Simple Bounds on Serial Signature Analysis Aliasing for Random Testing

Nirmal R. Saxena, Piero Franco, and Edward J. McCluskey, *Fellow, IEEE*

***Abstract*—It is shown that the aliasing probability is bounded above by $(1+\epsilon)/L \approx 1/L$ (ϵ small for large L) for test lengths L less than the period, L_c, of the signature polynomial; for test lengths L that are multiples of L_c, the aliasing probability is bounded above by 1; and, for test lengths L greater than L_c and not a multiple of L_c, the aliasing probability is bounded above by $2/(L_c+1)$. These simple bounds avoid any exponential complexity associated with the exact computation of the aliasing probability. Simple bounds also apply to signature analysis based on any linear finite state machine (including linear cellular automaton).**

From these simple bounds it follows that the aliasing probability in a signature analysis design using β intermediate signatures is bounded by $((1+\epsilon)^\beta \beta^\beta)/L^\beta$, for $\beta < L$ and $L/\beta < L_c$. By using intermediate signatures the aliasing probability can be substantially reduced.

***Index Terms*— Dual codes, linear codes, signature analysis, weight distribution.**

I. Introduction

SIGNATURE analysis is a widely used data compaction method for built-in self test in VLSI. In data compaction, reduced response data are used for checking testing results. Signature analysis is a compaction method based on linear feedback shift registers. Compaction methods achieve substantial reduction of the response data; however, this reduction is achieved at the expense of fault coverage. Aliasing causes loss of effective fault coverage.

The important issues in signature analysis design are: the signature register size r; the feedback connections for the signature register (feedback connections are determined by the signature polynomial $U(X)$), the test length L, and the loss of coverage due to aliasing. One way to address these issues is to include signature analysis in the fault simulation experiments. But, this precludes fault dropping (because the entire test length L has to be applied to check the signature) and increases the simulation time substantially. The simulation experiments do not provide any design insight for feedback connections of the signature register. Also, the experimentally estimated aliasing probability holds good only for the modeled faults.

An alternative to fault simulation is to express the faulty behavior of the CUT in terms of an error model. In the past [15], [16], equally likely errors, single bit errors, and burst errors were some of the error characteristics that have been assumed. In particular cases, it may be possible to justify the use of these models; however, the general applicability of these models for error characteristics in VLSI circuits, seems questionable. The Bernoulli model [13] has been widely used by several researchers. This model is reasonable for combinational circuits with the restriction that random test patterns are applied, and the faults do not cause sequential behavior. In the Bernoulli model, output errors are assumed to occur with probability p in the presence of a fault, and these output errors are independent events. The probability p for a fault is called the *detection probability* [7].

To address signature analysis design issues, an exact formula for the aliasing probability as a function of parameters: r, $U(X)$, L, and p can be derived. Using the exact formula, various parameters can be examined to obtain optimum signature analysis designs. This is easier said than done because the complexity associated with the exact analysis is exponential. The calculations of p and the aliasing probability are NP-hard problems [10]. Exact analysis is infeasible for practical design parameters. A feasible approach is to derive upper bounds on the aliasing probability. At design time, the designer has the freedom to choose the signature polynomial $U(X)$, the signature register size r, and the test length L; however, the designer has uncertainty about what faults occur in the circuit. The value of p is restricted by the nature of faults (depending on the fault, the value of p can be anywhere between 0 and 1 [11]). Even if the nature of faults is known the complexity of calculating p is exponential at worst. Bounds on the aliasing probability that depend on L and the signature register design (defined by $U(X)$) and that are independent of p will help the designer. In this paper, such bounds are called *simple bounds*. Simple bounds avoid the exponential complexity associated with the analysis of various design choices. Simple bounds reported in this paper are shown to hold for any linear finite state machine, including cellular automaton, implementation of signature analysis.

II. Simple Bounds: Motivation

Algorithms to compute exact aliasing probability have been presented in [5], [10], and [17]. The algorithm presented in [17] calculates aliasing probability for primitive polynomials,

Manuscript received July 5, 1991; revised December 11, 1991. This work was supported in part by the Hewlett-Packard Resident Fellowship, in part by the National Science Foundation under Grant MIP-8709128, and in part by the Innovative Science and Technology Office of the Strategic Defense Initiative Organization administered through the Office of Naval Research under Contract N00014-85-K-0600.

The authors are with the Center for Reliable Computing ERL 460, Computer Systems Laboratory, Departments of Electrical Engineering and Computer Science, Stanford University, Stanford, CA 94305-4055.

IEEE Log Number 9108207.

0018-9340/92$03.00 © 1992 IEEE

for test lengths L less than $2^r - 1$. This requires $O(r2^r)$ time complexity and $O(2^r)$ space complexity for a particular value of test length $L < 2^r - 1$. To obtain aliasing probability values for an entire range of test length L, the algorithm presented in [17] would require $O(L2^r)$ time complexity and $O(2^r)$ space complexity. The algorithm presented in this paper and in [5] computes exact aliasing probability for all test lengths L and for all primitive and nonprimitive polynomials. The complexity required is $O(L2^r)$ and $O(L)$ in time and space, respectively. Table I shows the computation time for exact aliasing probability up to test length $L = 1000$. The entries in the last two rows ($r = 31$) are the projected time. The computation in Table I is based on the algorithm [Appendix] presented in this paper. The algorithms in [5] and [17] take comparable time.

