Ring Generators—New Devices for Embedded Test Applications

Grzegorz Mrugalski, Janusz Rajski, *Associate Member, IEEE*, and Jerzy Tyszer, *Senior Member, IEEE*

***Abstract*—This paper presents a novel methodology of designing generators and compactors of test data. The essence of the proposed approach is to use a set of transformations, which alters the structure of the conventional linear feedback shift registers (LFSRs) while preserving the transition function of the original circuits. It is shown that after applying the transition function preserving transformations in a certain order, the resultant circuits feature a significantly reduced the number of levels of XOR logic, minimized internal fanouts, and simplified circuit layout and routing, as compared to previous schemes based on external feedback LFSRs, internal feedback LFSRs, and cellular automata, all implementing the same characteristic polynomial. Consequently, the proposed devices can operate at higher speeds than those of conventional solutions and become highly modular structures.**

***Index Terms*—Built-in self-test, design for testability, linear feedback shift registers (LFSRs), phase shifters, transition function preserving transformations.**

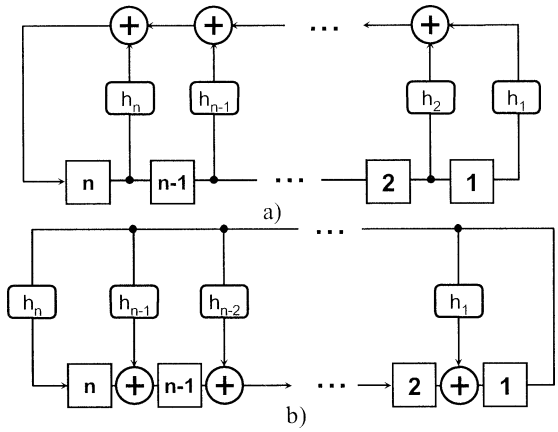

Fig. 1. Two canonical forms of LFSRs.

I. Introduction

HIGH integration of system-on-a-chip designs is challenging the test realm in a number of ways. To be successful, the newest test generation schemes must respond to a wide spectrum of problems related to high performance, small area, and modular implementations. Pseudorandom test pattern generators, structures commonly employed in a variety of test applications, have to further conform to four basic requirements: 1) high fault coverage; 2) short test-application time; 3) simple hardware necessary to produce test data; 4) and ease of automation of synthesis techniques that generate this hardware. Consequently, many schemes have been proposed to accomplish various tradeoffs between these factors. By far, the most popular devices used to generate pseudorandom sequences are linear feedback shift registers (LFSRs) [5] and local-neighborhood linear cellular automata (CAs) [6]. These devices are also deployed to perform test-data decompression and test-response compaction [21]. Needless to say, they find a variety of applications in such diverse areas as error detection and correction, data transmission, mobile telephony, cryptography, data compression, white noise generation, and many others. Therefore, their synthesis and possible areas of applicability have drawn considerable attention through the years [1], [2], [7], [9], [16], [26].

An n-bit LFSR consists of n memory elements and XOR gates connected as shown in Fig. 1(a). In particular, the modulo 2 sum of the selected stages is fed back to the left-most stage of the register. Such an implementation is called an external feedback LFSR. The transition function of this LFSR can be represented in terms of an $n \times n$ matrix $\mathbf{Me}$ referred to as the characteristic (or companion) matrix. The characteristic matrix assumes the following form:

$$\mathbf{Me} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \cdots & 1 \\ h_1 & h_2 & h_3 & \cdots & h_{n-1} & h_n \end{bmatrix} \tag{1}$$

where $h_i \in \{0,1\}$, $i = 1, 2, \ldots, n$. Recall that the ith row determines the dependency of the ith cell on the remaining stages of the LFSR. The same LFSR can also be represented by its characteristic polynomial (or equation). It is obtained [9] as the determinant of the matrix $\mathbf{Me} - \mathbf{I}x$, i.e.,

$$f(x) = \det(\mathbf{Me} - \mathbf{I}x) = \begin{vmatrix} x & 1 & 0 & \cdots & 0 & 0 \\ 0 & x & 1 & \cdots & 0 & 0 \\ 0 & 0 & x & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \cdots & 1 \\ h_1 & h_2 & h_3 & \cdots & h_{n-1} & h_n + x \end{vmatrix}$$
$$= x^n + h_n x^{n-1} + \cdots + h_2 x + h_1 \tag{2}$$

where the term $h_i x^{i-1}$ refers to the ith flip-flop of the register, such that, if $h_i = 1$, then there is a feedback tap taken

Manuscript received July 8, 2003; revised November 29, 2003. An earlier version of this paper appeared in *Proceedings of the IEEE VLSI Test Symposium*, 2003, pp. 57–62. This paper was recommended by Associate Editor S. Hellebrand.

G. Mrugalski and J. Rajski are with Mentor Graphics Corporation, Wilsonville, OR 97070 USA (e-mail: grzegorz_mrugalski@mentor.com; janusz_rajski@mentor.com).

J. Tyszer is with the Institute of Electronics and Telecommunications, Poznań University of Technology, 60-965 Poznań, Poland (e-mail: tyszer@et.put.poznan.pl).

Digital Object Identifier 10.1109/TCAD.2004.831584

from this flip-flop. A characteristic polynomial, which causes an n-bit LFSR to go through all possible $2^n - 1$ nonzero states is called a primitive polynomial, while the corresponding LFSR is known as a maximum-length LFSR producing the output sequence termed a maximum-length sequence or m-sequence.

Fig. 1(b) shows an alternative LFSR implementation with interspersed XOR gates. This configuration is referred to as an internal feedback LFSR. Its distinctive feature is that the output of the rightmost stage of the LFSR is fed back to those stages which are indicated by the following characteristic matrix:

$$\mathbf{Mi} = \begin{bmatrix} h_1 & 1 & 0 & 0 & \cdots & 0 \\ h_2 & 0 & 1 & 0 & \cdots & 0 \\ h_3 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ h_{n-1} & 0 & 0 & 0 & \cdots & 1 \\ h_n & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}. \tag{3}$$

It can be easily verified that the corresponding characteristic polynomial $g(x) = det(\mathbf{Mi} - \mathbf{I}x)$ is given in this case by $h_n x^n + h_{n-1}x^{n-1} + \cdots + h_1 x + 1$. Again, if this LFSR is initialized with a nonzero value and its characteristic polynomial is a primitive one, this circuit will produce all $2^n - 1$ nonzero states before entering the seed state.

The depth of the linear logic in their feedback paths limits the operating speed of external feedback LFSRs. Moreover, implementation of internal feedback LFSRs 1) may cause severe frequency degradation [10] due to long feedback lines and 2) may take up a considerable area as it involves a large fanout on the right-most stage output. The limitations of both canonical forms of the LFSRs are noticeably pronounced for polynomials with a large number of terms. Finally, the irregularity of their interconnection structures precludes conventional LFSRs from becoming easily cascadable units. CAs, though modular, suffer from the resulting complexity due to at least one XOR gate occurring at each stage. For larger structures, this drawback can be undesirable, and it may prevent the use of CAs in many situations.

Several attempts were made to improve the performance of conventional LFSRs. Both, papers available in the technical literature and many patents disclose various LFSR configurations aimed primarily at providing LFSRs with higher operating speeds and improved overall performance. In particular, the proposed solutions include:

- hybrid designs [23], [24] that allow one to reduce the number of XOR gates (in certain cases even by half) by using both canonical forms of the feedback logic in the same register;
- windmill machines [25] consisting of several generators with a common stage; the resulting state transition rate is elevated, however, at a cost of additional registers;
- decimation techniques [3], [14] allowing one to sum up a number of m-sequences produced by independent devices; in addition to several LFSRs, a multiphase clock generator and an extra multiplexer are required to implement this scheme;
- interconnections of several shift registers [11], [26]; although suitable for modular applications, these solutions can employ only trinomials and produce highly correlated patterns since XOR gates are used only to interconnect adjacent modules of the generator;
- linear finite state machines with embedded deterministic patterns [8]; a cumbersome logic synthesis of such devices as well as complexity of the resulting hardware virtually preclude their usage in majority of test applications;
- designs based on T-type flip-flops [3], while having several constrains of conventional schemes, these devices are also not very well integrated with automated design flows.

In this paper, we propose a new, comprehensive and systematic methodology that can be used to automate the synthesis process of optimized linear finite state machines. Our primary objective is to provide a method that can be used to obtain high-speed modular structures, which are capable of generating a desired m-sequence while having significantly reduced both the number of levels of logic and the internal fanout counts. Consequently, such devices may feature minimal delays on critical paths and a reduced number of elements driven by the same stem. These goals are achieved by introducing transformations (Section II) that will generate a large variety of LFSR implementations having the same characteristic polynomial, so that the whole space of realizations can be easily explored in order to make tradeoffs and meet design goals. We will show that in virtually all practical cases it is possible to reduce the internal fanouts to two and the number of levels of logic to one two-input XOR gate placed between any pair of memory elements, regardless of the number of terms in the feedback polynomial.

This new design scheme is best exemplified by an architecture named a *ring generator* as described in Section III. A typical ring generator has a smaller number of levels of logic than a corresponding external feedback LFSR. It also features a smaller fanout than the original internal feedback LFSR. Therefore, it can achieve higher performance than any other LFSR-based device used so far. Furthermore, without interfering with the feedback polynomial, the ring generators let designers minimize routing complexity, optimize wire sizing, and make the overall layout as compact as possible. A detailed comparison of ring generators with other existing solutions is provided in Section IV. In Section V, more advanced applications of the proposed methodology are discussed. For the sake of self containment, we demonstrate in Section VI how appropriate phase shifters for ring generators can be obtained in a time-efficient manner by generalizing the methods proposed earlier [19], [22]. The paper concludes with Section VII.

