Checking Experiments for Stable Memory Faults in Sequential Machines

HIDEO FUJIWARA and KOZO KINOSHITA*

Abstract—This paper is concerned with stable memory faults and presents the design of checking experiments for the stable faults in sequential machines. It is shown that if a machine M is reduced, then any proper stable fault of M results in a machine with fewer states. Accordingly, checking experiments for stable faults need only to verify that the machine under test has the same number of states as the fault free machine M has.

This paper also introduces a machine which has an optimal checking experiment of length $n - 1 + [\log_2 n]$ for stable faults where n is the number of states. A design procedure is presented in which an arbitrary machine is augmented to the machine having an optimal checking experiment by adding a special input symbol to the original machine. Hardware implementation for it is also considered.

***Index Terms*—checking experiments, memory faults, fault detection, easily testable machines, sequential machines, feedback shift registers.**

I. INTRODUCTION

In general, a fault in a sequential machine affects the next-state function and/or the output function, and it transforms the original machine to another machine which has different functions. The study of checking experiments for sequential machines has been primarily concerned with the classes of faults that affect both the next-state function and the output function [3]-[6]. However, for the cases in which the next-state function and the output function are realized by separate circuits, it is reasonable to consider subclasses of faults that affect either the next-state function or the output function, but not both. Thus we can consider the next-state fault and the output fault, separately. This approach first appeared in Friedman and Menon [7] where some reduction in the length of checking experiments was achieved. For the output faults, Boute [8] gave a method for designing optimal and near optimal checking experiments.

A further subclass of next-state faults can be considered when the next-state functions are realized by two separate circuits which are memory and combational circuit. Meyer [1] introduced a machine representation of memory faults and developed the synthesis of fault tolerant sequential machines.

*The authors are with the Department of Electronic Engineering, Osaka University, Osaka, Japan.

0099-1708/78/0100-0003 $01.98/0 © 1978 Computer Science Press, Inc.

where

1) $S^\mu = \text{Im}(\mu) = \{\mu(S_i) | S_i \epsilon S\}$, the image of μ.
2) $\delta^\mu = \mu\delta$ restricted to $S^\mu \times I$.
3) $\lambda^\mu = \lambda$ restricted to $S^\mu \times I$.

A memory fault μ is *stable* if $\mu\mu = \mu$, i.e., $\mu(\mu(S_i)) = \mu(S_i)$ for all states S_i. The identity function is referred to as an *improper* fault, all other faults being *proper*.

Stable faults are of interest since many types of physical memory faults may be represented by a machine fault of this type. In particular, any combination of "stuck-at 0" and "stuck-at 1" faults in one or more two-state memory cells is represented by a stable fault of the corresponding sequential machine [1].

Example: To illustrate memory faults, consider machine M_1 given by Fig. 1. Function tables of faults μ_1 and μ_2 are specified by Fig. 2, and the corresponding faulty machines are shown in Fig. 3. μ_1 is not stable, but μ_2 is stable.

	0	1
A	A , 1	B , 0
B	C , 0	A , 0
C	B , 0	C , 1

Fig. 1 Machine M_1

	μ_1	μ_2
A	B	A
B	C	B
C	A	B

Fig. 2 Memory faults of machine M_1

$M_1^{\mu_1}$

	0	1
A	B , 1	C , 0
B	A , 0	B , 0
C	C , 0	A , 1

$M_1^{\mu_2}$

	0	1
A	A , 1	B , 0
B	B , 0	A , 0

Fig. 3 The results of faults μ_1 and μ_2

This paper is concerned with stable memory faults and presents the design of checking experiments for the stable faults in sequential machines. It is shown that if a machine M is reduced, then any proper stable fault of M results in a machine with fewer states. Accordingly, checking experiments for stable faults need only to verify that the machine under test has the same number of states as the fault free machine M has. We also introduce a machine which has an optimal checking experiment of length $n - 1 + \lceil \log_2 n \rceil$ for stable faults where n is the number of states and $\lceil x \rceil$ is the smallest integer not smaller than x. A design procedure is presented in which an arbitrary machine is augmented to the machine having an optimal checking experiment by adding a special input symbol to the original machine. Hardware implementation for it is also considered.

