_0/1$	$S_0/0$	$S_1/0$

Table 4b
(faulty machine)

x_1x_2 / y	00	01	10	11
$S_1(1)$	$S_1/1$	$S_0/0$	$S_0/1$	$S_1/0$
$S_0(0)$	$S_0/1$	$S_1/1$	$S_1/0$	$S_0/0$

In the presence of the fault,
$Y^f = x_1 \oplus x_2 \oplus y$ and $Z^f = \bar{x}_2 y + \bar{x}_1 \bar{y}$

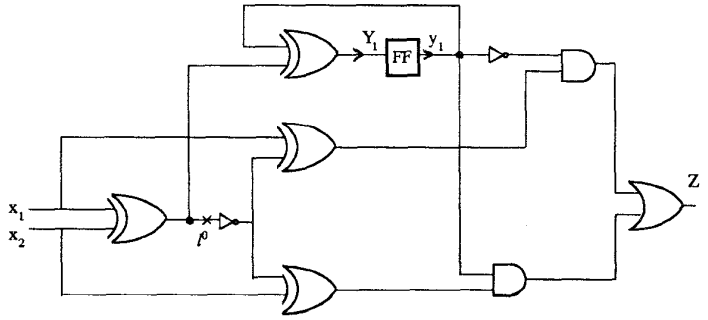

Fig. 4 : A combinationally irredundant sequential circuit with isomorph-redundancy of type-C under the fault l s-a-0

4.4 Isomorph fault in a two-level circuit

Isomorph-redundancy may be observed under a multiple stuck-at fault even if the next-state and output functions are designed with two-level prime and irredundant circuits. However, no single stuck-at fault in such a sequential circuit can cause isomorphism [16].

Example 5 : Consider the machine of Table 1a. The machine is **reduced**. However, the standard synthesis procedure may lead to isomorph-redundancy under a multiple fault.

Denoting states as $\{y_1y_2\}$, we encode S_0 as 00, S_1 as 01, S_2 as 10, S_3 as 11; this yields :

$Y_1 = \bar{y}_1x + y_1\bar{y}_2 + y_1\bar{x}$; $Y_2 = y_2x + y_1\bar{y}_2 + \bar{y}_2\bar{x}$;

$Z = y_1y_2$ (Fig. 5a).

The circuit is shown in Fig. 5b, which is combinationally irredundant. The multiple fault

$\{l_1^0, l_2^0\}$change the machine as shown in Table 1b. The fault changes only Y_2 to:

$Y_2^f = y_1\bar{y}_2 + \bar{y}_2 = \bar{y}_2$, causing the faulty machine **isomorphic** to the original machine.

5. Classification of isomorph-redundancy

The above examples motivate us to classify isomorph-redundancy into three types: If the fault affects both the next state function(s) and the output(s), we call it *type-A*. if it changes only the next state function (resp. only the output), we call it *type-B* (resp. *type-C*). The isomorph-faults in Examples 2, 3 and 4 are of type-A, B and C respectively. Example 5 is of type-B.

In Example 2, the next-state functions are swapped under the fault, i.e., Y_1(faulty) = Y_2(fault-free) and vice-versa. As there exist numerous examples of swapping pair of functions [3], it is possible to construct an *infinite family* of combinationally irredundant sequential circuits which exhibit isomorph-redundancy of type-A. Similar examples can be constructed for type-B or type-C fault.

Properties

Let M(I, O, S, δ, Z) be a sequential machine that changes to M^f(I, O, S, δ^f, Z^f) under an isomorph fault. Clearly, there exists a permutation Π that maps the set of states onto itself, i.e., $\Pi : S \rightarrow S$, which in turn induces a set of disjoint ordered cycles in S.

Example 6 : Let an isomorph fault induce the following permutation of states : $\Pi(S_0, S_1, S_2, S_3, S_4, S_5, S_6, S_7) \rightarrow (S_2, S_1, S_3, S_5, S_6, S_0, S_4, S_7)$.

In this case, Π induces the following cycles: $\{S_0, S_2, S_3, S_5\}$, $\{S_4, S_6\}$, $\{S_1\}$, $\{S_7\}$. In other words, the fault changes the state S_0 to S_2, S_2 to S_3, S_3 to S_5, S_5 back to S_0. Similarly, for the second cycle, S_4 and S_6 are interchanged. The last two elements show cycles of unit length, i.e., they map onto themselves.