The practical test lengths L (are of order 10^6 in Intel 486) [4] and practical register sizes ($r = 31$ in IBM RS/6000) [8] are large enough to make the exact aliasing computation infeasible.

III. Derivation of Simple Bounds

In this section, a simple bound is derived on the aliasing probability using the following approach. First, three closed-form upper bounds on the aliasing probability, $P_{al}(L, p, U(X))$, as a function of the detection probability p, the test length L, and the signature polynomial $U(X)$ are derived. Second, these bounds are combined to obtain a tight simple upper bound. In deriving the first two bounds only the period, L_c, of the signature polynomial $U(X)$ is used as its defining characteristic.

Definition 1: The period L_c of a polynomial $U(X)$ (without X as a factor) of degree r is the smallest positive L_c such that $U(X)$ divides $X^{L_c} + 1$. If $U(X)$ has X as a factor then the period L_c of this polynomial is the period of the largest degree factor $U_1(X)$ (without X as a factor) of $U(X)$. If $U(X)$ is primitive then $L_c = 2^r - 1$.

If $U(X)$ is nonprimitive then $L_c < 2^r - 1$.

Theorem 1: Given a polynomial $U(X)$ without X as a factor, with period L_c and a single output response corresponding to a random pattern test length L, then for $L < L_c$, $P_{al}(L, p, U(X))$ is bounded above by the function

$$f_1(L,p) = \frac{1-p^{L+1}}{(L+1)(1-p)} - \frac{(1-p)^L}{L+1} - \frac{Lp^2(1-p)^{L-2}}{2} - p(1-p)^{L-1} - p^L$$

and also bounded above by the function $f_2(L, p) =$

$$\frac{1-(1-p)^{L+1}}{(L+1)p} - \frac{p^L}{L+1} - \frac{Lp(1-p)^{L-1}}{2} - \frac{L(L-1)p^2(1-p)^{L-2}}{6} - (1-p)^L.$$

Proof: See [11].

$f_1(L, p)$ and $f_2(L, p)$ define two closed-form upper bounds on $P_{al}(L, p, U(X))$. For p close to zero, $f_1(L, p)$ is tighter than $f_2(L, p)$; however, for p close to one, $f_2(L, p)$ provides a tighter bound on $P_{al}(L, p, U(X))$. This is illustrated in Fig. 1.

TABLE I
Computation Times for Exact Aliasing on Sparc IPC Workstation

Polynomial degree r	Test Length L	Computation Time
9	1000	2 s
11	1000	8 s
12	1000	16 s
31	**1000**	**97 days**
31	**1000000**	**265 yrs**

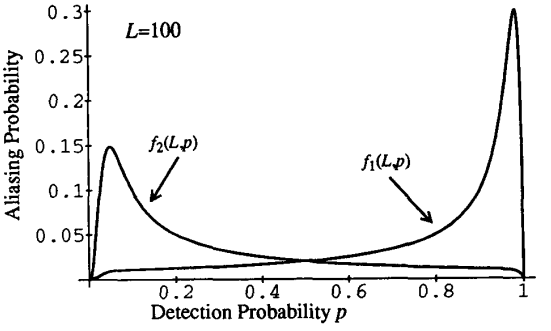


Fig. 1. Plot of bounds f_1 and f_2 as a function of p.

The plots in Fig. 1 are for test length $L = 100$. Although for all practical purposes, $L = 100$ is a small test length it does illustrate the nature of $f_1(L, p)$ and $f_2(L, p)$ as a function of p. These two bounds hold good for test lengths L less than L_c. Bounds $f_1(L, p)$ and $f_2(L, p)$ are not tight in neighborhood of $p = 0.5$. Another bound $f_4(L, p)$[1] is derived based on dual codes in this paper and this bound is tighter in the neighborhood of $p = 0.5$. By combining bounds $f_1(L, p)$, $f_2(L, p)$, and $f_4(L, p)$ a tighter simple bound on the aliasing probability is derived. In the following sections bound $f_4(L, p)$ is established.

A. Bounds Using Dual Codes

The respective weight enumerators N_w and M_w of a $(L, L-r)$ linear code and its dual, (L, r) code, are related by (1) [1] (McWilliams Identity). Here $|\zeta| = 2^{L-r}$ is the number of codewords in the $(L, L-r)$ linear code.

$$\sum_{w=0}^{L} N_w (x+y)^{L-w}(x-y)^w = |\zeta| \sum_{w=0}^{L} M_w x^{L-w} y^w. \quad (1)$$

Substituting $x - y = p$ and $x + y = 1 - p$, we have

$$\sum_{w=0}^{L} N_w (1-p)^{L-w} p^w = 2^{-r} \sum_{w=0}^{L} M_w (1-2p)^w. \quad (2)$$

The left-hand side of (2) is the expression for the aliasing probability [10], $P_{al}(L, p, U(X))$, including the error-free case. The above equation suggests that the weight distribution of dual codes can be used to compute the exact aliasing probability. The dual code formulation of calculating exact aliasing probability has been used earlier [17], [6].

[1] $f_4(L, p)$ is used because $f_3(L, p)$ has been used in [11] for Ivanov's Bound.

B. Parity Check Matrix

The parity check matrix, H, of the code generated by $U(X)$ can be constructed using the modular form *linear feedback shift register* (LFSR). The columns in matrix H are generated by successive states of LFSR. The initial state of the LFSR is the column vector $[1\ 0\ 0\ \cdots 0]^T$. The number of columns generated are L. Fig. 2. illustrates the generation of parity check matrix for polynomial $X^3 + X + 1$, for length $L = 6$ code.

