Technical Report

High-Speed Hybrid Ring Generator Design Providing Maximum-Length Sequences with Low Hardware Cost

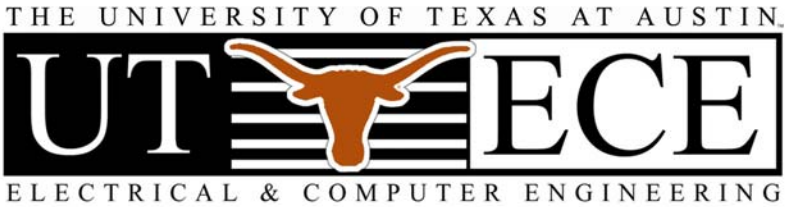
Laung-Terng Wang, Nur A. Touba, Richard P. Brent, Hui Wang, and Hui Xu

UT-CERC-12-01

October 4, 2011

Computer Engineering Research Center
The University of Texas at Austin

1 University Station, C8800
Austin, Texas 78712-0323
Telephone: 512-471-8000
Fax: 512-471-8967
http://www.cerc.utexas.edu



High-Speed Hybrid Ring Generator Design Providing Maximum-Length Sequences with Low Hardware Cost

Laung-Terng Wang[1], Nur A. Touba[2], Richard P. Brent[3], Hui Wang[4], and Hui Xu[4]

[1] SynTest Technologies, 505 S. Pastoria Ave., Suite 101, Sunnyvale, CA 94086, USA
[2] Department of Electrical and Computer Engineering, University of Texas, Austin, TX 78712, USA
[3] Mathematical Sciences Institute, Australian National University, Canberra, ACT 0200, Australia
[4] School of Microelectronics, Shanghai Jiao Tong University, Shanghai 200240, China

Abstract

A new class of hybrid ring generators is developed to generate maximum-length sequences with low hardware cost. The new design improves the operational speed of the hybrid linear feedback shift register (LFSR) proposed in [12] to receive the high speed and simplified layout benefits of the ring generator offered in [6]. As a result, the hybrid ring generator offers unmatched benefits over existing linear feedback shift register (LFSR) based designs. Assume k 2-input XOR gates are required in a standard or modular LFSR design. These benefits include requiring only (k+1)/2 XOR gates, having at most one level of a 2-input XOR gate between any pair of flip-flops, enabling the output of each flip-flop to drive at most 2 fanout nodes, and creating a highly regular structure that makes the new design more layout and timing friendly.

1. Introduction

With rapid advances in semiconductor process technologies and the explosive growth of the consumer electronics market, design of ***maximum-length sequence generators*** (MLSGs) to generate binary sequences for high-performance applications has reemerged as an important research topic. These applications range from computer engineering [1, 2] to communications [3] to cryptography [4].

The authors in [5] further commented that in communications and digital broadcasting, these high-speed MLSGs, such as ring generators [5-7], can randomize transmitted bitstreams, which prevent short repeating sequences from forming spectral lines that can complicate symbol tracking at the receiver or interfere with other transmissions. The *global positioning system* (GPS) can use these MLSGs to rapidly produce a sequence indicating high-precision relative time offsets. Cellular telephony and Bluetooth systems can use MLSGs as shrinking or alternating step generators in stream ciphers. These MLSGs can be deployed in a direct-sequence spread-spectrum radio or in various programmable sound generators. Finally, *high-definition television* (HDTV), digital audio broadcasting systems, gigabit Ethernet scramblers, and satellite communication systems might also adapt MLSGs due to their high performance and generic design flexibility.

Such MLSGs are often realized by ***maximum-length linear feedback shift registers*** (**LFSRs**). These maximum-length LFSRs are typically constructed in a standard or modular form, where one or more XOR gates are interspersed between a flip-flop and the feedback path to generate a desired maximum-length sequence (often called an ***m*-sequence**) [8]. If k 2-input XOR gates are required to generate an m-sequence, then the signal on the feedback path would have to propagate through k XOR gates (as in the **standard LFSR**) or must be strong enough to drive k+1 fanout nodes (as in the **modular LFSR**). In either case, the circuit is slowed and may not be applicable for high-performance applications.