II. CHECKING EXPERIMENTS FOR STABLE FAULTS

The sequential machines considered in this paper are assumed to be finite state, synchronous, deterministic, strongly connected, and completely specified. The machine M will be represented by a quintuple $M = (S, I, O, \delta, \lambda)$ where $S = \{S_1, S_2, \cdots, S_n\}$ is a finite set of states, $I = \{I_1, I_2, \cdots, I_m\}$ is a finite set of input symbols, $O = \{O_1, O_2, \cdots, O_l\}$ is a finite set of output symbols, $\delta: S \times I \rightarrow S$ is called the next-state function, and $\lambda: S \times I \rightarrow 0$ is called the output function.

A *checking experiment* is an input-output sequence such that when the input sequence is applied to the machine under test, an output sequence is produced which establishes whether or not the machine under test is equivalent to the correctly operating machine, subject to some fault assumptions. An experiment is said to be *adaptive* or *preset* depending on whether the next input signal to apply is or is not based upon the output signals previously produced by the machine. A *synchonizing sequence* for a sequential machine is an input sequence whose application is guaranteed to leave the machine in a certain final state, regardless of the particular initial state of the machine. A *homing sequence* for a sequential machine is an input sequence whose application makes it possible to determine the final state of the machine by observing the corresponding output sequence that the machine produces. A *distinguishing sequence* is an input sequence whose application makes it possible to determine the initial state of the machine by observing the corresponding output sequence that the machine produces. A *transfer sequence* from state S_i to state S_j is an input sequence which transfers the machine from state S_i to state S_j.

A *memory fault of a sequential machine M* [1] is a function on the states of M. If $M = (S, I, O, \delta, \lambda)$ is a sequential machine and $\mu: S \rightarrow S$ is a memory fault of M, the *result of* μ is the sequential machine

$$M^{\mu} = (S^{\mu}, I, O, \delta^{\mu}, \lambda^{\mu})$$

For any proper stable fault, we have the following theorem.

Theorem 1: If M is reduced and μ is a proper stable fault, then $|S^\mu| < |S|$ where S and S^μ are the state-sets of M and M^μ and $|A|$ is the number of elements of a set A.

Proof: Assume that $|S^\mu| = |S|$. By definition, $S^\mu = \text{Im}(\mu)$. Thus $|S^\mu| = |S|$ implies $|\text{Im}(\mu)| = |S|$. Hence μ is a bijection.

Since μ is stable, $\mu\mu = \mu$ and hence

$$\mu\mu\mu^{-1} = \mu\mu^{-1} \tag{1}$$

Since μ is a bijection,

$$\mu^{-1}\mu = \mu\mu^{-1} = I \text{ (the identity function)} \tag{2}$$

From (1) and (2) we have, $\mu = I$. This is a contradiction. Q.E.D.

This theorem verifies that in order to check stable memory faults we have only to establish whether the machine under test has the same number of states as the fault free machine has. Based on the theorem, we will consider the design of checking experiments for stable faults in sequential machines. Our method is mainly based on that of Hennie [3], and we assume that readers are familiar with the principle of his method.

First, we consider sequential machines having distinguishing sequences. Let $M = (S, I, O, \delta, \lambda)$ be an n-state m-input machine. Let X_d be a distinguishing sequence. The checking experiment consists of two parts. The first part of the checking experiment is the initializing part which brings the machine under test to the starting state for the experiment. This can be done by a homing sequence or a synchronizing sequence. The second part of the checking experiment carries the correctly operating machine through all its states, displays all the different responses to X_d, and thus verifies that there exist n distinct states. The second part of the checking experiment thus has the form:

Input:	$T(-, S_i)$		X_d
State:		S_i	
Output:			Z_i

for all states S_i of M, where $Z_i = \lambda(S_i, X_d)$ and $T(-, S_i)$ is a transfer sequence from some state to state S_i.