The next state in an r stage LFSR is a linear transformation of the previous state and this linear transformation is characterized by an $r \times r$ matrix A. The characteristic polynomial of the matrix is $U(X)$. For example, the matrix A for the LFSR in Fig. 2 is given by

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \tag{3}$$

By the nature of construction of the parity check matrix the first r columns in H form an $r \times r$ identity matrix I. It is easy to show that columns $r+1$ through $2r$ in H form an $r \times r$ submatrix that is same as A^r. For example, in Fig. 2

$$A^3 = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}. \tag{4}$$

This is the submatrix formed by columns 4, 5, and 6 in H (Fig. 2). In general, any $r \times L$ parity check matrix, H, generated by polynomial $U(X)$ can be compactly represented by the following equation, where I is the $r \times r$ identity matrix and the powers of $r \times r$ submatrix A represent successive blocks of r columns in H. If L is not a multiple of r, then there will be remaining $L - \lfloor \frac{L}{r} \rfloor$ columns in H that are not described by powers of A.

$$H = \left[I\ A^r\ A^{2r} \cdots A^{\lfloor \frac{L}{r} \rfloor r} \cdots \right]. \tag{5}$$

Theorem 2: The minimum Hamming distance of the dual code formed by the polynomial $U(X)$, without X as a factor, is greater than or equal to $\lfloor L/r \rfloor$.

Proof: The dual code defined by $U(X)$ is generated by using the parity check matrix H of $U(X)$ as the generator matrix. Let A be the $r \times r$ matrix that represents the next state linear transformation of the states of the LFSR defined by $U(X)$. Since we are given that $U(X)$ does not have X as a factor this implies that matrix A is nonsingular. This fact follows from the property that $U(X)$ is the characteristic polynomial of A. This further implies that any power of A is also nonsingular. The dual code is generated by the row-space of matrix H. Therefore, the minimum Hamming distance of the dual code is the minimum nonzero weight of the code generated by H. Using the (5) formulation of the matrix H, the dual code can be thought of as a concatenated code generated by submatrices $I, Ar, \cdots, A^{r\lfloor \frac{L}{r} \rfloor}$. The minimum Hamming distance $d_{min} = d(H)$ is given by

$$\begin{aligned} d_{\min} = d(H) \geq d(I) + d(A^r) + d(A^{2r}) \\ + \cdots + d\left(A^{r\lfloor \frac{L}{r} \rfloor}\right) + \cdots. \end{aligned} \tag{6}$$

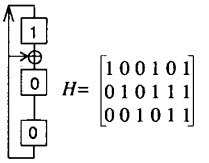


Fig. 2. Parity check matrix for $U(X) = X^3 + X + 1$.

Since the matrix A is nonsingular the minimum distance, $d(A^i)$, for any i, of the code generated by any power of A is at least one. Therefore, we have

$$d_{\min} = d(H) \geq \lfloor \frac{L}{r} \rfloor. \tag{7}$$

C. Partitioning Codewords Generated by H

The set of codewords generated by H can be partitioned into $r+1$ distinct classes $C_0, C_1, \cdots, C_r$. Class C_i has codewords such that they have i ones in the first r bits. The first r bits of the codewords are generated by the identity submatrix I in H. Therefore, there are $\binom{r}{i}$ (i.e., all linear combinations of rows in H by picking only r rows) codewords in class C_i. From the minimum Hamming distance property, codewords in C_i, $i > 0$, will have weights greater than or equal to $\lfloor L/r \rfloor + i - 1$. Let class C_i be partitioned into disjoint sets according to the weights of the codewords. Let $B_{i,j}$ be the number of codewords in C_i having weight $\lfloor L/r \rfloor + j + i - 1$. Clearly,

$$\sum_{w=0}^{L} M_w = \sum_{i=0}^{r} |C_i| = \sum_{i=0}^{r} \sum_{j \geq 0} B_{i,j} = 2^r. \tag{8}$$

The aliasing probability formula can be rewritten as

$$\begin{aligned} P_{al}(L,p,U(X)) = 2^{-r} \sum_{i=1}^{r} \sum_{j \geq 0} B_{i,j} (1-2p)^{\lfloor \frac{L}{r} \rfloor + i + j - 1} \\ + 2^{-r} - (1-p)^L. \end{aligned} \tag{9}$$

D. Dual Code Bound $f_4(L,p)$ on Aliasing Probability

Using (9) we derive upper bounds on the aliasing probability. We have the following inequality:

$$\begin{aligned} P_{al}(L,p,U(X)) \leq 2^{-r} \sum_{i=1}^{r} \left\{ |(1-2p)|^{\lfloor \frac{L}{r} \rfloor + i + j - 1} \right. \\ \left. \cdot \sum_{j \geq 0} B_{i,j} \right\} + 2^{-r} - (1-p)^L. \end{aligned} \tag{10}$$

This is because $|(1-2p)|^{\lfloor \frac{L}{r} \rfloor + i - 1} \geq (1-2p)^{\lfloor \frac{L}{r} \rfloor + i + j - 1}\ \forall\ (j \geq 0)$.

