1. Introduction

With growing complexity of integrated circuits and systems, the cost of testing has become ever more significant. BIST is increasingly being applied as an effective means to reduce the cost of testing. The test complexity is cut down through systematic structured approach and the test time is reduced through concurrency. BIST also gives us the ability for testing parts in-circuit, and possibly on-line [McCluskey 85].

For built-in self testing, one must be able to generate and apply test patterns internally. Due to simplicity and compactness, LFSR is popularly used as the apparatus for generating pseudo-random test patterns. Pseudo-random test patterns are effective when the circuit is random-pattern testable. A typical circuit may however have some random pattern resistant (RPR) faults. As a result, it is often required to supplement random-patterns with few deterministically generated patterns. This situation may sometimes be alleviated by using weighted random patterns or through logic changes to improve random-pattern testability [Eichelberger 91]. Several techniques that embed deterministic patterns in pseudo-random sequence have been proposed [Dufaza 91], [Hellebrand 92], [Vasudevan 93]. These methods either have large area overhead or can embed only a limited number of deterministic patterns.

1.1 On Pseudo-Deterministic Testing

An optimal choice of pseudo-random subsequences can be made, if positions of deterministic test vectors, say corresponding to RPR faults, is known a priori. Through appropriate choice of seeds, the hardware apparatus for pseudo-random testing can be made to additionally cover deterministic patterns. This method of pseudo-deterministic test application is especially well suited for BIST, where the test vectors have to be generated internally.

When the LFSR has fewer stages, say less than 16, one can use the brute-force method of running it through the maximal sequence and thereby determine positions of deterministic patterns. However, as the number of stages become larger, it becomes prohibitive to do so. The theory of discrete logarithms has been used to address this problem.

A related problem of interest is that of identifying RPR faults. Random-pattern resistance is a function of pseudo-random test length. The exponential dependency between pseudo-random test length and fault coverage, suggests that there is a point beyond which increasing the test length does not improve fault coverage significantly [McCluskey 88]. For a given pseudo-random test length, an efficient method for identifying RPR faults has been suggested in [Waicukauski 85].

In order to find an optimal set of seeds, one must consider a reasonable size subset, if not all, of the test vectors for every RPR fault class. At least one of the test vectors for each RPR fault class needs to be covered by the pseudo-deterministic sequence. Clearly, if a very short test length is chosen, the working set of patterns to consider can become very large. Therefore, pseudo-deterministic test is best thought of as an enhancement and not an alternative to pseudo-random test. This trade-off and related heuristics have been discussed in [Lempel 94].

1.2 Modular Form LFSR - Generator of *GF(2^n)*

The theory of discrete logarithm has been applied to determine positions of bit patterns in the sequence generated by a modular form LFSR [Mukund 91], [Lempel 94]. Consider the modular realization of a n-stage LFSR in Fig. 1, with the feedback polynomial

$$p(x) = c_0 + c_1 x + \ldots + c_{n-1} x^{n-1} + x^n \quad .$$

We call this, the Shift Division Circuit (SDC).

Let $p(x)$ be primitive and α be one of its roots in *GF(2^n)*. Then we can identify the state:

$$\beta = \begin{bmatrix} b_0 \ b_1 \ \ldots \ b_{n-1} \end{bmatrix}$$

of the SDC with the field element:

$$\beta = b_0 + b_1 \alpha + \ldots + b_{n-1} \alpha^{n-1} \quad .$$

Notice that every step of work performed by the device corresponds to multiplying the field element β by the field element α. This means, starting from any non-zero state β_0, the SDC generates all the other non-zero elements of *GF(2^n)* successively as:

$$\beta_0, \beta_0 \alpha, \beta_0 \alpha^2, \ldots$$

In particular, in order to determine how many steps it will take for the SDC, starting from the state β_0, to arrive at the state β, we need only compute $log_{\alpha}\beta - log_{\alpha}\beta_0$.