Although the checking experiment is functionally subdivided into two parts, these parts need not be physically separated from each other.

Example: Let us construct a checking experiment for machine M_1 given by Fig. 1. $X_d = 01$ is both a distinguishing sequence and a homing sequence. By applying X_d and observing the output response, we can establish the initial state and the final state. Suppose that the machine is in state A, then the rest of the checking experiment is:

Input:		01	01	01
State:	A	B	C	
Output:		10	01	00

Next, we consider machines not having any distinguishing sequence. Let $\{X_{i1}, X_{i2}, \ldots, X_{ik}\}$ denote a set of characterizing sequences for state S_i of machine M (defined in Farmer [6]). Let $T(S_i, S_j)$ denote the input sequence required to transfer machine M from state S_i to state S_j.

Let

$$Y_{i1} = X_{i1}T(Q_{i1}, S_i)$$
$$Y_{i2} = Y_{i1}^n X_{i2}T(Q_{i2}, S_i)$$
$$Y_{i3} = Y_{i2}^n Y_{i1}^n X_{i3}T(Q_{i3}, S_i)$$

$$\ldots\ldots\ldots$$

$$Y_{i(k-1)} = Y_{i(k-2)}^n Y_{i(k-3)}^n \ldots Y_{i1}^n X_{i(k-1)}T(Q_{i(k-1)}, S_i)$$
$$Y_{ik} = Y_{i(k-1)}^n Y_{i(k-2)}^n \ldots Y_{i1}^n X_{ik}$$
$$L_{S_i} = Y_{ik}$$

where Y_{il}^n denotes the concatenation of n identical input sequence Y_{il} and $Q_{il} = \sigma(S_i, X_{il})$ for $1 \leq l \leq k$. L_{S_i} is defined as a *locating sequence* for state S_i of M (defined in Farmer [6]).

Similar to the case of machines having distinguishing sequences, for a machine M not having any distinguishing sequence, the checking experiment on M consists of two parts. The first part is the initializing part. The second part of the checking experiment carries the correctly operating machine through all its states, displays all the different responses to L_{S_i}. The second part thus has the form:

Input:	$T(-, S_i)$		L_{S_i}
State:		S_i	
Output:			Z_{S_i}

for all states S_i of M.

Example: Consider machine M_2 shown in Fig. 4. We see that the set $\{0, 10\}$ is a set of characterizing sequences for machine M_2. Locating sequences for each state are as follows:

$$L_A = (00)^4 10$$
$$L_B = (00)^4 10$$
$$L_C = (01)^4 10$$
$$L_D = 0^4 10$$

$X_h = 010$ is a homing sequence of M. By applying X_h and observing the

	0	1
A	B , 0	D , 0
B	A , 0	B , 0
C	D , 1	A , 0
D	D , 1	C , 0

Fig. 4 Machine M_2

output response, we can establish the final state. Suppose that machine under test is in state A, then the rest of the checking experiment is:

Input:	L_A	L_D	$T(D, C) = 1$	L_C	L_B
State:	A	D	D	C	B
Output:	Z_A	Z_D	0	Z_C	Z_B

III. OPTIMAL CHECKING EXPERIMENTS

In this section we consider a procedure to augment a given machine by adding an extra input symbol so that the augmented machine has an optimal checking experiment defined below. We assume that the given machine is not required to be reduced, strongly-connected, or completely specified.

In the previous section we have shown that in order to check stable faults it is necessary and sufficient to establish whether the machine under test has the same number of states as the fault-free machine has. In order to establish the existence of n distinct states (n is the number of states of the fault-free machine), a distinguishing sequence must appear at least n times in a checking experiment. The length of the distinguishing sequence which identifys n different states is not smaller than $\lceil \log_2 n \rceil$ when the machine has only two output symbols. Thus, in this case the length of the checking experiment is not smaller than $n - 1 + \lceil \log_2 n \rceil$ even if the n distinguishing sequences can be overlapped with each other in the experiment. Therefore, we can make the following definition: An *optimal checking experiment* for stable memory faults is a checking experiment of length $n - 1 + \lceil \log_2 n \rceil$.