Since $\sum_{j \geq 0} B_{j,i} = \binom{r}{i}$ we have

$$\begin{aligned} P_{al}(L,p,U(X)) \leq 2^{-r} \sum_{i=1}^{r} |(1-2p)|^{\lfloor \frac{L}{r} \rfloor + i - 1} \binom{r}{i} \\ + 2^{-r} - (1-p)^L. \end{aligned} \tag{11}$$

After some algebra, we have bound, $f_4(L,p)$, on the aliasing probability $P_{al}(L,p,U(X))$

$$f_4(L,p) = |1-2p|^{\lfloor \frac{L}{r} \rfloor - 1}((1+|1-2p|)^r - 1)2^{-r} + 2^{-r} - (1-p)^L. \quad (12)$$

IV. Combining Bounds f_1, f_2, and f_4

First we replace, functions f_1, f_2, and f_4 by three monotone functions g_1, g_2, g_4, respectively. These monotone functions bound f_1, f_2, and f_3 as follows:

$$f_1(L,p) < g_1(L,p) = \frac{1}{(L+1)(1-p)}$$

$$f_2(L,p) < g_2(L,p) = \frac{1}{(L+1)p}$$

$$f_4(L,p) \le g_4(L,p) = |1-2p|^{\lfloor \frac{L}{r} \rfloor - 1}((1+|1-2p|)^r - 1)2^{-r} + 2^{-r}.$$

A simple bound on the aliasing probability for test lengths L less than L_c is derived by taking the maxima of the function $\min\{g_1, g_2, g_4\}$ over the entire range [0,1] of p. For a given L the function $g_4(L,p)$ monotonically decreases as p goes from 0 to 0.5, and monotonically increases as p goes from 0.5 to 1.

For small values of p, the min function is dominated by $g_1(L,p)$. For mid-values of p, $g_4(L,p)$ dominates the min function. Function $g_2(L,p)$ dominates the min function for large values of p. There exist p_{small} and p_{large} (as illustrated in Figs. 3 and 4) $0 < p_{small} < p_{large} < 1$, such that:

1) $g_1(L,p) \le g_4(L,p)$, for $p \le p_{small}$
2) $g_4(L,p) < g_1(L,p)$, and $g_4(L,p) < g_2(L,p)$, for $p_{small} < p < p_{large}$
3) $g_2(L,p) \le g_4(L,p)$, for $p \ge p_{large}$.

The values p_{small} and p_{large} can be calculated by solving equations $g_1(L,p) = g_4(L,p)$ and $g_2(L,p) = g_4(L,p)$, respectively. It follows from the behavior of these functions that a simple bound on the aliasing probability is the maximum of $g_1(L,p_{small})$ and $g_2(L,p_{large})$. Solving for the exact values of p_{small} and p_{large} is a difficult problem. However, in order to derive simple bounds on the aliasing probability it is sufficient to derive an upper bound on p_{small} and a lower bound on p_{large}. This is because $g_1(L,p_{small}) = 1/((1+L)(1-p_{small}))$ and $g_2(L,p_{large}) = 1/((1+L)p_{large})$.

Using standard numerical techniques, bounds on p_{small} and p_{large} can be derived. Clearly, the bounds on p_{small} and p_{large} depend on L and r.

Let us define $\epsilon(L,r) = \max\{(1-p_{small})^{-1}, (p_{large})^{-1}\} - 1$. $\epsilon(L,r)$ is small for large values of test length L. It follows from the definition of $\epsilon(L,r)$ that $1+\epsilon(L,r) = \max\{(1-p_{small})^{-1}, (p_{large})^{-1}\}$. By solving the following equation (for closed-form bounds on p_{small} and p_{large}) a closed-form upper bound on $\epsilon(L,r)$ can be derived.

$$|1-2p|^{\lfloor \frac{L}{r} \rfloor} + 2^{-r} = \frac{1}{L+1}.$$

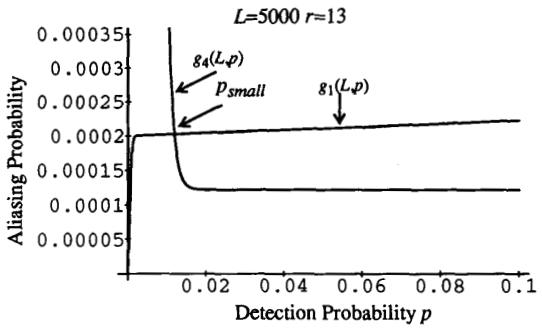


Fig. 3. Plot of bounds g_1 and g_4 as a function of p.

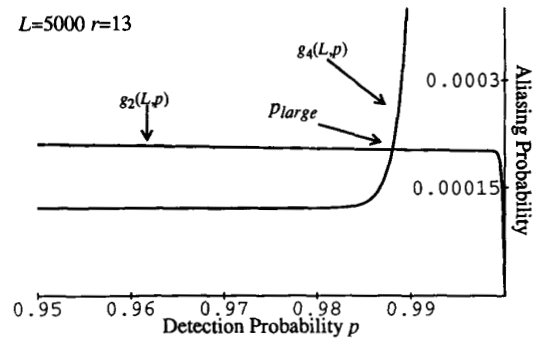


Fig. 4. Plot of bounds g_4 and g_2 as a function of p.