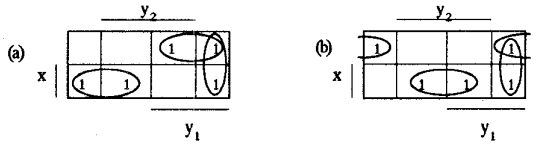

Fig. 5a : K-map (a) for Y_1 and (b) for Y_2

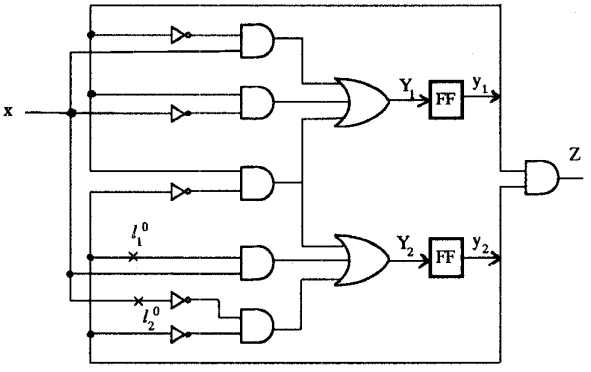

Fig. 5b : A combinationally-irredundant sequential circuit with isomorph-redundancy of type-B

The *length of a cycle* is defined to be the number of states involved in the cycle.

In general, a permutation cycle ρ_i of length p, denoted by $\rho_i = \{S_{i1}, S_{i2}, \ldots S_{ij}, S_{i,j+1}, \ldots S_{ip}\}$, is an ordered set of states. Then the state S_{ij} of the fault-free machine is to be re-encoded by $S_{i,(j+1) \bmod p}$, for $1 \le j \le p$ in the faulty machine.

We define **pred** and **succ** of a state as follows:

$$\text{pred}(S_{ij}) = S_{i,(j+p-1) \bmod p} \text{ for } p > 1,$$
$$= S_{ij} \text{ for } p = 1;$$

and, $$\text{succ}(S_{ij}) = S_{i,(j+1) \bmod p} \text{ for } p > 1,$$
$$= S_{ij} \text{ for } p = 1.$$

For each $S_{ij} \in \Pi$, its pred and succ are unique.

Example 7 : $\text{succ}(S_0) = S_2$, $\text{pred}(S_1) = S_1$.

Lemma 2 [17]: If an isomorph fault described by a permutation Π transforms M to M^f, then the following conditions hold for any input I_a:

(i) $\delta^f(I_a, S_j) = \text{succ}(\delta\{I_a, \text{pred}(S_j)\})$

(ii) $Z^f\{I_a, S_j) = Z\{I_a, \text{pred}(S_j)\}$.

The occurrence of an isomorph fault is highly dependent on the state table and its circuit structure. Given a state table of a sequential machine, numerous circuit realizations are possible. One pertinent question is: Can we predict anything about isomorphism (whether or not it can occur under a stuck-at fault) from the state table alone, irrespective of its circuit design ? We now show that, it is often possible to ascertain the occurrence of type-B and type-C faults.

Theorem 1: No type-B (C) isomorph-fault can achieve a permutation Π, if for each permutation cycle $\rho_i = \{S_{i1}, S_{i2}, \ldots S_{ij}, S_{i,j+1}, \ldots S_{ip}\}$, for $p > 1$, the following condition is false (true): $Z\{I_a,S_{i1}\} = Z\{I_a,S_{i2}\} = Z\{I_a,S_{i3}\} = \ldots\ldots = Z\{I_a,S_{ij}\} = Z\{I_a,S_{i,j+1}\} = \ldots\ldots = Z\{I_a,S_{ip}\}$.

Proof: Follows from Lemma 2.

Example 8 : In Example 4 (Table-4a), no type-B fault can cause a permutation :

$\Pi(S_0, S_1) \rightarrow (S_1, S_0)$, as $Z\{01, S_0\} = Z\{01, S_1\}$ and $Z\{10, S_0\} = Z\{10, S_1\}$.

Similarly, in Example 3 (Table-3a), no type-C fault can cause a permutation $\Pi(S_0, S_1, S_2, S_3) \rightarrow (S_0, S_2, S_1, S_3)$, because in the cycle $\{S_1, S_2\}$, for every input I_a, $Z\{I_a,S_1\} = Z\{I_a,S_2\}$.

Corollary 1: If $\forall S_i$, $Z\{I_a,S_i\}$ is distinct from $Z\{I_a,S_j\}$, $i \neq j$ for at least one input $I_a \in I$, then an isomorph fault of type-B can never occur regardless of the circuit realization.

Example 9: No sequential circuit realizing Table-2a, can have isomorph-redundancy of type-B.

Theorem 2 : No type-C fault can achieve an isomorphism defined by a given permutation Π, if the following condition is true: $\delta\{I_a,S_i\} \neq succ(\delta\{I_a,pred(S_i)\})$, for any state $S_i \in S$.

Proof: Follows from Lemma 2.

Example 10 : No type C fault can change the machine of Table 2a to Table 2b, regardless of its circuit realization as the correponding Π induces the following cycles: $\{S_0\}$, $\{S_1, S_2\}$, $\{S_3\}$, and for input = 12, $\delta\{12,S_1\} \neq succ(\delta\{12,pred(S_1)\})$.