II. Sequence Preserving Transformations of LFSRs

Consider an internal feedback LFSR with n memory elements and a number of feedback connections. Each of the latter components is assumed to consist of the following three items: a *source tap* corresponding to an output of a storage device feeding this particular connection, the actual *feedback line* of a span defined by a given characteristic polynomial, and an XOR gate placed at the *destination* of the feedback connection, that is, at the input to another storage device (see Fig. 1). The LFSR architecture can then be transformed by moving its feedback connections across memory elements for the purpose of performance optimization and to minimize the total length of the

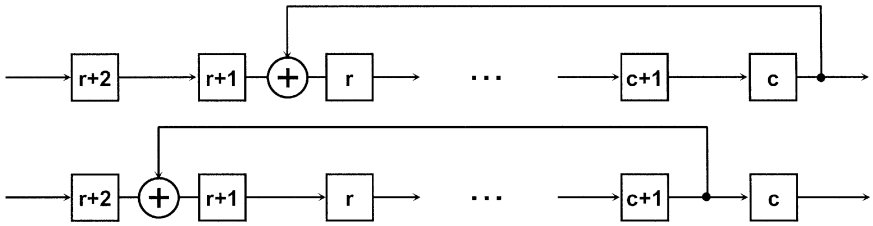

Fig. 2. EL transformation.

feedback lines. While altering the structure of the LFSR, however, these transformations should preserve its original transition function.

During a single transformation, the source tap, the destination gate, or both may cross other feedback line terminals. In such cases, appropriate actions may be required whereby the original functionality of the circuit undergoing transformations is maintained. Accordingly, in this section we describe two transformations which allow for modification of the original circuit by following rules applicable in various situations. One of these transformations is an elementary shift to the left (EL). It is used to move a single connection line by one memory element to the left. As shown in Fig. 2, when transformation EL is applied to a circuit (only that part of the LFSR which is affected by EL is depicted), it moves the XOR gate from the input of a given storage device to the input of its predecessor (here, flip-flops r and $r+1$, respectively), and it relocates the source tap of the feedback line accordingly. The fundamental property of transformation EL gives the following theorem.

Theorem 1: If transformation EL is applied to a feedback connection of a given LFSR in such a way that neither its source tap crosses any XOR gate nor its XOR gate crosses another source tap, then the transition function of the LFSR is preserved.

Proof: With no loss of generality, we may assume that before applying transformation EL a given LFSR has certain number of feedback connections represented by nonzero entries in its characteristic matrix $\mathbf{M}$ and placed below the main diagonal. Let us also assume that transformation EL is applied to a feedback connection originating at flip-flop c and feeding flip-flop r. Hence, the characteristic polynomial of the original LFSR is given by $h(x)=\det(\mathbf{M}-\mathbf{I}x)$, where matrix $\mathbf{M}-\mathbf{I}x$ has the following form (note the location of one in row r and column c):

$$\mathbf{M}-\mathbf{I}x= \begin{array}{c} \\ c \\ \\ \\ \\ \\ r \\ \\ \\ \\ \end{array} \begin{bmatrix} x & 1 & 0 & & & & & & & 0 \\ & x & 1 & & & & & & & 0 \\ & & \ddots & \ddots & & & & & & \vdots \\ & & & x & 1 & & & & & 0 \\ 0 & 0 & \cdots & 0 & x & 1 & & & & 0 \\ & & & & \vdots & \ddots & \ddots & & & \vdots \\ & & & \rightarrow & 1 & \cdots & x & 1 & \cdots & 0 \\ & & & & \uparrow & & & \ddots & \ddots & \vdots \\ & & & & & & & & x & 1 \\ 1 & & & & & & & & & x \end{bmatrix}. \tag{4}$$

Let $\mathbf{H}=\mathbf{M}-\mathbf{I}x$. It is worth noting, that matrix $\mathbf{H}$ has zeros in row c (except entry x and a one next to it) as, otherwise, it would not be possible to carry out transformation EL. Indeed, a nonzero entry would indicate the presence of XOR gate that the moving source tap would have to cross. Moreover, there are zeros in column $r+1$, as, otherwise, the XOR gate moving from flip-flip r would have to cross another source tap at the output of flip-flop $r+1$. The determinant of matrix $\mathbf{H}$ can be computed by multiplying the entries in column c of $\mathbf{H}$ by their cofactors $C_{i,c}$ and adding the resulting products, that is

$$\begin{aligned}\det(\mathbf{H}) &= 1\cdot C_{c-1,c}+x\cdot C_{c,c}+1\cdot C_{r,c}+\alpha \\ &= C_{c-1,c}+x\cdot C_{c,c}+C_{r,c}+\alpha \end{aligned} \tag{5}$$

where α is a sum of cofactors associated with the remaining nonzero entries in column c which are not the subject of transformation EL. In a similar manner, one can obtain matrix $\mathbf{Ht}$ that represents the transformed LFSR obtained by moving a one in row r and column c to row $r+1$ and column $c+1$. The characteristic polynomial of the transformed LFSR is given by

$$\begin{aligned}\det(\mathbf{Ht}) &= 1\cdot C_{c,c+1}+x\cdot C_{c+1,c+1}+1\cdot C_{r+1,c+1}+\beta \\ &= C_{c,c+1}+x\cdot C_{c+1,c+1}+C_{r+1,c+1}+\beta \end{aligned} \tag{6}$$

where β is a sum of cofactors associated with the remaining nonzero entries in column $c+1$. We will prove that $\det(\mathbf{H})=\det(\mathbf{Ht})$.

In order to determine $C_{c-1,c}=(-1)^{c-1+c}\det(\mathbf{H}_{c-1,c})$, it suffices to compute the determinant of the submatrix $\mathbf{H}_{c-1,c}$ that remains after row $c-1$ and column c are deleted from $\mathbf{H}$. This is because all calculations are carried out in the Galois field modulo 2. The same rule applies to the remaining components of (5) and (6). From (4), it follows that $\det(\mathbf{H}_{c-1,c})$ and $\det(\mathbf{Ht}_{c,c+1})$ are given by the following formulas:

$$\det(\mathbf{H}_{c-1,c})= \begin{vmatrix} x & 1 & & & & & & & \\ & x & 1 & & & & & & \\ & & \ddots & \ddots & & & & & \\ & & & x & 1 & \downarrow & & & \\ 0 & 0 & \cdots & 0 & 0 & 1 & \Leftarrow & & \\ & & & & & x & 1 & & \\ & & & & & & \ddots & \ddots & \\ & & & & & & & x & 1 \\ 1 & & & & & & & & x \end{vmatrix}$$

$$\det(\mathbf{Ht}_{c,c+1}) = \begin{vmatrix} x & 1 & & & & & & & \\ & x & 1 & & & & & & \\ & & \ddots & \ddots & & & & & \\ & & & x & 1 & \Downarrow & & & \\ & & & & x & 1 & \leftarrow & & \\ & & & & & 0 & 1 & & \\ & & & & & & x & 1 & \\ & & & & & & & \ddots & \\ 1 & & & & & & & & x \end{vmatrix}. \quad (7)$$

As can be verified, $\mathbf{H}_{c-1,c}$ features a single nonzero entry in row $c-1$ (indicated by a double arrow). This entry is also located in column c (a single arrow). Furthermore, $\mathbf{Ht}_{c,c+1}$ has a one in column c, which occurs in row $c-1$, as well. Therefore, the determinant of $\mathbf{H}_{c-1,c}$ can be evaluated by using cofactor expansion along row $c-1$, while the determinant of $\mathbf{Ht}_{c,c+1}$ can be evaluated by using cofactor expansion along column c. Clearly, deleting in both cases row $c-1$ and column c yields the identical submatrices. As for $\det(\mathbf{Ht}_{c,c+1})$, in general it may feature other components α corresponding to the remaining nonzero entries in column c. However, they are the same as those obtained by using cofactor expansion along column c, and then row c of matrix $\mathbf{H}$ in (5), and thus $\det(\mathbf{H}_{c-1,c}) + \alpha = \det(\mathbf{Ht}_{c,c+1})$.

A similar technique can be used to demonstrate that $\det(\mathbf{H}_{c,c}) = \det(\mathbf{Ht}_{c+1,c+1}) + \beta$. Indeed, the respective determinants are as follows:

$$\det(\mathbf{H}_{c,c}) = \begin{vmatrix} x & 1 & & & & & & & \\ & x & 1 & & & & & & \\ & & \ddots & \ddots & & & & & \\ & & & x & 0 & & & & \\ & & & & x & 1 & \leftarrow & & \\ & & & & \Uparrow & x & 1 & & \\ & & & & & & \ddots & \ddots & \\ & & & & & & & x & 1 \\ 1 & & & & & & & & x \end{vmatrix}$$

$$\det(\mathbf{Ht}_{c+1,c+1}) = \begin{vmatrix} x & 1 & & & & & & & \\ & x & 1 & & & & & & \\ & & \ddots & \ddots & & & & & \\ & & & x & 1 & & & & \\ & & & \Rightarrow & x & 0 & & & \\ & & & & \uparrow & x & 1 & & \\ & & & & & & x & 1 & \\ & & & & & & & \ddots & \\ 1 & & & & & & & & x \end{vmatrix}. \quad (8)$$

The determinant of $\mathbf{Ht}_{c,c}$ can be obtained by using cofactor expansion along column c (double arrow), while the determinant of $\mathbf{Ht}_{c+1,c+1}$ is evaluated by cofactor expansion along row c. As earlier, deleting in both cases row c and column c yields the identical submatrices. The determinant of $\mathbf{Ht}_{c,c}$ may consist of additional components β, provided there are further nonzero entries in column c. However, they are equal to determinants, which are derived by using cofactor expansion along column $c+1$ of matrix $\mathbf{Ht}$, as shown in (6). Hence, we have the equality shown above.