A closed-form upper bound on p_{small} is

$$p_{small} < \frac{1}{2}\left(\frac{\sqrt[\lfloor \frac{L}{r} \rfloor]{2^r(L+1)} - \sqrt[\lfloor \frac{L}{r} \rfloor]{2^r - (L+1)}}{\sqrt[\lfloor \frac{L}{r} \rfloor]{2^r(L+1)}}\right).$$

and a closed-form lower bound on p_{large} is

$$p_{large} > 1 - \frac{1}{2}\left(\frac{\sqrt[\lfloor \frac{L}{r} \rfloor]{2^r(L+1)} - \sqrt[\lfloor \frac{L}{r} \rfloor]{2^r - (L+1)}}{\sqrt[\lfloor \frac{L}{r} \rfloor]{2^r(L+1)}}\right).$$

Therefore, a closed-form bound on $\epsilon(L,r)$ is

$$\epsilon(L,r) < 2\frac{\sqrt[\lfloor \frac{L}{r} \rfloor]{2^r(L+1)}}{\sqrt[\lfloor \frac{L}{r} \rfloor]{2^r(L+1)} - \sqrt[\lfloor \frac{L}{r} \rfloor]{2^r - (L+1)}} - 1.$$

Table II compares the value of ϵ calculated (listed as exact in the table) by using numerical techniques (bisection method) and the value obtained from the closed form bound. It is clear from the table that ϵ values are small and also that the closed-form upper bound on ϵ is reasonably tight.

It was already shown that a simple bound on the aliasing probability is $\max\{g_1(L,p_{small}), g_2(L,p_{large})\} \le \max\{(1-p_{small})^{-1}, (p_{large})^{-1}\}1/(L+1) = (1+\epsilon(L,r))/(L+1)$. This is almost $1/L$ because ϵ is very small compared to 1. This refined simple bound holds good for test lengths $L < L_c$.

Thus far we have derived a simple bound on the aliasing probability for test length less than L_c. For test lengths $L \ge L_c$, we use the simple bounds derived in [10]. Table III summarizes these simple bounds.

TABLE II
COMPARISON OF EXACT $\epsilon(L,r)$ AND ITS CLOSED-FORM UPPER BOUND

L, r	Exact	Bound
$10^3, 16$	47×10^{-3}	55.7×10^{-3}
$8\times10^3, 16$	8.9×10^{-3}	9.1×10^{-3}
$10^4, 14$	6.9×10^{-3}	7.1×10^{-3}

TABLE III
SIMPLE BOUNDS SUMMARY

$L < L_c$	$L = hL_c$	$L > L_c$ and $L \neq hL_c$
$\frac{1+\epsilon}{L}$	1	$\frac{2}{L_c+1}$

V. APPLICATION TO LINEAR FINITE STATE MACHINES

The simple bounds derived in this paper apply to any linear finite state machine implementation of signature analysis. A linear finite state machine is a finite state machine in which the next state is determined by a linear combination of the present input and a fixed linear transformation of the present state. In this paper our interest is in linear operations over GF(2). Algebraically, a linear finite state machine can be characterized by the equation $Y_{i+1} = \boldsymbol{A}Y_i + z_i\boldsymbol{D}$. Y_{i+1}, Y_i are column vectors of dimension r denoting the next state and the present state respectively. $\boldsymbol{A}$ is an $r \times r$ matrix representing the linear transformation. The scalar variable z_i is the binary serial input at time i. $\boldsymbol{D}$ is a column vector of dimension r which defines the manner in which the serial input is fed into the state machine. For example, in Fig. 5, the defining parameters in the linear finite state machine are

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}; \quad \boldsymbol{D} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}; \quad r = 3.$$

The serial signature analysis for any general linear finite state machine can be easily instrumented as follows: the serial input to the finite state machine is provided by the serial output of the combinational circuit under test. Upon application of test responses, the state machine undergoes state transitions starting initially with a fixed state Y_0 (usually all zero). The final state Y_L after an application of a length L test is the signature of the circuit under test. This is the most general definition of the signature. The polynomial division definition [10] applies only to modular LFSR's. The LFSR implementation of signature analysis discussed in the foregoing sections is one particular implementation of a linear finite state machine. One of the reasons why LFSR's are popular is because they can be efficiently implemented (Fig. 6). Also, in addition to the apparatus of linear algebra the well-studied [1] GF(2) polynomial algebra is useful in analyzing the aliasing behavior in LFSR's. Associated with every matrix $\boldsymbol{A}$ is a characteristic polynomial defined by the determinant of the matrix $\boldsymbol{A} + \boldsymbol{IX}$, where $\boldsymbol{I}$ is the identity matrix. In the case of LFSR implementation, this characteristic polynomial is the same as the signature polynomial $U(X)$. It

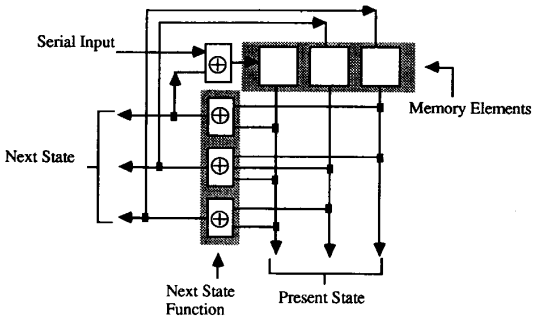


Fig. 5. Linear finite state machine based signature analysis.

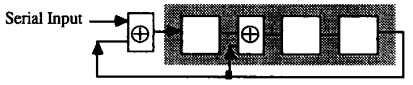


Fig. 6. LFSR implementation of linear finite state machine.

is possible for two different linear finite state machines to have the same characteristic polynomial. For example, the characteristic polynomial for both the Fig. 5 and the Fig. 6 implementations is $X^3 + X + 1$. Matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ that have the same characteristic polynomial are said to be *similar*.