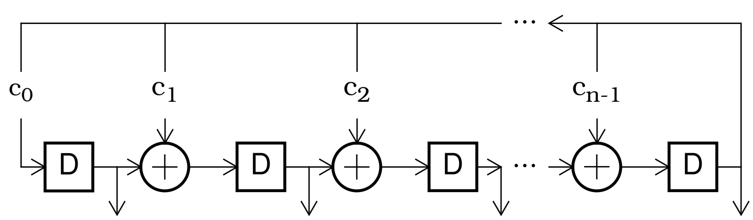

Figure 1. Modular Form LFSR or SDC

The problem of computing the discrete logarithm, $m = log_{\alpha}\beta$, has received wide attention in the area of Cryptography [Pohlig 78], [Coppersmith 84], [Odlyzko 84]. If the value of n is so chosen that 2^n-1 is a large prime or at least one of its prime factors is large, then it is very hard to extract logarithms in the field *GF(2^n)*. On the other hand, it is much easier to exponentiate. It is this disparity that led Diffe and Hellman [Diffe 76] to propose a cryptographic scheme based on exponentiation in finite field.

However, in the pseudo-deterministic test application, the value n can be chosen such that 2^n-1 is *smooth* (expressible as a product of relatively small primes). Fortunately, most values of n in the range of interest, say $16 < n < 100$, are such that 2^n-1 is smooth. Methods for computing discrete logarithms that are particularly suited for this application and the procedure for making choices of n have been proposed in [Mukund 91].

2. Motivation

Consider the pseudo-random BIST situation of Fig. 2. The input registers can be config

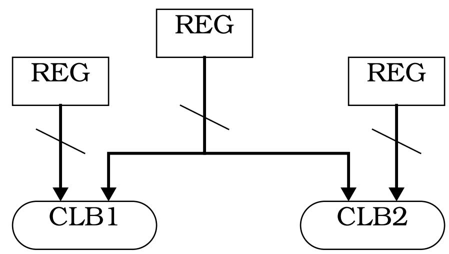

Figure 2. A BIST Example

ured as independent LFSRs to test random pattern testable faults in CLB1 and CLB2. However, for pseudo-deterministic testing, it is essential that all the inputs of every CLB come from the same source. As a result, we need to configure the three registers into one large LFSR, such that only a subset of the LFSR stages are connected to each CLB. The number of possible test patterns, for every RPR fault class, grows exponentially with the difference in the number of LFSR stages and the number of CLB inputs. The problem of identifying positions of deterministic test patterns is soon rendered impractical. In reality, the interplay between the inputs of CLBs is even more complex, especially in non-data path situations.

In order to satisfy the constraint that the inputs to every CLB must come from the same source, we adopt LFSR/SR, a device proposed for pseudo-exhaustive testing [Barzilai 83]. LFSR/SR is simply a standard form LFSR driving an arbitrary length shift register (scan-chain). The inputs of CLBs are connected to the shift register at arbitrarily chosen tap positions. Given certain tapping configuration, a method for selecting the feedback polynomial is discussed in [Barzilai 83]. In practice, the number of stages in the driving LFSR is one or two stages more than the number of inputs of the maximum input CLB*.

Our intent is to find a method to determine positions of bit-patterns, at arbitrary tap configurations, in the sequence generated by the LFSR/SR.

3. Standard Form LFSR - The Dual

In preparation, we first develop a linear transformation that will help us identify positions of bit-patterns in a standard form LFSR sequence.

*. Most combinational networks have more than one output. In many cases each of the outputs depend on only a subset of the inputs. We can hence consider the input sets on which every output depends as the target, instead of CLBs [McCluskey 84]. This can potentially make the driving LFSR smaller and hence further simplify the problem of pseudo-deterministic test.

We call the autonomous time-sequential circuit of Fig. 3,

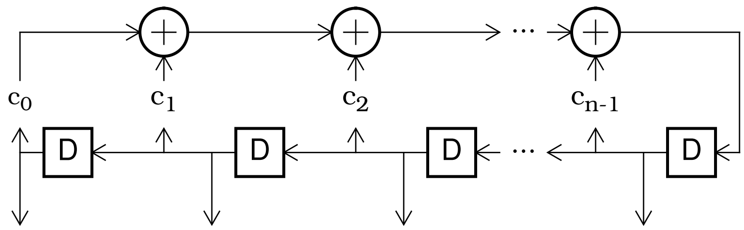

Figure 3. Standard Form LFSR

with the transition matrix:

$$T = \begin{bmatrix} 0 & 0 & \dots & 0 & c_0 \\ 1 & 0 & \dots & 0 & c_1 \\ 0 & 1 & \dots & 0 & c_2 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & c_{n-1} \end{bmatrix},$$

the standard form LFSR.