Finally, we have to compute $d_{r,c} = \det(\mathbf{H}_{r,c})$ and $d_{r+1,c+1} = \det(\mathbf{Ht}_{r+1,c+1})$. After deleting the respective rows and columns in matrices $\mathbf{H}$ and $\mathbf{Ht}$ we have

$$d_{r,c} = \begin{vmatrix} x & 1 & & & & & & & & & \\ & x & \ddots & & & & & & & & \\ & & \ddots & 1 & & & & & & & \\ & & & x & \downarrow & & & & & & \\ & & & \Rightarrow & 1 & & & & & & \\ & & & & x & 1 & & & & & \\ & & & & & x & \ddots & & & & \\ & & & & & & \ddots & 1 & & & \\ & & & & & & \rightarrow & 0 & x & 1 & \\ & & & & & & & \Uparrow & x & \ddots & \\ & & & & & & & & & \ddots & 1 \\ 1 & & & & & & & & & & x \end{vmatrix}$$

$$d_{r+1,c+1} = \begin{vmatrix} x & 1 & & & & & & & & \\ & x & \ddots & & & & & & & \\ & & \ddots & 1 & & & & & & \\ & & & x & 1 & & & & & \\ & & & \Rightarrow & x & & & & & \\ & & & & \uparrow & 1 & & & & \\ & & & & & x & \ddots & & & \\ & & & & & & \ddots & 1 & \Downarrow & \\ & & & & & & \rightarrow & x & 1 & 0 \\ & & & & & & & & & x \\ & & & & & & & & & \ddots \\ & & & & & & & & & \ddots \\ 1 & & & & & & & & & \end{vmatrix} \quad (9)$$

The determinants of both matrices can be derived by using cofactor expansion along row c, and subsequently along column r as indicated by double arrows. After these two steps, rows c and r, as well as columns c and r, are deleted in both cases, which leaves two identical submatrices. Recall that this is possible as there are only zeros in row c and column r except entries on the main diagonal.

As can be seen, in all examined and corresponding cases we have arrived with the same values of the respective determinants. Thus, the characteristic polynomial of the transformed LFSR is the same as that of the original register, and therefore transformation EL preserves the transition function of the LFSR. This concludes the proof. ■

It is worth noting that the proof of Theorem 1 indicates the ability of transformation EL to handle cases in which an XOR gate of a given feedback connection crosses another XOR gate or a source tap of a given feedback connection crosses another source tap. These situations are illustrated in Figs. 3 and 4, respectively. During both transformations, the circuit maintains its original transition function. Likewise EL, transformation elementary shift to the right (ER) can also be employed. It moves the XOR gate from the output of a given storage device to the output of its successor, and relocates the source tap of the feedback line by one memory element, as well. Again, the following theorem holds.

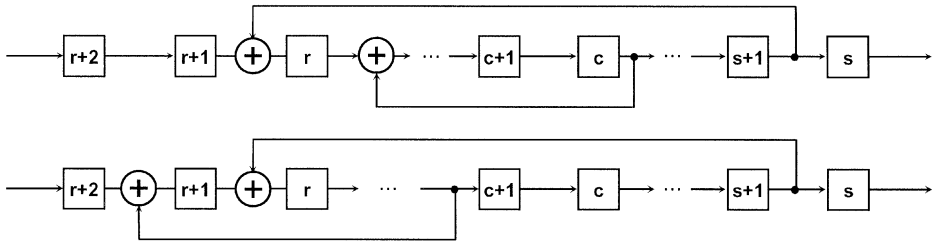

Fig. 3. Extension of transformation EL when crossing XOR gate.

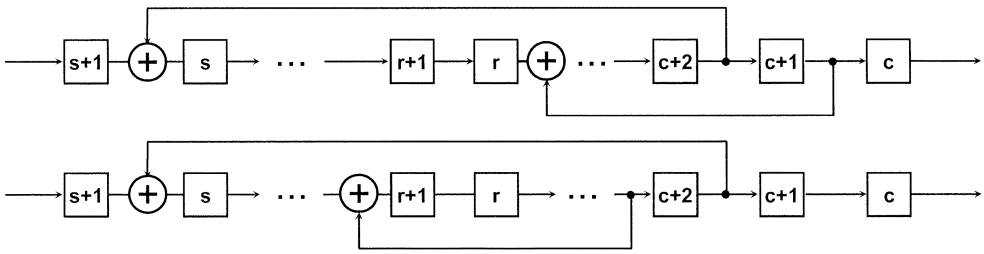

Fig. 4. Extension of transformation EL when crossing source tap.

Theorem 2: If transformation ER is applied to a feedback connection in such a way that neither its source tap crosses any XOR gate nor its XOR gate crosses another source tap, then the transition function of the LFSR is preserved.

Proof: Follows directly from the proof of Theorem 1. ■

III. Ring Generators

Transformations EL and ER can be employed to synthesize very fast linear finite state machines with a single level of XOR logic on any signal path, limited internal fanouts, and simplified overall routing. Consider a 32-bit internal feedback LFSR implementing the primitive polynomial $x^{32}+x^{18}+x^{14}+x^{9}+1$, as shown in Fig. 5(a). This LFSR has three feedback connections, all originating at the output of memory element 1. Note that the conventional, graphical layout of the circuit has been modified by forming a ring structure. This will allow us to demonstrate the most essential changes after applying the respective transformations. Fig. 5(b) illustrates a result of seven-step moving of all feedback lines in a counterclockwise direction. This is equivalent to applying transformation EL a number of times to all feedback connections. Subsequently, the feedback lines associated with coefficients x^9 and x^{14} are moved counterclockwise by two positions [Fig. 5(c)]. In a similar manner, the feedback line x^9 is further relocated by two flip-flops, which concludes the transformations. Fig. 5(d) shows the resultant maximum-length LFSR. Circuits similar to that one will be referred to as *ring generators*.

Advantages of the proposed transformations should now be apparent. First of all, only one two-input XOR gate occurs between designated flip-flops if a feedback logic is to be introduced. Second, the maximum internal fanout has been reduced by half from four devices fed by flip-flop 1 in the original circuit [Fig. 5(a)] to only two devices fed by any stem in the ring generator. Furthermore, there is a potential to reduce the total length of feedback lines. As Fig. 5 illustrates, if the register flip-flops could actually form an on-chip ring structure, then this particular benefit of using ring generators would be the best pronounced. Further examples of ring generators are illustrated in Fig. 6. The presented structures implement the following primitive polynomials: $x^{24}+x^{21}+x^{16}+x^{14}+x^{12}+x^{8}+1$ [Fig. 6(a)], $x^{32}+x^{27}+x^{14}+x^{12}+1$ [Fig. 6(b)], and $x^{32}+x^{28}+x^{23}+x^{20}+x^{17}+x^{12}+x^{8}+x^{4}+1$ [Fig. 6(c)]. They were obtained by using transformations applied to internal feedback LFSRs as described earlier. As can be seen, a typical ring generator has a smaller fanout than the original internal feedback LFSR. It also features a smaller number of levels of feedback logic than a corresponding external feedback LFSR. It is therefore well positioned to achieve higher performance than any other LFSR-based device used so far.

Every ring generator, including those presented above, can be obtained by using the following construction method.

Construction Method

Let the characteristic polynomial over GF(2) be $x^n + h_{n-1}x^{n-1} + \cdots + h_2x^2 + h_1x + 1$, where h_i is either zero or one. An n-bit ring generator implementing this polynomial can be constructed in such a way that if $h_i = 1$, $i = 1,\ldots n$, then locations S_i and D_i of source and destination taps of the corresponding feedback connections, respectively, are given by the following formulas:

$$S_i = 1 + \frac{(n-i)}{2} \qquad D_i = \frac{(n+i)}{2}. \tag{10}$$

As an example, consider polynomial $x^{32}+x^{27}+x^{14}+x^{12}+1$. Equation (10) yields the following architecture of the corresponding ring generator: $\mathrm{S}_{12} = 1 + (32-12)/2 = \mathbf{11}$, $\mathrm{D}_{12} = (32+12)/2 = \mathbf{22}$, $\mathrm{S}_{14} = 1 + (32-14)/2 = \mathbf{10}$, $\mathrm{D}_{14} = (32+14)/2 = \mathbf{23}$, $\mathrm{S}_{27} = 1 + (32-27)/2 = \mathbf{3}$, $\mathrm{D}_{3} = (32+27)/2 = \mathbf{29}$. A correspondence between these numbers and the actual implementation can be easily found in Fig. 6(b).

It is worth noting that there are characteristic polynomials for which internal loads cannot be reduced (taking an internal feedback LFSR as a reference) without expanding the XOR logic.

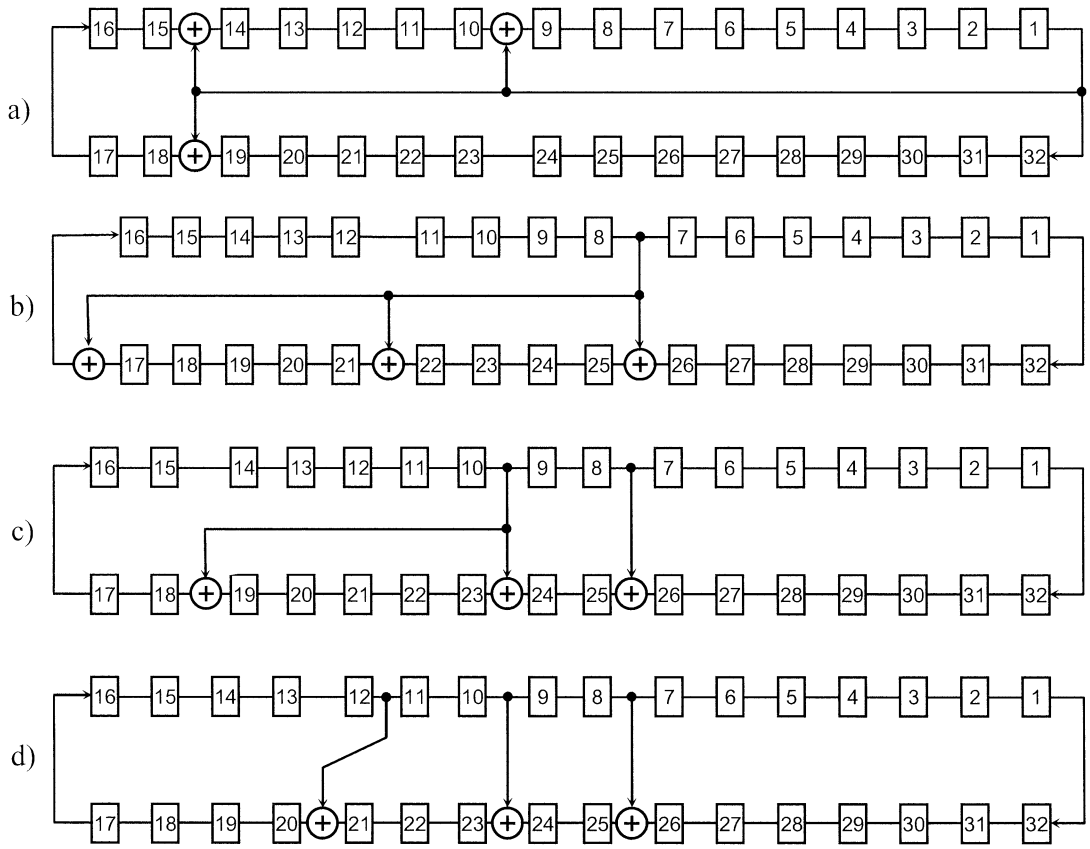

Fig. 5. LFSRs implementing polynomial $x^{32} + x^{18} + x^{14} + x^9 + 1$.