Another compact implementation of a linear finite state machine is the linear cellular automaton [12]. In [12] the equivalence of linear cellular automata and linear feedback shift registers with respect to aliasing behavior has been demonstrated. However, this equivalence was restricted to linear finite state machines based on irreducible characteristic polynomials. In this work, we extend this equivalence to include linear finite state machines based on reducible characteristic polynomials. The following theorem proves the equivalence, with respect to aliasing probability, of two signature register implementations based on two similar matrices.

Theorem 3: If $\boldsymbol{A}$ and $\boldsymbol{B}$ are two nonsingular and similar $r \times r$ matrices in GF(2) then the two signature register implementations based on $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively, have the same aliasing probability.

Proof: It is sufficient to prove that $\boldsymbol{A}$ and $\boldsymbol{B}$ have the same aliasing error responses. Since $\boldsymbol{A}$ and $\boldsymbol{B}$ are similar there exists a nonsingular matrix $\boldsymbol{P}$ such that $\boldsymbol{A} = \boldsymbol{P}^{-1}\boldsymbol{BP}$. Without any loss in generality assume that $\boldsymbol{A}$ is in the matrix form corresponding to a modular LFSR with characteristic polynomial $U(X)$.

Lemma: The error polynomial $E(X)$ is a multiple of $U(X)$ if and only if $\boldsymbol{E}(\boldsymbol{A})$ is an all zero matrix.

Proof of Lemma: Let $E(X)$ be the error response polynomial that is a multiple of $U(X)$. If the serial input is fed into the jth stage of the modular LFSR corresponding to $U(X)$ then the signature of $E(X)$ is $(X^{j-1}E(X)) \bmod U(X)$. Since $\boldsymbol{A}$ is a nonsingular matrix, $U(X)$ will not have X as a factor, this implies that $X^{j-1}E(X)$ is a multiple of $U(X)$ if and only if $E(X)$ is a multiple of $U(X)$. Therefore, the signature of $E(X)$ is zero independent of what stage the serial input

is fed into. In matrix terms, for all j, the column vector $Y_L = \boldsymbol{E}(\boldsymbol{A})\boldsymbol{I}_j$ is zero if and only if $E(X)$ is a multiple of $U(X)$, where $\boldsymbol{E}(\boldsymbol{A})$ is the matrix polynomial and $\boldsymbol{I}_j$ is the unit column vector with 1 in the jth row. Also, $\boldsymbol{E}(\boldsymbol{A})\boldsymbol{I}_j$ is a zero column vector for all j if and only if the matrix $\boldsymbol{E}(\boldsymbol{A})$ is all zero. This leads us to the assertion that the error polynomial $E(X)$ is a multiple of $U(X)$ if and only if $\boldsymbol{E}(\boldsymbol{A})$ is an all zero matrix. If $\boldsymbol{E}(\boldsymbol{A})$ is an all zero matrix it implies that $\boldsymbol{P}\boldsymbol{E}(\boldsymbol{A})\boldsymbol{P}^{-1}$ is also an all zero matrix; but, $\boldsymbol{P}\boldsymbol{E}(\boldsymbol{A})\boldsymbol{P}^{-1} = \boldsymbol{E}(\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{-1}) = \boldsymbol{E}(\boldsymbol{B})$, therefore $\boldsymbol{E}(\boldsymbol{B})$ is an all zero matrix. Since $\boldsymbol{P}$ is a nonsingular matrix we have $\boldsymbol{B} = \boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{-1}$. We can also show that if $\boldsymbol{E}(\boldsymbol{B})$ is an all zero matrix then $\boldsymbol{E}(\boldsymbol{A})$ is an all zero matrix. This leads us to the result: $E(X)$ is a multiple of $U(X)$ if and only if for all matrices $\boldsymbol{B}$ similar to $\boldsymbol{A}$, $\boldsymbol{E}(\boldsymbol{B})$ is an all zero matrix. In other words, an error response that produces a zero signature in the signature register corresponding to matrix $\boldsymbol{A}$ will also produce a zero signature in the signature register corresponding to matrix $\boldsymbol{B}$.

Now we will show that an error response that produces a nonzero signature in the signature register corresponding to matrix $\boldsymbol{A}$ will also produce a nonzero signature in the signature register corresponding to matrix $\boldsymbol{B}$. Let us start with error polynomial $E(X)$ that has a nonzero signature in the matrix $\boldsymbol{A}$ implementation of the signature register. In the polynomial division sense, the signatures of $E(X), XE(X), \cdots, X^{r-1}E(X)$ are all nonzero and distinct. If they are not distinct then it leads us to the contradiction that the period L_c of $U(X)$ is less than r. For a degree r polynomial $U(X)$ without X as a factor it is impossible to have the period L_c less than r. This implies that the r column vectors $\boldsymbol{E}(\boldsymbol{A})\boldsymbol{I}_j$ corresponding to all r distinct values of j are nonzero and distinct. This is only possible if and only if $\boldsymbol{E}(\boldsymbol{A})$ is a nonsingular matrix. It is a trivial fact to verify that all matrices similar to a nonsingular matrix are also nonsingular. This implies that $\boldsymbol{E}(\boldsymbol{B})$ is nonsingular, because $\boldsymbol{E}(\boldsymbol{B})$ is similar to $\boldsymbol{E}(\boldsymbol{A})$ by the similarity relation $\boldsymbol{E}(\boldsymbol{A}) = \boldsymbol{P}^{-1}\boldsymbol{E}(\boldsymbol{B})\boldsymbol{P}$. This implies that the column vector $\boldsymbol{E}(\boldsymbol{B})\boldsymbol{I}_j$ is nonzero which in turn means that an error response that produces a nonzero signature in the signature register corresponding to matrix $\boldsymbol{A}$ will also produce a nonzero signature in the signature register corresponding to matrix $\boldsymbol{B}$. This completes the proof because we have shown the error responses that alias in the signature register implementation of $\boldsymbol{A}$ are exactly the error responses (no more and no less) that alias in the signature register implementation of $\boldsymbol{B}$.