We also know that the transition matrix of an accompanying SDC with the same feedback polynomial is the transpose:

$$T^T = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \\ c_0 & c_1 & c_2 & \dots & c_{n-1} \end{bmatrix}$$

If the feedback polynomial is primitive and we start from any non-zero state, then both the circuits traverse through all the $2^n - 1$ distinct non-zero states. The position of a bit-pattern in the standard form LFSR sequence can be determined by the following theorem.

Theorem 1: There exists a one-to-one correspondence between the state sets of a standard form LFSR and a SDC with the same feedback polynomial, which is linear and respects the succession of states in both the devices.

Proof: Considering the invertible matrix:

$$A = \begin{bmatrix} c_1 & c_2 & c_3 & \dots & c_{n-1} & 1 \\ c_2 & c_3 & c_4 & \dots & 1 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ c_{n-1} & 1 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \end{bmatrix}, \qquad \textbf{(Equation 1)}$$

it is easy to verify that:

$$TA = AT^T$$

Thus, if we define the linear mapping σ by sending the state β of the standard form LFSR to the corresponding state:

$$\sigma(\beta) = \beta A \qquad \textbf{(Equation 2)}$$

of the accompanying SDC, then it is evidently invertible and we see from:

$$\sigma(\beta T) = \beta TA = \beta AT^T = \sigma(\beta) T^T,$$

that the transition of states in both the devices is preserved. Q. E. D.

We see from this theorem that in order to determine the distance between the test vectors β_0 and β in the standard form LFSR, we need only compute $log_\alpha \sigma(\beta) - log_\alpha \sigma(\beta_0)$.

4. LFSR/SR - The Apparatus

The apparatus particularly suited for pseudo-deterministic BIST is the LFSR/SR illustrated in Fig. 4. The shift register (scan chain) could potentially be inserted in a post-

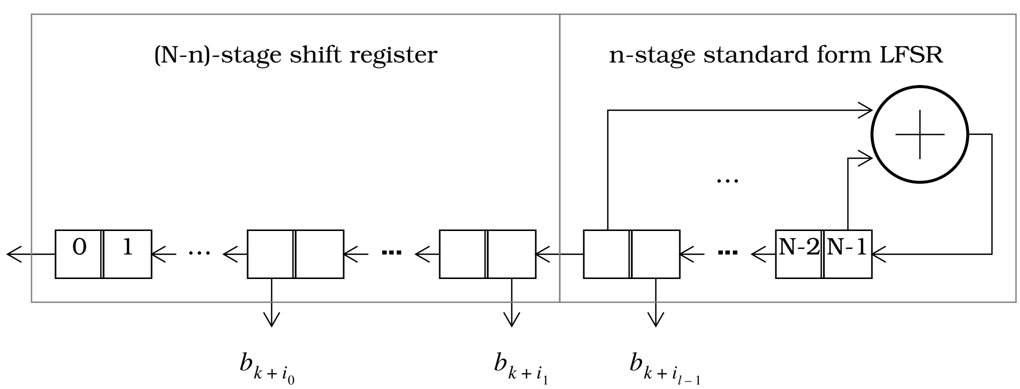

Figure 4. A l tap, N-stage LFSR/SR with n-stage driving LFSR

logic design step with the head of the shift register re-configured into a standard form LFSR.

As discussed in [Barzilai 83], one can generate test patterns of the form:

$$\delta_k = \left[b_{k+i_0}\ b_{k+i_1}\ \dots\ b_{k+i_{l-1}} \right], k \geq 0$$

by the help of an arbitrarily chosen tapping configuration:

$$\tau = \left[i_0\ i_1\ i_2\ \dots\ i_{l-1} \right], 0 \leq i_j < N-1$$

on a *N*-stage LFSR/SR, where the driving *n*-stage standard form LFSR has a primitive feedback polynomial. A problem of interest is to make the choice of a primitive feedback polynomial such that all *l*-patterns can be generated at a set of *s* arbitrarily chosen tapping configurations*:

$$\tau^{0, l_0}, \tau^{1, l_1}, \dots, \tau^{s, l_s}.$$

To this end, a theorem is obtained in [Barzilai 83]. Here we give a proof in a different perspective that illustrates how the position of a test pattern δ_k at the tapping configuration τ can be determined in the LFSR/SR sequence.