This is because several nonzero coefficients of the characteristic polynomial may be associated with consecutive powers of x. The primitive polynomial $x^8 + x^4 + x^3 + x^2 + 1$ may serve as a good example here. The resultant ring generator, obtained by virtue of the construction method, would have the following structure: (4,5), (3,5), (3,6). Noticeable congestions represented by the overloaded output of flop 3 as well as two XOR gates appearing on the input of flop 5, clearly deteriorate the overall performance of such a circuit. Furthermore, if a separation between successive feedback terms is not large enough, then even optimal ring generators are not suitable for a wide class of applications where high degree of modularity is required. Indeed, having characteristic polynomials with scattered feedback taps would allow one to design large generators that could be easily integrated with environments such as systems-on-a-chip, multiple embedded cores, and others based on similar design paradigms [15], [20]. An automated synthesis of ring generators should rest, therefore, on characteristic primitive polynomials with uniformly distributed feedback taps, i.e., polynomials that offer virtually the same separation between any two consecutive feedback taps. An example of such a primitive polynomial would be $x^{72}+x^{64}+x^{55}+x^{45}+x^{37}+x^{27}+x^{18}+x^9+1$, which is in marked contrast to the primitive polynomial $x^{72} + x^{49} + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$ of the same degree.

For reasons stated above, we launched a large search program with the objective of finding a comprehensive list of primitive polynomials with uniformly scattered coefficients. Using a technique described in [22], primitive polynomials featuring 5, 7, and 9 terms were identified for all degrees up to 660. They can be found in [20]. All polynomials listed in [20] are optimal in the sense of having the most uniformly distributed taps among all primitive polynomials.

IV. Comparative Analysis

Performance and structural properties of internal and external feedback LFSRs, hybrid LFSRs [24], linear CAs, and the proposed ring generators are compared in this section. The following criteria were adopted to compare various classes of generators: the number of XOR gates required to implement a given structure, the number of levels of the feedback logic, the size of internal fanouts, and degree of modularity. In all cases, it is assumed that a generator is to implement a characteristic polynomial of degree n featuring k feedback coefficients (coefficients x^n and x^0 are not counted).

A. Number of Two-Input XOR Gates

Clearly, internal and external feedback LFSRs require k two-input XOR gates. The same applies to the basic form of ring generators. Linear CAs with null boundary conditions may need two XOR gates per each slice except the boundary stages, if implementation of a given polynomial consists of cells 150 only, i.e., $x_c(t+1) = x_{c-1}(t)+x_c(t)+x_{c+1}(t)$, for $c = 2, \ldots, n-1$.

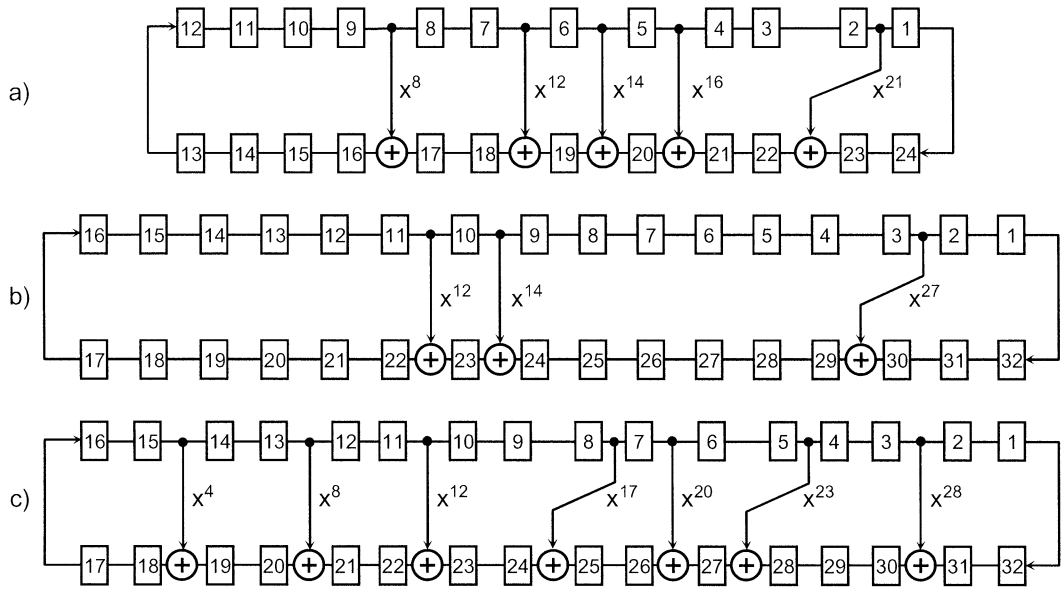

Fig. 6. Examples of ring generators.

In such a case, the total number of gates is equal to $2n-2$. As demonstrated in [24], the number of two-input XOR gates occurring in hybrid LFSRs amounts to $(k+1)/2$. It should be noted, however, that this result remains valid only for certain polynomials (see also Section V).

B. Levels of Logic

The effective speed of each class of generators is determined in terms of the number of two-input XOR gates through which signals have to propagate. Since the feedback logic of external feedback LFSRs can be implemented as a balanced XOR tree, the actual number of levels of logic does not exceed the value of $\log_2 k$. Internal feedback LFSRs have their XOR gates interspersed between memory elements. Thus, they are expected to be faster than external feedback LFSRs. However, long feedback lines can effectively slow down the whole circuit [10]. Performance of hybrid LFSRs is typically placed between the aforementioned quantities. Local neighborhood interactions in CA limit propagation delays to two XOR gates. Indeed, at least one cell 150 must be employed as there are no maximum length CAs featuring rule 90 exclusively [6]. Clearly, the ring generator is the fastest solution as it has only one two-input XOR gate delay and its short feedback lines do not cause frequency degradation.

C. Internal Fanouts

Since the output from any switching device has a definite limit to the amount of current it can supply, there is also a limit to the number of other devices that can be driven by a single output from that switch. In order to meet fanout requirements, buffers are used to boost and sharpen signals that might otherwise degrade below switching levels or be distorted. These buffers, however, may further deteriorate performance of the entire generator, and therefore the presence of large fanouts has to be regarded as disadvantageous. In light of this, internal feedback LFSRs are particularly susceptible to delays of this type because $k+1$ fanout branches occur on the output of one of their memory elements. On the other hand, external feedback LFSRs and ring generators have internal fanouts confined to two branches.

TABLE I
BASIC PARAMETERS OF LINEAR CIRCUITS

	XOR gates	Levels of logic	Fan-out
Internal feedback LFSR	k	$\log_2 k$	2
External fedback LFSR	k	1	$k+1$
Hybrid LFSR	$(k+1)/2$	$(1, \log_2 k)$	$(2, k+1)$
Cellular automata	$2n-2$	2	3
Ring generators	k	1	2

For the sake of completeness, all quantities discussed above are gathered in Table I. As can be seen, the superiority of the ring generators is pronounced clearly. They are the fastest devices, which, simultaneously, provide the best tradeoffs between the crucial design factors.

D. Modularity

The ring generators (and circuits they originate from) can also be characterized by their degree of modularity. For designs with a large number of scan chains, employing a ring generator may result in a large number of wires that would have to be routed across the entire chip. To alleviate this problem, different segments of a ring generator can be used to drive scan chains in different regions of the circuit. The placement of the generator can be accomplished easily provided a clean separation and a simple interface between successive components of the generator are defined. It is, therefore, appropriate to assume that any connection (a wire) in a generator can be used as a cut between candidate segments provided it links only two memory elements, and it does not feature any extra fanouts. The entire circuit can be then represented by means of a multigraph (i.e., having multiple edges between vertices) such that each vertex corresponds to one segment consisting of certain memory elements and XOR gates. Among various designs, the one whose multigraph conforms best to a regular graph (i.e., a graph with all vertices

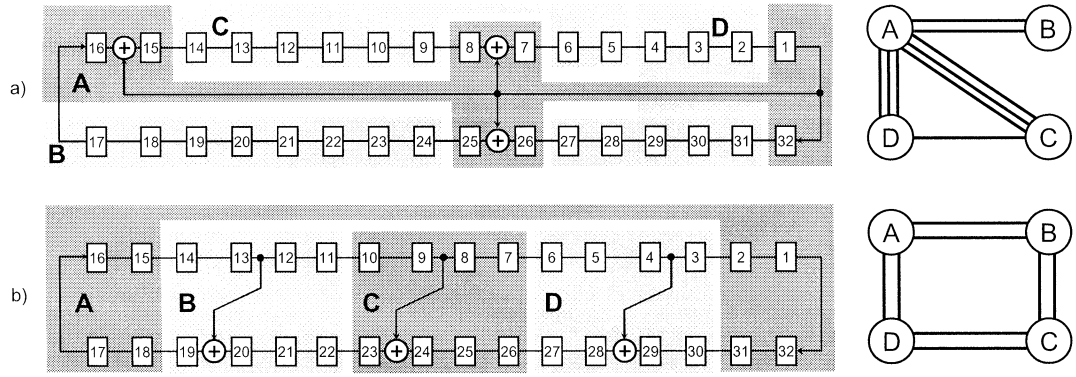

Fig. 7. Modularity – four segments.