To improve the performance of these conventional LFSRs, many approaches have been proposed. Most noticeable are the solutions that include **decimations** that allow summing up several m-sequences produced by independent devices with a multiphase clock generator [9]; **windmill machines** that elevate a state transition rate but need additional registers [10]; **hybrid LFSRs** that reduce the number of XOR gates to (k+1)/2 when the characteristic polynomial generating an m-sequence meets certain requirement [11, 12]; and **ring generators** that enable each flip-flop output to drive at most 2 fanout nodes and introduce at most one level of one 2-input XOR gate between any two flip-flops, if its characteristic polynomial does not contain consecutive terms [5-7].

These MLSGs, however, do not offer the combined benefits of using a smaller number of XOR gates and enabling any flip-flop to drive no more than 2 fanout nodes. This paper addresses this problem by constructing a new class of MLSGs (**hybrid ring generators**). When its characteristic polynomial meets certain requirement, the MLSG will use the same number of XOR gates as the hybrid LFSR [12] and preserve the high speed and simplified layout benefits of the ring generator [6]. The only benefit that the proposed hybrid ring generator does not preserve is that when a ring generator using k XOR gates couples to a phase shifter, the phase shifter used to drive multiple scan chains can have lower hardware cost than one coupled to a hybrid ring generator using only (k+1)/2 XOR gates, given a minimum **interchannel separation** criterion placed between any two m-sequences appearing at any two scan chain inputs [5, 13].

2. Background

There are two conventional forms of LFSR designs: standard LFSR and modular LFSR. Despite different state trajectories, both structures are capable of generating an *m*-sequence for each stage output.

2.1 Standard LFSRs

Fig. 1 shows an *n*-stage standard LFSR. It consists of *n* flip-flops and a number of XOR gates. Since XOR gates are placed on the external feedback path, the standard LFSR is also referred to as an **external-XOR LFSR** [8].

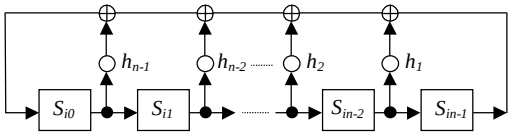


Figure 1. An *n*-stage (external-XOR) standard LFSR.

2.2 Modular LFSRs

Similarly, an *n*-stage modular LFSR with each XOR gate placed between two adjacent flip-flops, as shown in Fig. 2, is referred to as an **internal-XOR LFSR** [8]. This circuit runs faster than its corresponding standard LFSR, because each stage introduces at most one XOR-gate delay.

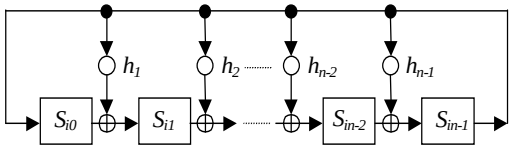


Figure 2. An *n*-stage (internal-XOR) modular LFSR.

2.3 LFSR Properties

The internal structure of the *n*-stage LFSR in each figure can be described by specifying a **characteristic polynomial** of degree *n*, $f(x)$, in which the symbol h_i is either 1 or 0, depending on the existence or absence of the feedback path, where

$$f(x) = 1 + h_1x + h_2x^2 + \ldots + h_{n-1}x^{n-1} + x^n. \quad (1)$$

Let S_i represent the contents of the *n*-stage LFSR after *i*th shifts of the initial contents, S_0, of the LFSR, and $S_i(x)$ be the polynomial representation of S_i, where $i \geq 0$. Then, $S_i(x)$ is a polynomial of degree $n-1$, where

$$S_i(x) = x^iS_0(x) \bmod f(x)$$
$$= S_{i0} + S_{i1}x + S_{i2}x^2 + \ldots + S_{in-2}x^{n-2} + S_{in-1}x^{n-1}. \quad (2)$$

If T is the smallest positive integer such that $f(x)$ divides $1+ x^T$, then the integer T is called the **period** of the LFSR. If $T = 2^n - 1$, then the *n*-stage LFSR generating the maximum-length sequence or *m*-sequence is called a **maximum-length LFSR** and thus can serve as an MLSG.