Theorem 2: For any integer i_j, we divide x^{i_j} by *p(x)* to obtain the remainder

$$r_{i_j}(x) = x^{i_j} mod(p(x)) = c_{i_j,0} + c_{i_j,1}x + \dots + c_{i_j,n-1}x^{n-1}.$$

If the feedback polynomial *p(x)* is primitive, then the tapping configuration τ generates all *l*-patterns if and only if, the vectors:

$$\gamma(i_j) = \left[c_{i_j,0}\ c_{i_j,1}\ \dots\ c_{i_j,n-1} \right], 0 \leq j \leq l-1 \qquad \textbf{(Equation 3)}$$

are linearly independent.

Proof: Suppose the LFSR starts working with the non-zero state:

$$\beta_0 = \left[b_0\ b_1\ \dots\ b_{n-1} \right]$$

and we write:

$$\beta_k = \beta_0 T^k.$$

Divide x^{i_j} by *p(x)* to obtain:

$$x^{i_j} = q_{i_j}(x) p(x) + r_{i_j}(x), deg(r_{i_j}(x)) < n.$$

Then, since *p(T) = 0*, we see from:

$$\beta_{k+i_j} = \beta_0 T^{k+i_j} = \beta_k T^{i_j} = \beta_k r_{i_j}(T),$$

*. For the sake of clarity only one of the *s* tapping configurations is illustrated in Fig. 4.

that the signal b_{k+i_j}, being the zeroth component of the vector β_{k+i_j}, can be computed as:

$$b_{k+i_j} = \beta_k \gamma^T(i_j),$$

which means we have:

$$\delta_k = \beta_k C,$$

where:

$$C = \left[\gamma^T(i_0)\ \gamma^T(i_1)\ \dots\ \gamma^T(i_{l-1})\right]. \qquad \textbf{(Equation 4)}$$

Thus we see, if the columns of C are linearly dependent, then fixed non-trivial linear relations can be found between the components of δ_k. This means not all l-patterns can be obtained from the tap configuration τ. If, on the contrary, the columns of C are linearly independent, then since β_k can be any non-zero n-vector, every given l-pattern can be obtained by selecting a suitable k. Q. E. D.

In particular, if we wish the tapping configuration to generate all the l-patterns then l cannot exceed n.

On the other hand, if $l < n$ and the matrix C has rank l, then for any given l-vector δ_k, the linear system:

$$\beta C = \delta_k \qquad \textbf{(Equation 5)}$$

has $2^{(n-l)}$ solutions for β, say $V(\delta_k)$. Thus, if the driving LFSR starts with an initial state β_0, then for every step:

$$\upsilon = (N-n-k) + (log_\alpha \sigma(\beta) - log_\alpha \sigma(\beta_0))\ mod\,(2^n - 1), \qquad \beta \in V(\delta_k) \qquad \textbf{(Equation 6)}$$

of work, the configuration τ will output the pattern δ_k.

5. An Illustration

In this section, we provide an example to illustrate the application of LFSR/SR in pseudo-deterministic testing. Consider the scenario shown in Fig. 5. The design has a