Consider the set of binary p-tuples B^p formed by the pth Cartesian power of $B = \{0,1\}$. The pth order de Bruijn graph G_p is a directed graph with 2^p vertices, labeled by the elements of B^p. The vertices x and y of G_p are joined by an arc (x, y), directed from x to y, if and only if $(x_1, x_2, \ldots, x_{p-1}) = (y_2, y_3, \ldots, y_p)$ where $x = (x_1, x_2, \ldots, x_p)$ and $y = (y_1, y_2, \ldots, y_p) \in B^p$. The de Bruijn graphs of order 2 and 3 are shown in Fig. 5.

In general, in the de Bruijn graph of order p, we can find a binary sequence of period L for all possible $L \leq 2^p$ such that each of the subse-

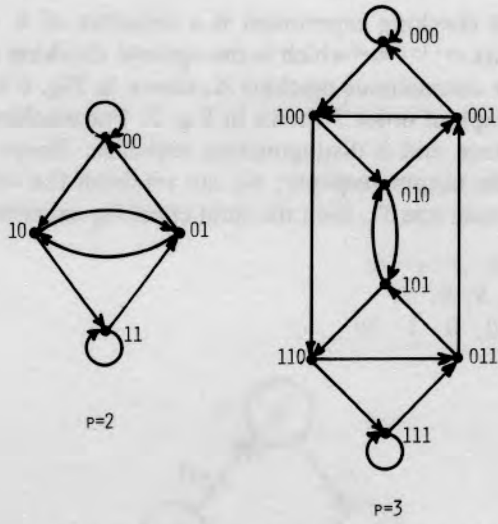

Fig. 5 The de Bruijn graphs of order $p = 2$ and 3

quences of length p is distinct (see Golomb [12] for proof). By using this sequence, we introduce an n-state autonomous machine A_n having an optimal checking experiment in the following manner. Let $[v_1, v_2, \ldots, v_n]$ be a cycle of length n, namely a closed sequence of n distinct vertices in the de Bruijn graph of order $p = \lceil \log_2 n \rceil$. Then machine $A_n = (S, \{\epsilon\}, \{0, 1\}, \sigma, \lambda)$ is an n-state autonomous machine which satisfies the following conditions:

1) State S_i corresponds to vertex v_i for $1 \le i \le n$.
2) Let $x_1^{(i)}x_2^{(i)} \ldots x_p^{(i)}$ be a binary number assigned to vertex v_i which corresponds to state S_i. The next-state function δ and the output function λ are defined as:

$$\delta(S_i, \epsilon) = S_{i+1} \qquad (1 \le i \le n-1)$$

$$\delta(S_n, \epsilon) = S_1$$

$$\lambda(S_i, \epsilon) = 0 \quad \text{if} \quad x_p^{(i)} = 0$$

$$= 1 \quad \text{if} \quad x_p^{(i)} = 1 \qquad (1 \le i \le n)$$

From this definition, it is obvious that any sequence of p consecutive inputs ϵ^p generates an output sequence which is a binary number corresponding to the initial state. Thus machine A_n has a distinguishing sequence ϵ^p, where $p = \lceil \log_2 n \rceil$. The checking experiment contains the following sequences:

Input: ϵ^p
State: S_i

for all states S_i.

Thus the total checking experiment is a sequence of $n - 1 + [\log_2 n]$ consecutive inputs $\epsilon^{n-1+[\log_2 n]}$ which is the optimal checking experiment.