Define a **primitive polynomial** of degree *n* over **Galois field** GF(2), $p(x)$, as a polynomial that divides $1 + x^T$, but not $1 + x^i$, for any integer $i < T$, where $T = 2^n-1$ [8]. A primitive polynomial is **irreducible**. For illustration purpose, Figs. 3 and 4 show a 5-stage standard LFSR and a 5-stage modular LFSR with $f(x) = 1 + x^2 + x^3 + x^4 + x^5$, respectively. As can be seen, each circuit uses a total of 3 2-input XOR gates. The output signal at flip-flop 4 needs to propagate through 3 XOR gates to reach flip-flop 0 in Fig. 3 or must be strong enough to drive 4 fanout nodes in Fig. 4. The characteristic polynomial, $f(x)$, used to construct the circuits is a primitive polynomial, and thus each LFSR can serve as an MLSG. Let

$$r(x) = f(x)^{-1} = x^nf(1/x). \quad (3)$$

Then, $r(x)$ is defined as a **reciprocal polynomial** of $f(x)$ [8]. A reciprocal polynomial of a primitive polynomial is also a primitive polynomial. Hence, if the reciprocal polynomial of $f(x)$ is used to construct a standard or modular LFSR with $r(x) = 1 + x^2 + x^3 + x^4 + x^5$, then the LFSR can also serve as an MLSG.

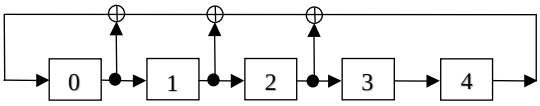


Figure 3. A 5-stage standard LFSR implementing $f(x) = 1+x^2+x^3+x^4+x^5$.

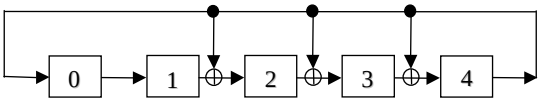


Figure 4. A 5-stage modular LFSR implementing $f(x) = 1+x^2+x^3+x^4+x^5$.

2.4 Hybrid LFSRs

Let a polynomial over GF(2), $1 + a(x) = b(x) + c(x)$, be said to be **fully decomposable** iff both $b(x)$ and $c(x)$ have no common terms and there exists an integer j such that $c(x) = x^jb(x)$, where $j \geq 1$. For example, if $1 + f(x)$ is fully decomposable such that

$$f(x) = 1 + b(x) + x^jb(x) \quad (4)$$

then a **(hybrid) top-bottom LFSR** [12] can be constructed using the feedback connection notation

$$s(x) = 1 + \wedge x^j + x^jb(x) \quad (5)$$

where $\wedge x^j$ indicates that the XOR gate with one input taken from the *j*th stage output of the LFSR is connected to the feedback path, not between stages. Similarly, if $f(x) + x^n$ is fully decomposable such that

$$f(x) = b(x) + x^jb(x) + x^n \quad (6)$$

then a **(hybrid) bottom-top LFSR** [12] can be constructed using the feedback connection notation

$$s(x) = b(x) + \wedge x^{n-j} + x^n. \quad (7)$$

Assume a maximum-length LFSR uses k 2-input XOR gates to generate an m-sequence. It was shown in [12] that *if 1 + f(x) or f(x) + x^n for constructing a standard or modular LFSR is fully decomposable, then a hybrid LFSR can be realized with only (k+1)/2 XOR gates. Also, if a top-bottom LFSR exists for f(x), then a bottom-top LFSR will exist for its reciprocal polynomial r(x), and vice versa.*

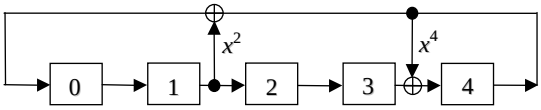


Figure 5. A 5-stage top-bottom LFSR using $s(x) = 1+\wedge x^2+x^4+x^5$ to implement $f(x) = 1+x^2+x^3+x^4+x^5$.

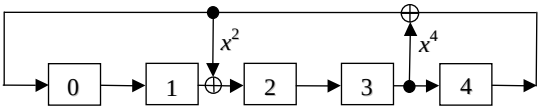


Figure 6. A 5-stage bottom-top LFSR using $s(x) = 1+x^2+\wedge x^4+x^5$ to implement $f(x) = 1+x+x^2+x^3+x^5$.