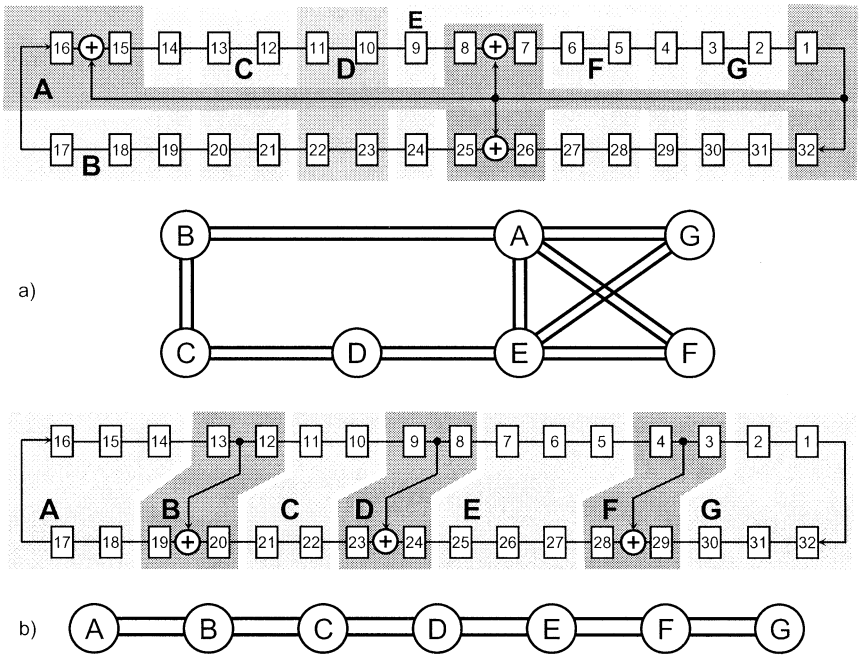

Fig. 8. Modularity – seven segments.

having the same number of touching edges) is regarded as the most modular one.

Figs. 7 and 8 illustrate an internal feedback LFSR and the corresponding ring generator, both implementing the primitive polynomial $x^{32}+x^{25}+x^{15}+x^{7}+1$, and both subject to partitions into four and seven, approximately equally sized modules, respectively. Their multigraphs are also shown. As can be seen in Fig. 7(b), the ring generator forms a regular graph, while its LFSR counterpart has a much less regular structure [Fig. 7(a)]. Similarly, partition of the registers into seven modules results in structures shown in Fig. 8, with the ring generator again featuring a much higher regularity. Here, the objective was to keep the number of wires between any pair of modules no bigger than two. Clearly, one may arrive with different partitions (and thus different multigraphs). Typically, however, the ring generators are much more amenable to implementing modular configurations than other linear finite state machines.

V. Further Applications

Besides acting as sources of pseudorandom test patterns, ring generators can also be employed to perform embedded decompression of test patterns [21] or to act as test response compactors by assuming the role of a MISR [5]. In these applications, external data are provided to the register as compressed deterministic test patterns or test responses. In order to accept test data in parallel, several inputs together with additional XOR gates have to be placed between the register flip-flops, as shown in Fig. 9(a) for a 32-bit ring compactor having 16 parallel inputs and using the primitive polynomial $x^{32}+x^{25}+x^{15}+x^{7}+1$. In many cases, additional XOR gates may deteriorate the circuit performance as they introduce extra delays between consecutive stages of the register (see the input of flip-flop 28 in Fig. 9(a). Fortunately, this problem can be resolved by applying transformations to further adjust a feedback tap configuration.

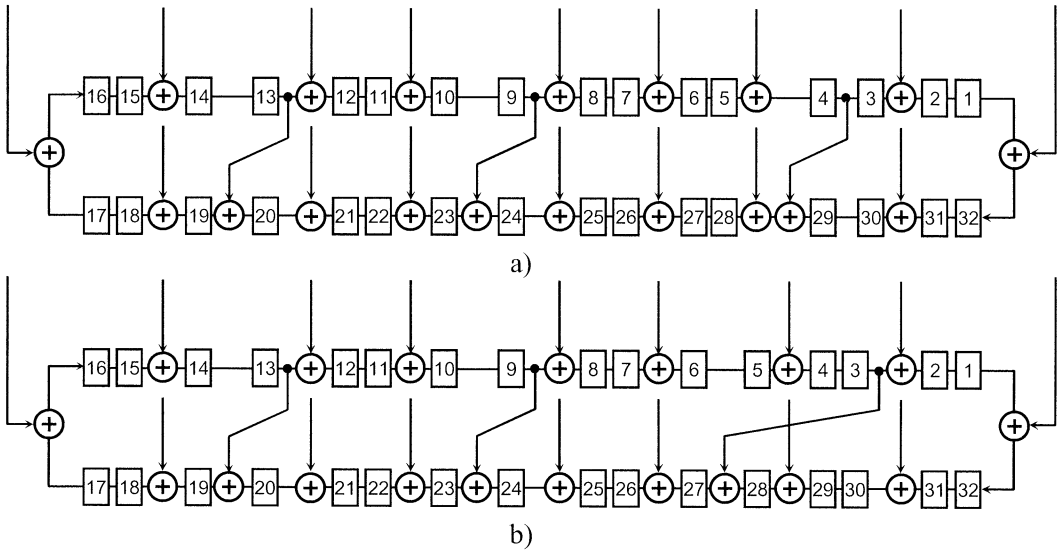

Fig. 9. Ring generator driven by the external test data.

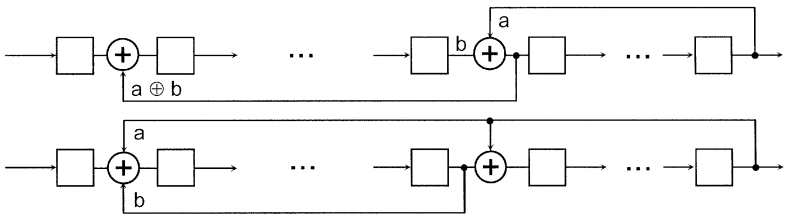

Fig. 10. Transformation SDL.

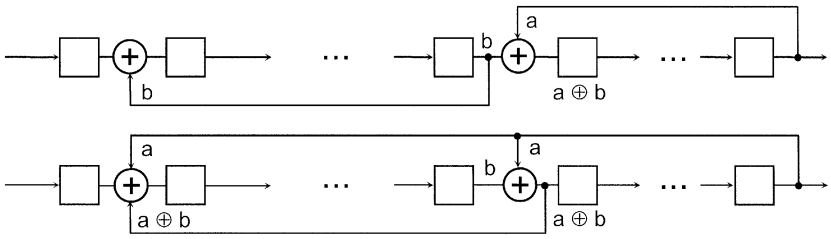

Fig. 11. Transformation SDR.

This way, one can avoid undesired expansion of the XOR logic at the inputs of some flip-flops. For instance, using the transformations presented in the previous sections, a feedback structure of the circuitry shown in Fig. 9(a) can now be rearranged so that a destination tap and a parallel input do not collide at the input of flip-flop 28, as presented in Fig. 9(b) for the previously mentioned characteristic polynomial. As a result, a single two-input XOR gate appears at the input of every flip-flop implementing either a feedback tap or a parallel input.

As shown in the previous sections, transformations EL and ER are not applicable when an XOR gate is to cross a source tap of another feedback connection or vice versa. In the following, we describe four transformations that lift this constraint, while still using the simple paradigm of moving feedback lines across memory elements. The most crucial feature of these transformations is the ability to add a new feedback line to maintain a transition function of the original LFSR. The first transformation is used when a source tap crosses a destination XOR gate while shifting to the Left (SDL), as shown in Fig. 10.

Theorem 3: Transformation SDL preserves the transition function of a given LFSR.

Proof: From Fig. 10, it follows that once the source tap crosses the XOR gate, data carried by the corresponding feedback line are not equivalent to $a \oplus b$ as it has become equal to b. Hence, in order to restore the former functionality on the output of the destination XOR gate, an extra copy of symbol a must be provided from the place of its origin by forming a new feedback line. ■

It is worth noting that symbol a can represent several feedback paths reaching their destination at this particular gate. If this is the case, all such feedback connections should be extended as required by transformation SDL. This observation applies to the remaining transformations as well.

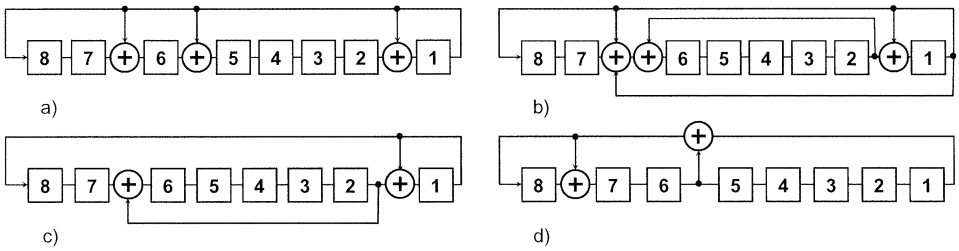

Fig. 12. Reduction of XOR gates for polynomial $x^8 + x^6 + x^5 + x + 1$.

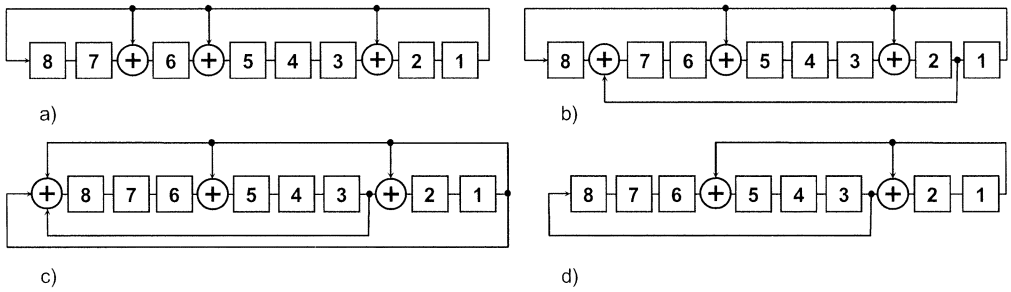

Fig. 13. Reduction of XOR gates for polynomial $x^8 + x^6 + x^5 + x^2 + 1$.

Graphical summary of transformation employed when a source tap crosses a destination XOR gate while shifting to the Right (SDR) is given in Fig. 11.

Theorem 4: Transformation SDR preserves the transition function of a given LFSR.