Example: The autonomous machine A_4 shown in Fig. 6 is obtained from the de Bruijn graph of order 2 shown in Fig. 5. For machine A_4, $\epsilon\epsilon$ is both a homing sequence and a distinguishing sequence. Hence by applying $\epsilon\epsilon$ and observing the output response, we can establish the initial state. Suppose the initial state was S_1, then the total checking experiment is:

Input:	ϵ	ϵ	ϵ	$\epsilon\epsilon$
State:	S_1	S_2	S_3	S_4
Output:	0	0	1	10

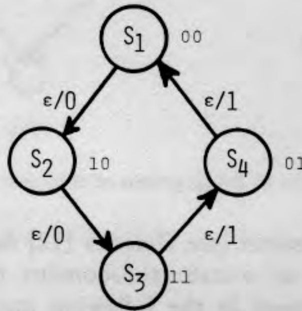

Fig. 6 Autonomous machine A_4

In order to augment a given machine M to a machine M^* having an optimal checking experiment, we have only to add machine A_n to M as shown in Fig. 7. Let $M = (S, I, O, \delta, \lambda)$ by a given machine, where $S = \{S_1, S_2, \ldots, S_n\}$, $I = \{I_1, I_2, \ldots, I_m\}$ and $O = \{O_1, O_2, \ldots, O_l\}$. Then we can give a procedure for augmenting the given machine M so that the augmented machine M^* has an optimal checking experiment.

Augmentation Procedure:

1) Find a cycle $[v_1, v_2, \ldots, v_n]$ of length n in the de Bruijn graph of order $p = \lceil \log_2 n \rceil$, and let each state S_i correspond to a vertex v_i.
2) Add a new input symbol ϵ to M. The next-state function δ and the output function λ for the new input symbol ϵ are defined as:

$$\delta(S_i, \epsilon) = S_{i+1} \quad \text{for} \quad 1 \le i \le n-1$$

$$\delta(S_n, \epsilon) = S_1$$

$$\lambda(S_i, \epsilon) = O_1 \quad \text{if} \quad x_p^{(i)} = 0$$

$$= O_2 \quad \text{if} \quad x_p^{(i)} = 1$$

where $x_1^{(i)} x_2^{(i)} \ldots x_p^{(i)}$ denotes a binary number assigned to vertex v_i corresponding to state S_i.

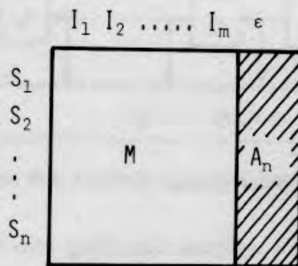

Fig. 7 Augmented machine M^*

Example: Consider machine M_2 given by Fig. 4. Since machine M_2 has four states, we consider the de Bruijn graph of order 2. Then we obtain a cycle of length 4[(00), (10), (11), (01)]. Let states A, B, C and D correspond to vertices (00), (10), (11) and (01), respectively. Then we obtain the augmented machine M_2* shown in Fig. 8. M_2* has an optimal checking experiment $\epsilon\epsilon\epsilon\epsilon\epsilon$.

	0	1	ε
A	B , 0	D , 0	B , 0
B	A , 0	B , 0	C , 0
C	D , 1	A , 0	D , 1
D	D , 1	C , 0	A , 1

Fig. 8 Augmented machine M_2*

IV. HARDWARE IMPLEMENTATION

In Section III we have considered augmentation procedure at the functional level. In this section we consider, at the circuit level, modifying a given circuit so that the modified circuit has an optimal checking experiment of length $n - 1 + \lceil \log_2 n \rceil$.

Let us consider a binary p-stage feedback shift register (FSR) shown in Fig. 9. It consists of p clocked delay elements (stages) and a feedback logic producing a Boolean function $f(x_1, x_2, \ldots, x_p)$, where $x_i \in B = \{0, 1\}$ indicates the binary content of stage i. One of the well-known results is the existence of a p-stage FSR with a cycle of length k, for any $p \geq 1$ and any $1 \leq k \leq 2^p$. A cycle of length k corresponds to a closed sequence of k distinct vertices in the de Bruijn graph G_p. Let f_n be a feedback function which generates a maximum length ($n = 2^p$) cycle, and let C_n be a p-stage FSR realizing the feedback function f_n. Methods of finding such a feedback function are available [12][13].