A. Application of Simple Bounds to $U(X)$ with X as a Factor

Simple bounds also apply to signature polynomials $U(X)$, with X as a factor, for certain restricted LFSR implementations of serial signature analysis. The restriction on LFSR implementations applies to the manner in which the serial output of the circuit under test is connected to the LFSR implementing $U(X)$. For a detailed discussion see [10].

VI. Conclusions

Simple bounds on the aliasing probability were presented. A useful guideline to the system designer would be to use a signature polynomial with period greater than the test length. For example, with test length $L = 10^6$, a primitive polynomial with degree greater than or equal to 20 guarantees the aliasing probability to be less than 0.0001%. The bounds presented in this paper apply to any linear finite state machine implementation of signature analysis and therefore include linear cellular automata. This result has practical importance because the system designer can first choose a characteristic polynomial with an appropriate period. After selecting a characteristic polynomial, the designer can then choose an efficient (from the standpoint of VLSI implementation) linear finite state machine representation that realizes this polynomial. The preferred linear finite state machine may be a cellular automaton because of its regular structure.

Using intermediate signatures (sometimes also called segmented signatures) [9] is a way to reduce aliasing probability. For test length L, if β intermediate signatures at uniform intervals of length L/β are used then it follows from the simple bounds that the aliasing probability is bounded above by $((1+\epsilon)^{\beta}\beta^{\beta})/L^{\beta}$, for $\beta < L$ and $L/\beta < L_c$. By using intermediate signatures the aliasing probability can be substantially reduced. The designer can choose an appropriate β value. Experimental results in [9] and [18] demonstrate the effectiveness of using intermediate signatures.

This paper covered simple bounds on serial signature analysis. Simple bounds have also been derived for multiple-input signature analyzers; however, the proof of these bounds requires a different combinatorial treatment. The results for multiple-input signature analyzers will be reported in a separate paper.

Appendix

The following pseudo-code represents the computation of the weight distribution of the dual code:

```
main ()
{
unsigned long r; /* signature register size */
unsigned long L; /* test length L*/
unsigned long *M; /* weight distribution of dual code */
unsigned long *H; /* parity check matrix */
unsigned long count,limit;
unsigned long mask,poly;
unsigned long i,j,k;
M = malloc((size_t) ((L+1)*sizeof(unsigned long)));
```

```
H = malloc((size_t) (L*sizeof(unsigned long)));
scanf("%lu",&poly);
mask = 1;
for (i=1; i < r; i++) {
mask = 2*mask;
}
M[0]=1;
for (i=1;i< =L;i++)
M[i]=0;
H[0]=0x1;
for (i=1;i< L;i++)
{
if ((H[i-1]&mask) >0)
H[i] = (((H[i-1] < < 1) ^ poly) & (2*mask-1));
else
H[i] = ((H[i-1] < < 1) & (2*mask-1));
}
for (i=1;i< 2^r;i++) /* order L2^r nested loops */
{
count=0;
for (j=0;j< L;j++)
count += parity(H[j]&i,r);
M[count] = M[count]+1;
}
```

REFERENCES

[1] E. R. Berlekamp, *Algebraic Coding Theory*, revised ed. Laguna Hills, CA: Aegean Park Press, 1984.

[2] M. Damiani, *et al.*, "An analytical model for the aliasing probability in signature analysis testing," *IEEE Trans. Comput.-Aided Design*, vol. 8, pp. 1133–1144, Nov. 1989.

[3] M. R. Garey and D. S. Johnson, *Computers and Intractability A Guide to the Theory of NP-Completeness*. San Francisco, CA: Freeman, 1978.

[4] P. Gelsinger *et al.*, "Computer aided design and built-in self-test on the i486TM CPU," in *Proc. ICCD*, Oct. 1989, pp. 199–201.

[5] A. Ivanov and V. K. Agarwal, "An analysis of the probabilistic behavior of linear feedback signature registers," *IEEE Trans. Comput.-Aided Design*, vol. 8, pp. 1074–1088, Oct. 1989.

[6] K. Iwasaki and N. Yamaguchi, "Design of signature circuits based on weight distribution of error-correcting codes," in *Proc. ITC*, Sept. 1990, pp. 779–785.

[7] E. J. McCluskey *et al.*, "Probability models for pseudorandom test sequences," *IEEE Trans. Comput.-Aided Design*, vol. 7, pp. 68–74, Jan. 1988.

[8] I. M. Ratiu and H. B. Bokoglu, "Pseudorandom built-in self-test methodology and implementation for the IBM RISC System/6000 processor," *IBM J. Res. Develop.*, vol. 34, no. 1, pp. 78–84, Jan. 1990.

[9] N. R. Saxena, "Test compression methods," M.S. thesis, Univ. Iowa, May 1984.