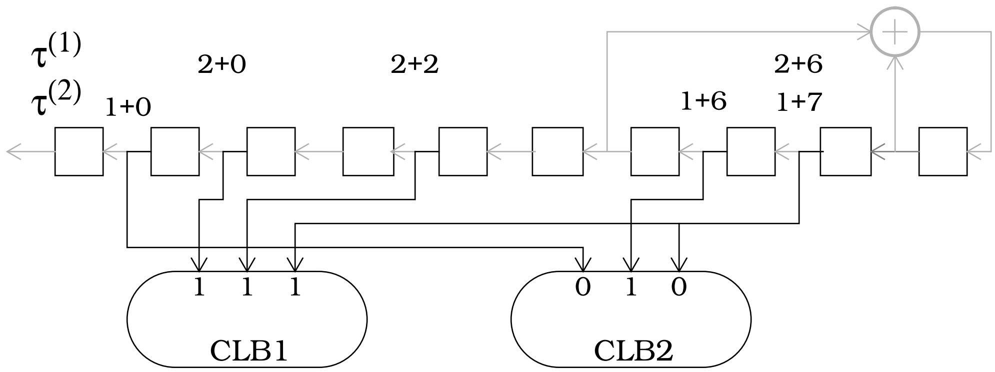

Figure 5. An Application of the Pseudo-Deterministic Test Apparatus

set of registers, some of which are connected to the inputs of CLB1 and CLB2. The goal is to apply the deterministic pattern $\delta_2^{(1)} = \begin{bmatrix}1 & 1 & 1\end{bmatrix}$ to CLB1 and $\delta_1^{(2)} = \begin{bmatrix}0 & 1 & 0\end{bmatrix}$ to CLB2.

The registers are configured into a shift register. The order in which the registers are connected is arbitrary. As a result, we have tap configurations $\tau^{(1)}$ and $\tau^{(2)}$ connected to CLB1 and CLB2 respectively. We can pick a primitive feedback polynomial for the driving LFSR using the method suggested in [6], such that the columns of the C matrices corresponding to $\tau^{(1)}$ and $\tau^{(2)}$ are linearly independent. Instead, let us arbitrarily pick a 4-stage LFSR with the primitive feedback polynomial: $x^4 = 1 + x^3$, and test for the linear independence of the columns of C matrices. From Eqn. 3, we have:

$$
\begin{aligned}
\gamma^{(1)}(0) &= x^0 mod(p(x)) = [1000],\\
\gamma^{(1)}(2) &= x^2 mod(p(x)) = [0010],\\
\gamma^{(1)}(6) &= x^6 mod(p(x)) = x^4x^2 = (1+x^3)x^2 = x^2+x^5 = x^2+(1+x^3)x\\
&= x+x^2+x^4 = 1+x+x^2+x^3 = [1111],\\
\gamma^{(2)}(0) &= x^0 mod(p(x)) = [1000],\\
\gamma^{(2)}(6) &= [1111],\\
\gamma^{(2)}(7) &= x^6x = (1+x+x^2+x^3)x = x+x^2+x^3+x^4 = 1+x+x^2 = [1110]
\end{aligned}
$$

Therefore, from Eqn. 4,

$$
C^{(1)} = \begin{bmatrix}1 & 0 & 1\\0 & 0 & 1\\0 & 1 & 1\\0 & 0 & 1\end{bmatrix}, \qquad and \qquad C^{(2)} = \begin{bmatrix}1 & 1 & 1\\0 & 1 & 1\\0 & 1 & 1\\0 & 1 & 0\end{bmatrix}.
$$

By inspection, the columns of $C^{(1)}$ and $C^{(2)}$ are linearly independent.

We can compute discrete logarithms using methods suggested in [5]. Since n is small, all the elements of $GF(2^n)$ can be generated using the SDC shown in Fig. 6.

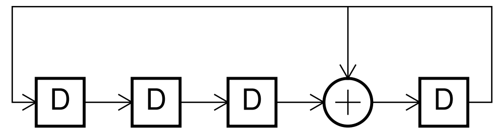

Figure 6. SDC with feedback polynomial, $x^4 = 1 + x^3$

Working the SDC from the root, $\alpha = \begin{bmatrix}0 & 1 & 0 & 0\end{bmatrix}$, we can readily obtain the discrete logarithm look-up table:

$$\begin{array}{llll}
log_\alpha\begin{bmatrix}0 & 1 & 0 & 0\end{bmatrix} = 1, & log_\alpha\begin{bmatrix}1 & 1 & 0 & 1\end{bmatrix} = 5, & log_\alpha\begin{bmatrix}1 & 0 & 1 & 0\end{bmatrix} = 9, & log_\alpha\begin{bmatrix}0 & 1 & 1 & 0\end{bmatrix} = 13, \\
log_\alpha\begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix} = 2, & log_\alpha\begin{bmatrix}1 & 1 & 1 & 1\end{bmatrix} = 6, & log_\alpha\begin{bmatrix}0 & 1 & 0 & 1\end{bmatrix} = 10, & log_\alpha\begin{bmatrix}0 & 0 & 1 & 1\end{bmatrix} = 14, \\
log_\alpha\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix} = 3, & log_\alpha\begin{bmatrix}1 & 1 & 1 & 0\end{bmatrix} = 7, & log_\alpha\begin{bmatrix}1 & 0 & 1 & 1\end{bmatrix} = 11, & log_\alpha\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix} = 15, \\
log_\alpha\begin{bmatrix}1 & 0 & 0 & 1\end{bmatrix} = 4, & log_\alpha\begin{bmatrix}0 & 1 & 1 & 1\end{bmatrix} = 8, & log_\alpha\begin{bmatrix}1 & 1 & 0 & 0\end{bmatrix} = 12, & log_\alpha\begin{bmatrix}0 & 1 & 0 & 0\end{bmatrix} = 1 \cdot
\end{array}$$