It is obvious that the FSR C_n is a realization of an n-state autonomous machine A_n defined in Section III. In order to augment a given machine

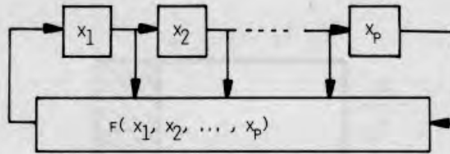

Fig. 9 Binary p-stage feedback shift register

M to a machine having an optimal checking experiment, we have only to embed the machine A_n in M. Therefore, to augment a given circuit C to a circuit having an optimal checking experiment, we have only to embed the FSR C_n in C, similarly.

Let us consider a circuit in which clocked delay flip-flops are used as the storage elements and the number of the states is a power of 2. Such a circuit may be regarded as one or more delay flip-flops connected into a combinational network, as shown in Fig. 10 for the case with two flip-flops. For such circuits, in order to embed the FSR C_n, let us modify the circuit design as shown in Fig. 11. One input, marked c in Fig. 11, has been added, and it controls the mode of the circuit. Switches have been inserted in each input lead of every flip-flop. When c is supplied with a 0, the circuit enters the normal mode in which the circuit operates exactly as it did before modifications were carried out. When c is supplied with a 1, the circuit enters a second mode in which the flip-flops are connected in a chain, so that they behave as a feedback shift register.

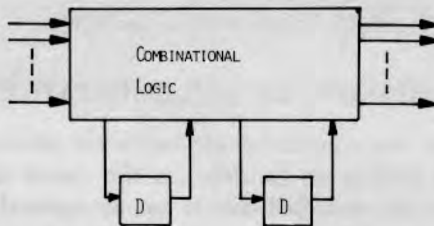

Fig. 10 Circuit with colocked delay elements

The checking experiment for a circuit that has been modified is any sequence of $n - 1 + \lceil \log_2 n \rceil$ consecutive inputs with $c = 1$ (in the feedback shift register mode). This sequence corresponds to a sequence of $n - 1 + \lceil \log_2 n \rceil$ consecutive inputs $\epsilon^{n-1+\lceil \log_2 n \rceil}$ which is an optimal checking experiment of the augmented machine in Section III. Note that these checking experiments are *preset*. Therefore, the procedure for testing stable faults in a modified circuit is very easy and simple as follows:

1) Switch to feedback shift register mode.
2) Advance a clock $n - 1 + \lceil \log_2 n \rceil$ times, applying an arbitrary input pattern with $c = 1$.

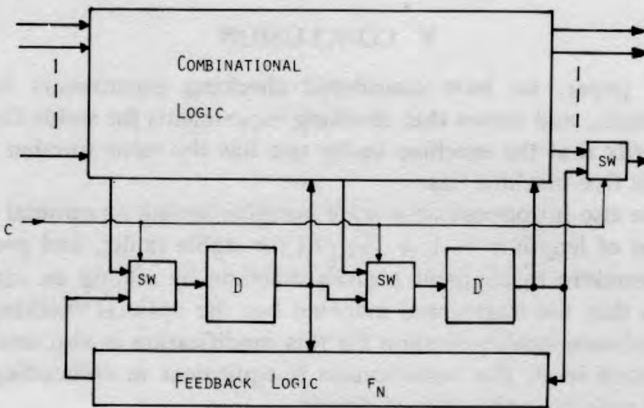

Fig. 11 Modified circuit

A similar modification scheme by Williams and Angell [14] can be also used, as shown in Fig. 12 for the case with two flip-flops. This modification needs less additional logic than the modification shown in Fig. 11. However, the modified circuit shown in Fig. 12 has no optimal checking experiment of length $n - 1 + \lceil \log_2 n \rceil$ as follows. The checking experiment consists of two parts. The first part is the initializing part which brings the circuit under test to the starting state for the experiment. The second part carries the circuit through all its states. The length of the first part is $\lceil \log_2 n \rceil$ and the length of the second part is $n - 1 + \lceil \log_2 n \rceil$. Two parts of the checking experiment can not be overlapped with each other for the circuit shown in Fig. 12. Hence, the length of the total checking experiment is $n - 1 + 2\lceil \log_2 n \rceil$ which is not optimal.