[10] N. R. Saxena, E. J. McCluskey, and P. Franco, "Bounds on signature analysis aliasing for random testing," Stanford CRC Tech. Rep., CRC TR #90–11, Dec. 1990.

[11] N. R. Saxena, P. Franco, and E. J. McCluskey, "Refined bounds on signature analysis aliasing for random testing," in *ITC91 Proc.*, Oct. 1991, pp. 818–827.

[12] M. Serra *et al.*, "The analysis of one-dimensional linear cellular automata and their aliasing properties," *IEEE Trans. Comput.-Aided Design*, vol. 9, pp. 767–778, July 1990.

[13] J. J. Shedletsky, "Random testing: Practicality vs. verified effectiveness," in *Proc. Seventeenth Annu. Int. Conf. Fault-Tolerant Comput.*, June 1977, pp. 175–179.

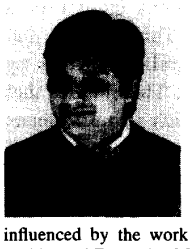
[14] T. W. Williams *et al.*, "Aliasing errors in signature analysis," *IEEE Design Test Comput.*, pp. 39–45, Apr. 1987.

[15] N. Benowitz, D. F. Calhoun, G. E. Alderson, J. E. Bauer, and C. T. Joeckel, "An advanced fault isolation system for digital logic," *IEEE Trans. Comput.*, vol. C-24, pp. 489–497, May 1975.

[16] R. A. Frohwerk, "Signature analysis: A new digital field service method," *Hewlett-Packard J.*, pp. 2–8, May 1977.

[17] D. K. Pradhan and S. K. Gupta, "A new framework for designing and analyzing BIST techniques and zero aliasing compression," *IEEE Trans. Comput.*, C-40, pp. 743–763, June 1991.

[18] I. Pomeranz, S. M. Reddy, and R. Tangirala, "On achieving zero aliasing for modeled faults," in *Proc. EDAC*, to be published.



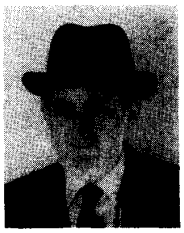
Nirmal R. Saxena received the B.E. degree (Electronics and Communications Engineering) from Osmania University, in 1982, the M.S. degree in electrical engineering from the University of Iowa in 1984, and the Ph.D. degree from Stanford University, in 1991.

From 1984 to 1991, he was with Hewlett-Packard. His Ph.D. at Stanford was supported by Hewlett-Packard's Honors Cooperative and Resident Fellowship programs. In 1991, he joined HaL Computer Systems. His research has been largely shaped and influenced by the work of his professors John P. Robinson, Sudhakär M. Reddy, and Edward J. McCluskey. His research interests include fault-tolerant computing, combinatorial mathematics, and computer architecture.

Dr. Saxena received a Gold Medal for topping the undergraduate level Mathematics Olympiad conducted by the Andhra Pradesh Mathematics Teachers Association.

Piero Franco received the B.Sc. degree in electrical engineering with distinction and the M.Sc. degree in engineering from the University of the Witwatersrand, Johannesburg, South Africa, in 1986 and 1989, respectively.

He is currently working towards the Ph.D. degree at the Center for Reliable Computing at Stanford University, Stanford, CA. His current research interest is in the design of reliable, testable systems, particularly delay fault testing and test response compaction.



Edward J. McCluskey (S'51–M'55–SM'59–F'65) received the A.B. degree (summa cum laude) in 1953 in mathematics and physics from Bowdoin College, the B.S. degree in 1953, the M.S. degree in 1953, and the Sc.D. degrees in 1956 in electrical engineering from M.I.T.

He worked on electronic switching systems at the Bell Telephone Laboratories from 1955 to 1959. In 1959, he moved to Princeton University, where he was Professor of Electrical Engineering and Director of the University Computer Center. In 1966, he joined Stanford University, where he is Professor of Electrical Engineering and Computer Science, as well as Director of the Center for Reliable Computing. He has published several books and book chapters. His most recent book is *Logic Design Principles with Emphasis on Testable Semicustom Circuits* (Englewood Cliffs, NJ: Prentice-Hall, 1986). Book chapters include *Design for Testability in Fault-tolerant Computing*, D. K. Pradhan, Ed., and chapters on Logic Design in the *Van Nostrand Reinhold Encyclopedia of Computer Science and Engineering* and in *Reference Data for Engineers*, E. C. Jordan, Ed. He is President of Stanford Logical Systems Institute which provides consulting services on fault-tolerant computing, testing, and design for testability. At Stanford University, he produces Ph.D's in computer engineering emphasizing testing and fault tolerance. His products include J. A. Abraham, E. Eichelberger, K. Parker, J. Savir, D. P. Siewiorek, K. Wagner, and many others too numerous to mention.

Dr. McCluskey served as the first President of the IEEE Computer Society and as a member of the AFIPS Executive Committee. He is a member of the Organizing Committees of the IEEE DFT and BIST Workshops and General Chairman of the IEEE-CRC BAST Workshop. He was a founding member of the Editorial Board of IEEE DESIGN AND TEST magazine. In 1984, he received the IEEE Centennial Medal and IEEE Computer Society Technical Achievement Award in Testing. In 1990, he received the EURO ASIC 90 Prize for Fundamental Outstanding Contribution to Logic Synthesis. The IEEE Computer Society chose him for its 1991 Taylor Booth Education Award.