From Eqn. 1, we have,

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

By solving for $\beta^{(1)}$ and $\beta^{(2)}$ in $\beta^{(1)}C^{(1)} = \delta_2^{(1)}$ and $\beta^{(2)}C^{(2)} = \delta_1^{(2)}$ respectively, we have:

$$V\left(\delta_2^{(1)}\right) = \{\begin{bmatrix}1 & 0 & 1 & 1\end{bmatrix}, \begin{bmatrix}1 & 1 & 1 & 0\end{bmatrix}\} \qquad and \qquad V\left(\delta_1^{(2)}\right) = \{\begin{bmatrix}0 & 1 & 1 & 1\end{bmatrix}, \begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}\} \cdot$$

Let the LFSR start at an arbitrary state, say $\beta_0 = \begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}$. Applying Eqn. 6, we have the positions of bit-patterns as:

$$\begin{aligned}
\upsilon_1^{(1)} &= (10-4-2) + \left(log_\alpha\left(\begin{bmatrix}1 & 0 & 1 & 1\end{bmatrix} \cdot A\right) - log_\alpha\left(\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix} \cdot A\right)\right) mod\,(15) \\
&= 4 + \left(log_\alpha\begin{bmatrix}0 & 1 & 1 & 1\end{bmatrix} - log_\alpha\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}\right) mod\,(15) = 4 + (8-15)\,mod\,(15) = 12,
\end{aligned}$$

$$\begin{aligned}
\upsilon_2^{(1)} &= (10-4-2) + \left(log_\alpha\left(\begin{bmatrix}1 & 1 & 1 & 0\end{bmatrix} \cdot A\right) - log_\alpha\left(\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix} \cdot A\right)\right) mod\,(15) \\
&= 4 + \left(log_\alpha\begin{bmatrix}1 & 0 & 0 & 1\end{bmatrix} - log_\alpha\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}\right) mod\,(15) = 4 + (4-15)\,mod\,(15) = 8, \\
\upsilon_1^{(2)} &= (10-4-1) + \left(log_\alpha\left(\begin{bmatrix}0 & 1 & 1 & 1\end{bmatrix} \cdot A\right) - log_\alpha\left(\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix} \cdot A\right)\right) mod\,(15) \\
&= 5 + \left(log_\alpha\begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix} - log_\alpha\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}\right) mod\,(15) = 5 + (2-15)\,mod\,(15) = 7, \\
\upsilon_2^{(2)} &= (10-4-1) + \left(log_\alpha\left(\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix} \cdot A\right) - log_\alpha\left(\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix} \cdot A\right)\right) mod\,(15) \\
&= 5 + \left(log_\alpha\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix} - log_\alpha\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}\right) mod\,(15) = 5 + (15-15)\,mod\,(15) = 5 \cdot
\end{aligned}$$

The bit pattern $\delta_2^{(1)} = \begin{bmatrix}1 & 1 & 1\end{bmatrix}$ at the tapping configuration $\tau^{(1)}$ is generated at the step 8 and 12. The bit pattern $\delta_1^{(2)} = \begin{bmatrix}0 & 1 & 0\end{bmatrix}$ at $\tau^{(2)}$ is generated at the step 5 and 7.

Let us now verify the above by actually running the LFSR/SR as shown in Fig. 7.