Proof: Initially, both feedback connections involved in this operation do not overlap. In fact, the feedback line to be shifted to the right begins at the output of the flip-flop feeding the other XOR gate. Therefore, the output of the gate is equal to $a \oplus b$. Next, the source tap crosses the XOR gate, thus changing functionality of the circuit. Again, to restore the former value on the output of the second XOR gate, an extra feedback connection has to be established, as shown in Fig. 11. This step will compensate the presence of variable a by taking advantage of the relationship $a \oplus b \oplus a = b$. ■

The rules that govern two remaining transformations, i.e., destination XOR gate crosses a source tap while shifting to the left (DSL) and destination XOR gate crosses a source tap while shifting to the right (DSR) are similar to those of transformations SDR and SDL, respectively. Their correctness follows directly from Figs. 10 and 11 where the shift operations should be carried out in the opposite direction compared to SDR and SDL.

Transformations presented in this section can be used to change the number of XOR gates in a feedback network and to cancel some of the existing connections. Consider an internal feedback LFSR implementing primitive polynomial $x^8 + x^6 + x^5 + x + 1$. Its original canonical architecture is shown in Fig. 12(a). Applying transformation SDL to the feedback connection represented by coefficient x^5 leads to the circuit shown in Fig. 12(b). According to the definition of SDL, an extra feedback has to be added by placing an additional XOR gate on the input of flip-flop 6. Since another XOR gate with the same connectivity already exists on the input of this memory element, effectively both gates cancel each other, and the total number of XOR gates is now two rather than three [Fig. 12(c)]. It is worth noting that a corresponding hybrid LFSR [24] can also be constructed using two XOR gates as shown in Fig. 12(d). However, this solution introduces one more XOR gate delay compared to the transformed circuit of Fig. 12(c). In general, hybrid LFSRs using a given number of XOR gates have the operating speed lower than the transformed LFSRs featuring the same number of gates.

We have examined all primitive polynomials of rank up to 24 featuring five coefficients (pentanomials). The following observations are drawn from this analysis:

- several primitive pentanomials, which can be implemented as hybrid LFSRs, can also be transformed as shown in this section; the new solutions offer higher operating speeds as they feature only a single XOR gate on any signal propagation path;
- there are primitive pentanomials that cannot be implemented as hybrid LFSRs, whereas they still have realizations featuring only two XOR gates.

To support the last observation, consider the problem of transforming an internal feedback LFSR with primitive polynomial $x^8+x^6+x^5+x^2+1$ [Fig. 13(a)]. This LFSR cannot be converted to a hybrid structure as demonstrated in [24]. However, by applying transformation EL to the feedback connection x^6, we obtain the structure shown in Fig. 13(b). Subsequently, transformation SDL applied to the same connection link creates an additional feedback line spanning the whole register [Fig. 13(c)]. Since the same connection already exists, they both cancel each other [Fig. 13(d)], thus reducing the number of XOR gates.

Since primitive pentanomials amenable to implementing with two XOR gates are not confined to those discussed in [24], a complete characterization of this class of polynomials is provided in Table II. It lists, in the row labeled as [24], the number of primitive pentanomials which can be implemented as hybrid LFSRs. Subsequently, the number of primitive pentanomials that can be treated only by the method presented in this paper are shown in

TABLE II
NUMBER OF PRIMITIVE PENTANOMIALS IMPLEMENTABLE WITH TWO XOR GATES

n	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
[24]	6	8	4	18	4	20	8	16	10	34	6	28	20	30	24	66	8
New	3	5	5	9	2	9	8	16	6	27	7	13	9	12	18	35	4
All	12	16	20	44	17	66	42	82	52	152	72	158	100	164	122	292	94

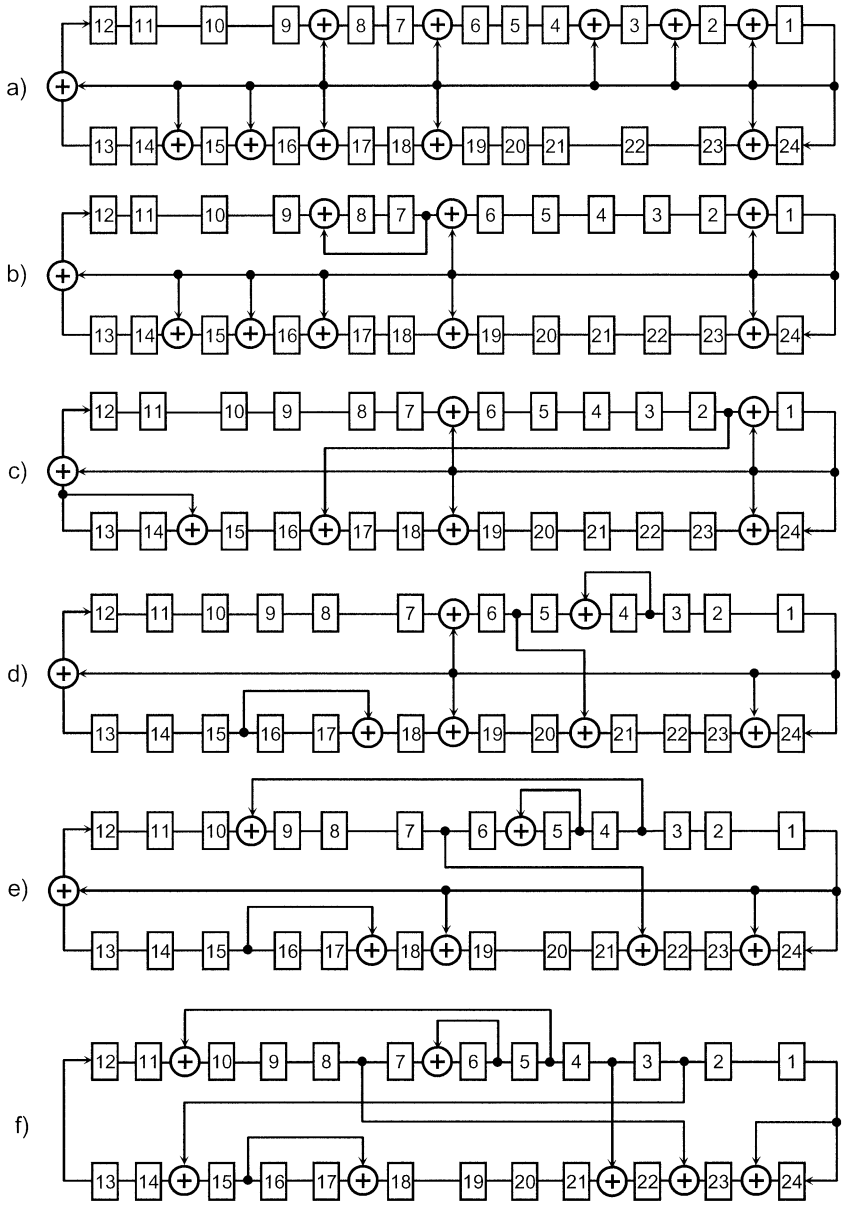

Fig. 14. Implementation of polynomial $x^{24}+x^{23}+x^{18}+x^{16}+x^{15}+x^{14}+x^{12}+x^{8}+x^{6}+x^{3}+x^{2}+x+1$.

the row "New." The last row of the table gives the total number of primitive pentanomials for each value of rank n.

Extending the last two examples, a construction method for synthesis of transformed LFSRs with only two XOR gates is given below.

Construction Method

Let the primitive pentanomial over GF(2) be $x^n + x^c + x^b + x^a + 1$, where $a < b < c < n$. An n-bit transformed LFSR implementing this polynomial can be constructed in such a way that locations (S, D) of source and destination taps, respectively, of the corresponding feedback connections are given by the following formulas:

- if $a+b=c$, then $(1,a)$, $(a+1,c)$, and there is link $(1,n)$, or alternatively $(1,b)$, $(b+1,c)$, and there is link $(1,n)$; note that this is the case in which a respective hybrid LFSR can also be constructed.
- if $a+c=n$, then $(1,a)$, $(1,b)$, and there is link $(n-c+1,n)$.

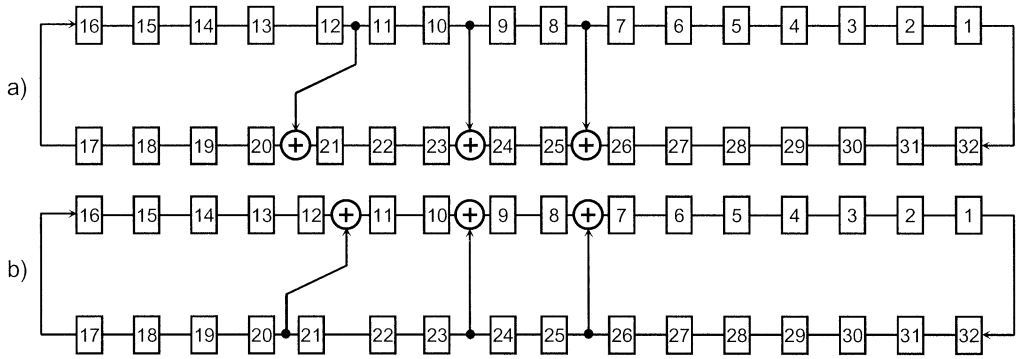

Fig. 15. Ring generator and its dual form.

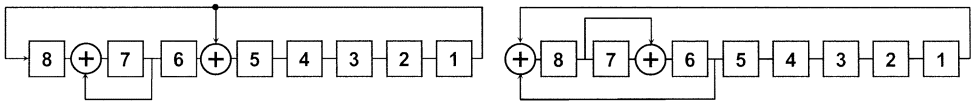

Fig. 16. Transformed LFSR and its dual form.

- if $a + b = n$, then $(1, a)$, $(1, c)$, and there is link $(n - b + 1, n)$, or alternatively $(1, b)$, $(1, c)$, and there is link $(n - a + 1, n)$.

Recall that locations (S, D) correspond to the output and the input of the indicated memory elements, respectively.

As a final example, consider primitive polynomial $x^{24}+x^{23}+x^{18}+x^{16}+x^{15}+x^{14}+x^{12}+x^8+x^6+x^3+x^2+x+1$. In this case, an internal feedback LFSR implementation uses as many as eleven two-input XOR gates [Fig. 14(a)]. The conversion then proceeds as shown in Fig. 14(b) – (f), eventually resulting in the XOR gate count reduced to seven (implementation with six XOR gates is also possible, though in such case it is not possible to maintain the internal fanouts at the level of two for any stem in the circuit).