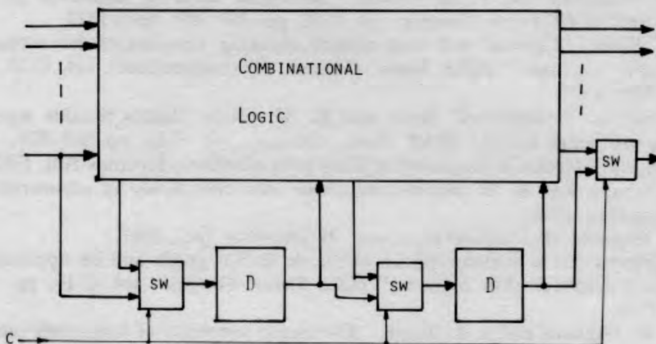

Fig. 12 Modification by Williams and Angell

V. CONCLUSION

In this paper, we have considered checking experiments for stable memory faults, and shown that checking experiments for stable faults need only to verify that the machine under test has the same number of states as the fault free machine has.

We have also introduced an n-state machine having an optimal checking experiment of length $n - 1 + \lceil \log_2 n \rceil$ for stable faults, and presented a design procedure to augment a given machine by adding an extra input symbol so that the augmented machine has the optimal checking experiment. Hardware implementation for this modification is also investigated. At the circuit level, this modification is equivalent to embedding a feedback shift register in the original circuit.

ACKNOWLEDGMENT

The authors wish to acknowledge the support and encouragement of Professor Hiroshi Ozaki of Osaka University.

REFERENCES

[1] J. F. Meyer, "Fault tolerant sequential machines," *IEEE Trans. Comput.*, vol. C-20, pp. 1167–1177, Oct. 1971.
[2] R. Boute and E. J. McCluskey, "Fault equivalence in sequential machines," *Sympo. on Computers and Automata*, Polytechnic Institute of Brooklyn, pp. 483–506, April 1971.
[3] F. C. Hennie, "Fault detecting experiments for sequential circuits," *Proc. 5th Ann. Sympo. on Switching Circuit Theory and Logical Design*, Princeton, N.J., pp. 95–110, Nov. 1964.
[4] G. Gonenc, "A method for the design of fault detection experiments," *IEEE Trans. Comput.* (Short Notes), vol. C-19, pp. 551–558, June 1970.
[5] E. P. Hsieh, "Checking experiments for sequential machines," *IEEE Trans. Comput.*, vol. C-20, pp. 1152–1166, Oct. 1971.
[6] D. E. Farmer, "Algorithms for designing fault-detection experiments for sequential machines," *IEEE Trans. Comput.*, vol. C-22, pp. 159–167, Feb. 1973.
[7] A. D. Friedman and P. R. Menon, "Restricted checking sequences for sequential machines," *IEEE Trans. Comput.*, vol. C-22, pp. 397–399, April 1973.
[8] R. T. Boute, "Optimal and near-optimal checking experiments for output faults in sequential machines," *IEEE Trans. Comput.* (Correspondence), vol. C-23, pp. 1207–1213, Nov. 1974.
[9] H. Fujiwara, Y. Nagao, T. Sasao and K. Kinoshita, "Easily testable sequential machines with extra inputs," *IEEE Trans. Comput.*, vol. C-24, pp. 821–826, Aug. 1975.
[10] A. Gill, *Introduction to the theory of finite-state machines*, McGraw-Hill, 1962.
[11] J. Hartmanis and R. E. Stearns, *Algebraic structure theory of sequential machines*, Prentice-Hall, 1966.
[12] S. W. Golomb, *Shift register sequences*, Holden-Day, Inc., 1967.
[13] A. Lempel, "On a homomorphism of the de Bruijn graph and its applications to the design of feedback shift registers," *IEEE Trans. Comput.*, vol. C-19, pp. 1204–1209, Dec. 1970.
[14] M. J. Y. Williams and J. B. Angell, "Enhancing testability of large-scale integrated circuits via test points and additional logic," *IEEE Trans. Comput.*, vol. C-22, pp. 46–60, Jan. 1973.