Corollary 2 : Let an isomorph fault of type-C be described by Π. If a state S_i lies in a permutation cycle of length 1, then for every input $I_a \in I$, $\delta\{I_a,S_i\}$ also lies in a cycle of length 1.

Corollary 3: A circuit realizing a strongly connected sequential machine cannot exhibit an isomorph fault of type C, if any state lies in a permutation cycle of length 1.

Proof : Follows from Lemma 1 and Corollary 2.

Example 11 : The machine shown in Table 3a is strongly connected. Regardlless of its circuit realization, no isomorph fault of type C can achieve a permutation Π such that some state lies in a cycle of length 1 (e.g., $\{S_0\}$, $\{S_1, S_2\}$, $\{S_3\}$).

6. Conclusions

No real isomorph-fault was known to exist till date. In this paper, we have presented some curious examples of isomorph-redundancy in combinationally irredundant, reduced and strongly connected sequential machines. An infinite family of such circuits can be constructed using the concept of swapping functions [3]. We also identified three kinds of such redundancy (type-A, B and C). For type-B and C redundancy to occur, the state table of the machine must satisfy certain conditions, regardless of its circuit realization. Complete characterization of occurrence of such faults is yet to be settled. The examples cited in this paper indicate that isomorph-redundancy in sequential circuits is not so unlikely to appear as thought before. Identification of isomorph-redundancy seems to be a difficult problem. New DFT techniques have been reported recently to remove isomorph and other sequential redundancies in an arbitrary circuit under the multiple stuck-at fault model [17].

References

1. S. Devadas, H-K T. Ma, A. R. Newton, and A. Sangiovanni Vincentelli, "Irredundant sequential machines via optimal logic synthesis," *IEEE TCAD,* vol. 9, no.1, pp. 8-18, 1990.
2. A. Ghosh, S. Devadas, and A. R. Newton, "Sequential test generation and synthesis for testability at the register-transfer and logic levels," *IEEE TCAD,* vol. 12, no. 5, pp. 579-598, May 1993.
3. D. K. Das, S. Chakraborty, and B. B. Bhattacharya, "On the nature of faulty functions in combinational circuits under stuck-at faults," *submitted for publication,* 1996.
4. S. Chakraborty, D. K. Das, and B. B. Bhattacharya, "Logical redundancies in irredundant combinational circuits," *JETTA,* vol. 4, pp. 120-125, 1993.
5. J. P. Hayes, "On the properties of irredundant logic networks," *IEEE TC,* vol. 25, pp. 884-892, 1976.
6. Z. Kohavi, *Switching and Finite Automata Theory.* McGraw-Hill, Inc., 1970.
7. S. Devadas, H-K T. Ma, and A. R. Newton, "Redundancies and don't cares in sequential logic synthesis," *JETTA,* pp. 15-30, 1990.
8. P. Ashar, S. Devadas, and A. R. Newton, "Irredundant Interacting sequential machines via optimal logic synthesis," *IEEE TCAD,* vol. 10, no. 3, pp. 311-325, 1991.
9. H. Coo, G. D. Hachtel, and F. Somenzi, "Redundancy identification/removal and test generation for sequential circuits using implicit state enumeration," *IEEE TCAD,* vol. 12, no. 7, pp. 935-945, 1993.
10. I. Pomeranz and S. M. Reddy, "Classification of faults in synchronous sequential circuits," *IEEE TC,* vol. 42, pp. 1066-1077, Sept. 1993.
11. I. Pomeranz, S. M. Reddy, and J. H. Patel, "Theory and practice of sequential machine testing and testability," in *Proc. of FTCS-23,* 1993, pp. 330-337.
12. I. Pomeranz and S. M. Reddy, "On identifying undetectable and redundant faults in synchronous sequential circuits," in *Proc. VTS,* 1994, pp. 8-14.
13. M. Abramovici, D. T. Miller, and R. K. Roy, "Dynamic redundancy identification in automatic test generation," *IEEE TCAD,* pp. 404-407, March 1992.
14. V. D. Agrawal and S. T. Chakradhar, "Combinational ATPG theorems for identifying untestable faults in sequential circuits," in *Proc. European Test Conference,* pp. 249-253, April 1993.
15. K. T. Cheng, "On removing redundancy in sequential circuits," in *Proc. 28-th DAC,* pp.164-169, June 1991.
16. S. Devadas and K. Keutzer, "A unified approach to the synthesis of fully testable sequential machines," *IEEE TCAD,* pp. 39-50, January 1991.
17. D. K. Das and B. B. Bhattacharya, "Testable design of non-scan sequential circuits using extra logic," in *Proc. Forth Asian Test Symposium,* Bangalore, Nov. 1995.