<table>
<tr><th></th><th>REG0</th><th>REG1</th><th>REG2</th><th>REG3</th><th>REG4</th><th>REG5</th><th>REG6</th><th>REG7</th><th>REG8</th><th>REG9</th></tr>
<tr><td></td><td></td><td></td><td>$\tau_0^{(1)}$</td><td></td><td>$\tau_1^{(1)}$</td><td></td><td></td><td></td><td>$\tau_2^{(1)}$</td><td></td></tr>
<tr><td></td><td></td><td>$\tau_0^{(2)}$</td><td></td><td></td><td></td><td></td><td></td><td>$\tau_1^{(2)}$</td><td>$\tau_2^{(2)}$</td><td></td></tr>
<tr><td>STEP</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td></td><td></td><td></td><td></td><td></td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>2</td><td></td><td></td><td></td><td></td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>3</td><td></td><td></td><td></td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>4</td><td></td><td></td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>5</td><td></td><td>|0|</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>|1|</td><td>|0|</td><td>1</td></tr>
<tr><td>6</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>7</td><td>0</td><td>|0|</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>|1|</td><td>|0|</td><td>1</td></tr>
<tr><td>8</td><td>0</td><td>1</td><td>‖1‖</td><td>1</td><td>‖1‖</td><td>0</td><td>1</td><td>0</td><td>‖1‖</td><td>1</td></tr>
<tr><td>9</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>10</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>11</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>12</td><td>1</td><td>0</td><td>‖1‖</td><td>0</td><td>‖1‖</td><td>1</td><td>0</td><td>0</td><td>‖1‖</td><td>0</td></tr>
<tr><td>13</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>14</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>15</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
</table>

Figure 7. A sequence generated by a (10,4) LFSR/SR

Notice that if we seed the driving LFSR with the pattern [0111], a sequence of length 7 covers the bit patterns of interest. A minimum sequence length of 6 is required for the driving LFSR to reach the farthest tap position, namely REG1.

6. Conclusion

For pseudo-deterministic test (pseudo-random test that also covers a set of known deterministic patterns), one needs to identify positions of bit-patterns in pseudo-random sequences. In this paper, a linear transformation [Eqn. 2] to relate the sequence generated by a standard form (external-XOR) LFSR with that of an accompanying modular form (internal-XOR) LFSR has been presented. The positions of bit-patterns are then identified by computing discrete logarithms in the field generated by the accompanying modular form LFSR. A further transformation [Eqn. 5] to relate the sequence generated by a standard form LFSR with the sequence generated at arbitrarily chosen tap positions in a LFSR/SR (shift register driven by a standard form LFSR) is also presented. A formula [Eqn. 6] for identifying positions of bit-patterns in the sequence generated at these tap positions has been provided.

Pseudo-deterministic test requires that all the inputs of every CLB (combinational logic block) come from the same pattern source. In a typical BIST scenario, there is interplay between the inputs of different CLBs. Hence, it is impractical to directly apply LFSR. In this paper, we have addressed this problem by proposing the use of LFSR/SR as the pattern source. An arbitrary length LFSR/SR can be created simply by scan chaining the registers driving the CLBs, whose inputs interplay. This apparatus makes pseudo-deterministic testing practical and cost effective.

Acknowledgments

The authors gratefully acknowledge Kencheng Zeng and Pran Kurup for helpful comments and suggestions. This work was supported in part by the Innovative Science and Technology Office of the Strategic Defense Initiative Organization and administered through the Office of Naval Research under Contract No. N00014-92-J-1782, by the National Science Foundation under Grant No. MIP-9107760, and by Cirrus Logic Inc.

References

[Barzilai 83] Barzilai, Z., D. Coppersmith, and A. L. Rosenberg, "Exhaustive Generation of Bit Patterns with Applications to VLSI Self-Testing," *IEEE Trans. Comput.*, Feb 1983.

[Coppersmith 84] Coppersmith, D., "Fast Evaluation of Logarithms in Fields of Characteristic Two," *IEEE Trans. Info. Theory*, July 1984.

[Diffe 76] Diffe, W., and M. E. Hellman, "New Directions in Cryptography," *IEEE Trans. Info. Theory*, 1976.

[Dufaza 91] Dufaza, C., and G. Cambon, "LFSR based Deterministic and Pseudo-Random Test Pattern Generator structures," *Proc. European Test Conference*, 1991.