VI. Synthesis of Associated Phase Shifters

Similar to conventional LFSRs, the ring generators are not free from structural dependencies and linearly dependent positions in their output sequences. Thus, the use of ring generators to feed scan chains directly from flip-flops not separated by any XOR gates will cause these scans to contain correlated test patterns, which may adversely compromise fault coverage [17]. It becomes imperative, therefore, to eliminate such dependencies and to allow a large number of scan chains to be driven by relatively short ring generators. This can be accomplished by placing phase shifters between the generators and the serial inputs of the scan chains [4], [12], [13], [19], [22]. Every scan chain is then driven by a circuitry, which corresponds to a linear combination of generator outputs. Given a ring generator, logic synthesis of phase shifters entails finding a set of linear combinations of the generator outputs (called XOR taps) such that each of the resulting sequences, obtained by adding these stages, will be shifted with respect to every other sequence by at least a prespecified number of bits.

It turns out that large and fast phase shifters can be obtained by selecting XOR tap combinations randomly [17], and then by checking if a sequence produced by a linear combination of newly selected XOR taps does not overlap with sequences generated on already existing outputs. The second step is discussed in this section. The proposed method generalizes the concept of duality, previously employed to obtain phase shifters for external and internal feedback LFSRs [19].

Given the structure of a ring generator as shown in Fig. 15(a), its dual form is derived by reversing the direction of all feedback connections [see Fig. 15(b)]. Hence, a dual ring generator features XOR gates placed on the outputs of those flip-flops that have been used to drive feedback taps in the original circuit, while the feedback lines originate now at the former locations of the respective XOR gates. In other words, an original feedback connection between flip-flops S and D are replaced in the dual generator by a feedback line between flip-flops $D-1$ and $S+1$. A fundamental relationship between the two forms of ring generators is summarized by the following theorem.

Theorem 5: If $\mathbf{R}$ is a characteristic matrix of an n-bit ring generator, and $\mathbf{D}$ is a characteristic matrix of the corresponding dual ring generator, then $\mathbf{R}^{\mathrm{T}} = \mathbf{D}^{-1}$, where $\mathbf{R}^{\mathrm{T}}$ is the transpose of $\mathbf{R}$, and $\mathbf{D}^{-1}$ is the inverse of $\mathbf{D}$.

Proof: It can be easily verified that matrix $\mathbf{D}$ features ones along the diagonal above the main diagonal, and a one in row n and column 1, while matrix $\mathbf{R}^{\mathrm{T}}$ contains ones along the diagonal below the main diagonal plus a one in row 1 and column n. Thus, matrix $\mathbf{D} \cdot \mathbf{R}^{\mathrm{T}}$ features ones located on the main diagonal. In fact, these are the only nonzero entries of this matrix. To prove it, consider a nonzero entry in row x and column y of $\mathbf{R}$, or alternatively a one in row y and column x of $\mathbf{R}^{\mathrm{T}}$. The

TABLE III
SYNTHESIS OF PHASE SHIFTER BY MEANS OF LOGIC SIMULATION

	States of original generator	States of dual generator	*m*-sequences													
			1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	1 0 0 0 0 0 0 0	1 0 0 0 0 0 0 0	0	1	1	1	0	0	0	1	1	1	0	1	1	1
1	0 1 0 0 0 0 0 0	0 1 1 0 0 0 0 0	1	0	1	1	1	0	0	0	1	1	1	0	1	1
2	0 1 1 0 0 0 0 0	1 0 1 1 0 0 0 0	0	1	0	1	1	1	0	0	0	1	1	1	0	1
3	0 1 1 1 0 0 0 0	1 1 1 1 1 0 0 0	0	0	1	0	1	1	1	0	0	0	1	1	1	0
4	0 1 1 1 1 0 0 0	1 1 0 1 1 1 0 0	0	0	0	1	0	1	1	1	0	0	0	1	1	1
5	0 1 1 1 1 1 0 0	0 1 0 0 1 1 1 0	0	0	0	0	1	0	1	1	1	0	0	0	1	1
6	0 1 1 1 1 1 1 0	0 0 1 0 0 1 1 1	0	0	0	0	0	1	0	1	1	1	0	0	0	1
7	0 1 1 1 1 1 1 1	0 0 0 1 0 0 1 1	0	0	0	0	0	0	1	0	1	1	1	0	0	0
8	1 1 1 0 1 1 1 1	1 0 0 0 1 0 0 1	0	0	0	0	0	0	0	1	0	1	1	1	0	0
9	1 0 1 0 0 1 1 1	1 1 1 0 0 1 0 0	1	0	0	0	0	0	0	0	1	0	1	1	1	0
10	1 1 0 0 0 0 1 1	1 1 0 1 0 0 1 0	1	1	0	0	0	0	0	0	0	1	0	1	1	1
11	1 0 1 1 0 0 0 1	0 1 0 0 1 0 0 1	1	1	1	0	0	0	0	0	0	0	1	0	1	1
12	1 1 0 0 1 0 0 0	1 0 1 0 0 1 0 0	1	1	1	1	0	0	0	0	0	0	0	1	0	1
13	0 0 1 0 0 1 0 0	1 1 1 1 0 0 1 0	1	1	1	1	1	0	0	0	0	0	0	0	1	0
14	0 0 0 1 0 0 1 0	1 1 0 1 1 0 0 1	0	1	1	1	1	1	0	0	0	0	0	0	0	1
15	0 0 0 0 1 0 0 1	1 1 0 0 1 1 0 0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
16	1 0 0 1 0 1 0 0	0 1 0 0 0 1 1 0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
17	0 1 0 0 1 0 1 0	0 0 1 0 0 0 1 1	1	0	0	0	1	1	1	1	1	0	0	0	0	0
18	0 1 1 0 0 1 0 1	0 0 0 1 0 0 0 1	0	1	0	0	0	1	1	1	1	1	0	0	0	0
19	1 1 1 0 0 0 1 0	1 0 0 0 1 0 0 0	0	0	1	0	0	0	1	1	1	1	1	0	0	0

corresponding one in matrix $\mathbf{D}$ occurs in row $y-1$ and column $x+1$, i.e.

$$\mathbf{R} = \begin{array}{c} \\ x \end{array} \begin{array}{c} \begin{array}{c} y \end{array} \\ \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & \vdots & \ddots & & & \\ & \rightarrow & 1 & \cdots & 1 & & \\ & & \uparrow & & & \ddots & \\ & & & & & & 1 \\ 1 & & & & & & \end{bmatrix} \end{array}$$

$$\mathbf{D} = \begin{array}{c} \\ y-1 \\ x \end{array} \begin{array}{c} \begin{array}{cc} y & x+1 \end{array} \\ \begin{bmatrix} 1 & & & & & \\ & \ddots & & & \downarrow & \\ & & 1 & \cdots & 1 & \leftarrow \\ & & & \ddots & \vdots & \\ & & & & 1 & \\ & & & & & \ddots \\ 1 & & & & & 1 \end{bmatrix} \end{array}$$

$$\mathbf{R}^{\mathrm{T}} = \begin{array}{c} \\ y \\ x+1 \end{array} \begin{array}{c} \begin{array}{c} x \end{array} \\ \begin{bmatrix} & & & & & 1 \\ 1 & & & & & \\ & \ddots & & \downarrow & & \\ & & 1 & \cdots & 1 & \leftarrow \\ & & & \ddots & \vdots & \\ & & & & 1 & \\ & & & & & \ddots \\ & & & & & 1 \end{bmatrix} \end{array}.$$

Consider an entry in row $y-1$ and column x of $\mathbf{D}\cdot\mathbf{R}^{\mathrm{T}}$. To determine its value, one has to multiply entries in row $y-1$ of $\mathbf{D}$ and column x of $\mathbf{R}^{\mathrm{T}}$. That is, as it follows from the above matrices, we have $\mathbf{D}_{y-1,y}\cdot\mathbf{R}^{\mathrm{T}}_{y,x} + \mathbf{D}_{y-1,x+1}\cdot\mathbf{R}^{\mathrm{T}}_{x+1,x} = 1\cdot 1 + 1\cdot 1 = 0$. These computations may involve other feedback coefficients, but the corresponding equations will always have a form similar to that above, i.e., it will consist of two nonzero components. Hence, all entries of $\mathbf{D}\cdot\mathbf{R}^{\mathrm{T}}$ but those on the main diagonal will assume the value of zero. Since $\mathbf{D}\cdot\mathbf{R}^{\mathrm{T}} = \mathbf{I}$, the equation $\mathbf{R}^{\mathrm{T}} = \mathbf{D}^{-1}$ is satisfied, and this concludes the proof. ■

It was shown [19] that if for a characteristic matrix $\mathbf{R}$ of an LFSR the relation $\mathbf{R}^{\mathrm{T}} = \mathbf{D}^{-1}$ holds, then the state of the dual generator after q clock cycles will indicate outputs of the original generator that have to be added to produce a string of bits spaced q shifts up the reference sequence, i.e., the sequence originating from a stage designated by the initial state of the dual circuit. As a result, logic simulation of dual ring generators can be used to carry out synthesis of phase shifters as demonstrated by the following example.

Consider an 8-bit transformed LFSR shown in Fig. 16. Operation of this circuit is illustrated in the second column of Table III for 19 clock cycles. Successive states of its dual form are listed in the next column of the same table. Some of the resulting m-sequences obtained by adding respective outputs of the circuit, as indicated by the states of its dual counterpart, are gathered in the fourth column of Table III. The resulting phaseshifts are shown in the header of this column. For instance, the state of dual generator occurring in the row "7," i.e., combination 00 010 011 points out that if stages 5, 2, and 1 of the original generator are added, then the resulting m-sequence will be shifted by seven bits. It can be easily verified that all presented sequences are indeed shifted by the desired number of bits (see the header) with regard to the m-sequence observed on the left-most bit of the original circuit.

As mentioned earlier, the above technique can be used to check if a sequence produced by a linear combination of newly selected XOR taps does not overlap with sequences generated on other outputs [22]. Let d be the required minimum separation. Clearly, if there is no overlap, then from a binary

combination β representing the new XOR taps onwards, for at least the next d clock cycles, there is no combination of XOR taps already included into a phase-shifter network. Similarly, accepted combinations should not occur among, at least, d successive states preceding state β in the state trajectory of the dual generator.

VII. Conclusion

In this paper, we have introduced a novel methodology of designing generators and compactors of test data. The essence of the proposed approach is to use a set of transformations which alter the structure of the conventional LFSRs, while preserving their transition function. It is shown that after applying these transformations in a certain order, the resultant devices feature significantly improved structural properties and remarkably enhanced overall performance, as compared to previous schemes based on LFSRs and CAs. Major advantages of the proposed technique include improving quality of the implementation, which is achieved by reducing the number of levels of XOR logic, reducing the internal fanouts, and simplifying the circuit layout and routing. Consequently, one can synthesize highly modular devices that can operate at higher speeds than conventional solutions. Yet, in many cases, it is possible to change the number of XOR gates and to add or delete feedback connections. It may result, as in the CAs case [6], in an enhanced randomness of the produced patterns. More importantly, however, the obtained realizations always correspond to characteristic polynomials of the original circuits, thus creating a large implementation domain for linear finite-state machines of a given size and the same transition function. Clearly, the proposed approach creates a robust framework for computer-aided design tools to explore the broad domain of possible solutions and different design tradeoffs.

References

[1] V. D. Agrawal, C. R. Kime, and K. K. Saluja, "A tutorial on built-in self-test, Part 1: Principles," *IEEE Design Test Comput.*, vol. 10, pp. 73–82, Mar. 1993.

[2] ——, "A tutorial on built-in self-test, Part 2: Applications," *IEEE Design Test Comput.*, vol. 10, pp. 69–77, June 1993.

[3] A. C. Arvillias and D. G. Maritsas, "Toggle-registers generating in parallel kth decimations of M-sequences $X^{\mathrm{r}} \otimes X^{\mathrm{i}} \otimes 1$: Design tables," *IEEE Trans. Comput.*, vol. 28, pp. 89–101, Feb. 1979.

[4] P. H. Bardell, "Design considerations for parallel pseudorandom pattern generators," *J. Electron. Testing: Theory Applicat.*, vol. 1, no. 1, pp. 73–87, 1990.

[5] P. H. Bardell, W. H. McAnney, and J. Savir, *Built-In Test for VLSI: Pseudorandom Techniques*. New York: Wiley, 1987.

[6] P. P. Chaudhuri, D. R. Chowdhury, S. Nandi, and S. Chattopadhyay, *Additive Cellular Automata*. Los Alamitos, CA: IEEE Comput. Soc.Press, 1997.

[7] R. David, *Random Testing of Digital Circuits*. New York: Marcel Dekker, 1998.

[8] C. Dufaza and Y. Zorian, "On some theoretical properties of LFSRs for BIST applications," in *Proc. Int. On-Line Testing Workshop*, 1997, pp. 184–189.

[9] S. W. Golomb, *Shift Register Sequences*. Launa Hills, CA: Aegean Park Press, 1982.

[10] K. Hatayama, M. Nakao, Y. Kiyoshige, and K. Natsume, "Application of high-quality built-in test to industrial designs," in *Proc. ITC*, 2002, pp. 1003–1012.

[11] W. J. Hurd, "Efficient generation of statistically good pseudonoise by linearly interconnected shift registers," *IEEE Trans. Comput.*, vol. 23, pp. 146–152, Feb. 1974.

[12] B. Ireland and J. E. Marshall, "Matrix method to determine shift-register connections for delayed pseudorandom binary sequences," *Electron. Lett.*, vol. 4, no. 15, pp. 309–310, 1968.

[13] K. J. Latawiec, "New method of generation of shifted linear pseudorandom binary sequences," *Proc. IEE*, vol. 121, pp. 905–906, Aug. 1974.

[14] A. Lempel and W. L. Eastman, "High speed generation of maximal length sequences," *IEEE Trans. Comput.*, vol. 20, pp. 227–229, Feb. 1971.

[15] E. J. Marinissen, R. Kapur, and Y. Zorian, "On using IEEE P1500 SECT for test plug-n-play," in *Proc. ITC*, 2000, pp. 770–777.

[16] S. Mourad and Y. Zorian, *Principles of Testing Electronic Systems*. New York: Wiley, 2000.

[17] G. Mrugalski, J. Rajski, and J. Tyszer, "Linear independence as evaluation criterion for two-dimensional test pattern generators," in *Proc. VLSI Test Symp.*, 2000, pp. 377–386.

[18] ——, "High speed ring generators and compactors of test data," in *Proc. VLSI Test Symp.*, 2003, pp. 57–62.

[19] J. Rajski and J. Tyszer, "Design of phase shifters for BIST applications," in *Proc. VLSI Test Symp.*, 1998, pp. 218–224.

[20] ——, "Primitive polynomials over GF(2) of degree up to 660 with uniformly distributed coefficients," *J. Electron. Testing: Theory Applicat.*, vol. 19, no. 6, pp. 645–657, December 2003.

[21] J. Rajski, J. Tyszer, M. Kassab, N. Mukherjee, R. Thompson, H. Tsai, A. Hertwig, N. Tamarapalli, G. Mrugalski, G. Eide, and J. Qian, "Embedded deterministic test for low cost manufacturing test," in *Proc ITC*, 2002, pp. 301–310.

[22] J. Rajski, N. Tamarapalli, and J. Tyszer, "Automated synthesis of phase shifters for built-in self-test applications," *IEEE Trans. Computer-Aided Design*, vol. 19, pp. 1175–1188, Oct. 2000.

[23] L.-T. Wang and E. J. McCluskey, "Circuits for pseudo-exhaustive test pattern generation," in *Proc. ITC*, 1986, pp. 25–37.

[24] ——, "Hybrid designs generating maximum-length sequences," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 91–99, Jan. 1988.

[25] W. W. Warlick and J. E. Hershey, "High-speed M-sequence generators," *IEEE Trans. Comput.*, vol. 29, pp. 398–400, May 1980.

[26] V. N. Yarmolik and S. N. Demidenko, *Generation and Application of Pseudorandom Sequences for Random Testing*. New York: Wiley, 1988.

Grzegorz Mrugalski received the M.S. and Ph.D. degrees in electrical engineering from the Poznań University of Technology, Poznań, Poland, in 1995 and 2002, respectively.

He spent several months working for Ecole d'Ingenieurs en Communications, Lille, France, and the University of Vigo, Spain. In September 2002, he joined Mentor Graphics Corporation, Wilsonville, OR, where he is currently a Software Development Engineer for the design-for-test products. Prior to joining Mentor Graphics, he worked at the Institute of Electronics and Telecommunications, Poznań University of Technology. His current research interests include computer-aided design of digital circuits, design for testability, built-in self-test, and test compression.

Janusz Rajski (A'87) received the M.Eng. degree in electrical engineering from the Technical University of Gdańsk, Gdańsk, Poland, in 1973 and the Ph.D. degree in electrical engineering from the Poznań University of Technology, Poznań, Poland, in 1982.

From 1973 to 1984, he was a member of the faculty at the Poznań University of Technology. In June 1984, he joined McGill University, Montreal, Canada, where he became an Associate Professor in 1989. In January 1995, he became a Chief Scientist at Mentor Graphics Corporation, Wilsonville, OR. He has performed contract work and has been a consultant to a number of companies in the area of testing. He has published more than 100 research papers. He is the coauthor of *Arithmetic Built-In Self-Test for Embedded Systems* (Englewood Cliffs, NJ: Prentice Hall, 1997). His main research interests include design automation and testing of VLSI systems, design for testability, built-in self-test, and logic synthesis.

Dr. Rajski was a guest Co-editor of the special issue of the IEEE COMMUNICATIONS MAGAZINE devoted to testing telecommunication hardware, and an Associate Editor for IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the IEEE TRANSACTIONS ON COMPUTERS, and the IEEE DESIGN AND TEST OF COMPUTERS MAGAZINE. He is a Member of the editorial board of the *Journal of Electronic Testing* (JETTA). He was a co-recipient of the 1993 Best Paper Award for a paper on synthesis of testable circuits published in the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the 1995 and 1998 Best Paper Awards at the IEEE VLSI Test Symposium, and the 1999 Honorable Mention Award at the IEEE International Test Conference. He was Guest Co-editor of the June 1990 and January 1992 special issues of *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* devoted to the 1987 and 1989 International Test Conferences, respectively. He has served on technical program committees of various conferences and coordinated the topic of automatic test pattern generation and delay testing for the International Test Conference. He is a Co-founder of the International Test Synthesis Workshop.

Jerzy Tyszer (M'91–SM'96) received the M.Eng. (honors) and Ph.D. degrees in electrical engineering from the Poznań University of Technology, Poznań, Poland, in 1981 and 1987, respectively, and the Dr. Habilis degree in telecommunications from the Technical University of Gdańsk, Gdańsk, Poland, in 1994.

From 1982 to 1990 he was a member of the faculty at the Poznań University of Technology, Poland. In January 1990, he joined McGill University, Montreal, Canada, where he was a Research Associate and an Adjunct Professor. In 1996, he became a Professor at the Institute of Electronics and Telecommunications, Poznań University of Technology, where he carries on his work in the areas of testing and synthesis of self-testable circuits. He has done contract work and has been a consultant in the area of testing to a number of companies. He has published seven books, more than 70 research papers in the following areas and jointly holds 16 patents. He is a coauthor of *Arithmetic Built-In Self-Test for Embedded Systems* (Englewood Cliffs, NJ: Prentice Hall, 1997) and the author of *Object-Oriented Computer Simulation of Discrete Event Systems* (Norwell, MA: Kluwer, 1999). His main research interests include design automation and testing of VLSI systems, design for testability, built-in self-test, digital switching, and computer simulation of discrete event systems.

He was a co-recipient of the 1995 and 1998 Best Paper Awards at the IEEE VLSI Test Symposium. In 1999, he was a Guest Co-editor of the special issue of the IEEE COMMUNICATIONS MAGAZINE devoted to telecommunication hardware testing. He has served on the technical program committees of various conferences.