[Eichelberger 91] Eichelberger, E. B., E. Lindbloom, J. A. Waicukauski and T. W. Williams, "Structured Logic Testing," *Prentice Hall, Englewood Cliffs*, New Jersey 1991.

[Hellebrand 92] Hellebrand, S., S. Tarnick and J. Rajski, "Generation of Vector Patterns Through Reseeding of Multiple-Polynomial Linear Feedback Shift Registers," *Proc. ITC*, 1992.

[Lempel 94] Lempel, M., Sandeep K. Gupta, and Melvin A. Breuer, "Test Embedding with Discrete Logarithms," *IEEE VLSI Test Symposium*, 1994.

[McCluskey 88] McCluskey, E. J., Samy Makar, Samiha Mourad, and Kenneth D. Wagner, "Probability Models for Pseudorandom Test Sequences," *IEEE Trans. Computer-Aided Design*, Jan. 1988.

[McCluskey 85] McCluskey, E. J., "Built-In Self-Test Techniques," *IEEE Design & Test*, April, 1985.

[McCluskey 84] McCluskey, E. J., "Verification Testing-A Pseudoexhaustive Test Technique," *IEEE Trans. Comput.*, June 1984.

[Mukund 91] Mukund, S. K., T.R.N. Rao, and Kencheng Zeng, "On Reducing Test Length in LFSR based Testing," *VLSI Design Symposium*, Jan. 1991.

[Odlyzko 84] Odlyzko, A. M., "Discrete Logarithms in Finite Fields and their Cryptographic Significance," Adv. in Cryptology (Proc. of Eurocrypt '84), *Lecture Notes in Computer Science, Vol. 209, Springer-Verlag*, New York, 1984.

[Pohlig 78] Pohlig, S. C., and Martin E. Hellman, “An Improved Algorithm for Computing Logarithms over *GF(p)* and Its Cryptographic Significance,” *IEEE Trans. Info. Theory*, Jan. 1978.

[Vasudevan 93] Vasudevan, B., D.E. Ross, M. Gala and K.L. Watson, “LFSR Based Deterministic Hardware for At-Speed BIST,” *Proc. VLSI Test Symp.*, 1993.

[Waicukauski 85] Waicukauski, J. A., Edward B. Eichelberger, Donato O. Forlenza, Eric Lindbloom, and Thomas McCarthy, “Fault Simulation for Structured VLSI,” *VLSI Systems Design*, Dec. 1985.

An Apparatus for Pseudo-Deterministic Testing

Shridhar K. Mukund* and Edward J. McCluskey

CENTER FOR RELIABLE COMPUTING

Computer Systems Laboratory

Departments of Electrical Engineering and Computer Science

Stanford University

Stanford, CA 94305

T.R.N. Rao

CENTER FOR ADVANCED COMPUTER STUDIES

University of Southwestern Louisiana

Lafayette, LA 70504

CRC Report No. 94-12

(CSL TR # 94-642)

October 1994

Abstract

Pseudo-random testing is popularly used, particularly in Built-In Self Test (BIST) applications. To achieve a desired fault coverage, pseudo-random patterns are often supplemented with few deterministic patterns. When positions of deterministic patterns in the pseudo-random sequence are known a priori, pseudo-random sub-sequences can be chosen such that they also cover these deterministic patterns. We call this method of test application, pseudo-deterministic testing. The theory of discrete logarithm has been applied to determine positions of bit-patterns in the pseudo-random sequence generated by a modular form or internal-XOR Linear Feedback Shift Register (LFSR) [Mukund 91], [Lempel 94]. However, the scheme requires that all the inputs of the combinational logic block (CLB), under test, come from the same LFSR source. This constraint in circuit configuration severely limits its application.

In this paper, we propose a practical and cost effective technique for pseudo-deterministic testing. For most part, the problem of circuit configuration has been simplified to one of scan path insertion, by employing LFSR/SR (an arbitrary length shift register driven by a standard form or external-XOR LFSR). To enable the usage of LFSR/SR as a pseudo-deterministic pattern source, we propose a method to determine positions of bit-patterns, at arbitrarily chosen tap configurations, in the LFSR/SR sequence.

*. This author is with R&D, Cirrus Logic Inc., pursuing graduate study in Electrical Engineering through the Honors Cooperative Program at Stanford University.

Table of Contents

List of Figures