Fig. 5 shows an example 5-stage top-bottom LFSR. The circuit implements the same $f(x)$, $1 + x^2 + x^3 + x^4 + x^5$, as that for Figs. 3 and 4. Since $f(x) = 1 + (x^2+x^3) + x^2(x^2+x^3)$, by Eq. 5, $s(x) = 1 + \wedge x^2 + x^2(x^2+x^3) = 1 + \wedge x^2 + x^4 + x^5$. As $f(x)$ is a primitive polynomial, the top-bottom LFSR will generate an m-sequence.

Fig. 6 shows a bottom-top LFSR that implements the reciprocal polynomial, $1+x+x^2+x^3+x^5$, of the primitive polynomial for Fig. 5. Since $f(x) = (1+x^2) + x(1+x^2) + x^5$, by Eq. 7, $s(x) = (1+x^2) + \wedge x^{5-1} + x^5 = 1 + x^2 + \wedge x^4 + x^5$. As a reciprocal polynomial of a primitive polynomial is a primitive polynomial, the bottom-top LFSR will also generate an m-sequence.

As can be seen, each circuit illustrated in Figs. 5 and 6 uses only two 2-input XOR gates, rather than three XOR gates for Figs. 3 and 4. Assume k XOR gates are required to implement a standard LFSR or a modular LFSR to produce an m-sequence, where the integer k must be an odd number. The hybrid LFSR design will require only $(k+1)/2$ 2-input XOR gates. Since the feedback path of the hybrid LFSR will drive fewer fanout nodes than that of the standard or modular LFSR, the hybrid design will have better operating performance.

3. Hybrid Ring Generators

One common drawback of using the standard LFSR, modular LFSR, and hybrid LFSR to generate pseudorandom bit sequences is the long delay associated with the feedback path. In the standard LFSR case, data at the output of the rightmost flip-flop would need to pass through k 2-input XOR gates to reach the leftmost flip-flop. In the modular LFSR case, the rightmost flip-flop would need to be strong enough to drive $k+1$ (fanout) nodes. In the hybrid LFSR case, the rightmost flip-flop would need to pass through one 2-input XOR gate before or after driving $(k+1)/2$ fanout nodes. Combined with their respective irregularity in design style, these types of LFSR designs may have difficulty to meet frequency requirement for high-performance applications.

3.1 Top-Bottom Ring Generator Design

Consider the circuit given in Fig. 7. Any two adjacent flip-flops contain at most one 2-input XOR gate and each flip-flop output drives at most 2 fanout nodes. The circuit is constructed in a ring structure so there is no long feedback path connecting the rightmost flip-flop to the leftmost flip-flop. A circuit in so constructed is referred to as a ring generator [6]. Since the XOR gates are placed on the top and bottom rows simultaneously, a ring generator constructed with this additional property is referred to as a **hybrid ring generator**. Also, if the first XOR gate connecting to the leftmost stages is placed on the top row, then the hybrid ring generator is referred to as a **(hybrid) top-bottom ring generator** (see Fig. 7). Similarly, if the first XOR gate connecting to the rightmost stages is placed on the bottom row, then the hybrid ring generator is referred to as a **(hybrid) bottom-top ring generator** (see Fig. 9). Note that in each top-bottom or bottom-top ring generator, there will be *one and only one* 2-input XOR gate connected to the top row, according to the construction methods of the hybrid LFSRs given in [12].

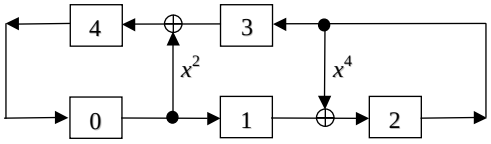


Figure 7. A 5-stage top-bottom ring generator constructed by $s(x) = 1+\wedge x^2+x^4+x^5$ given in Fig. 5.

Let $\boldsymbol{X} = \{x_0 \ldots x_4\}$ and $\boldsymbol{Z} = \{z_0 \ldots z_4\}$ represent the circuit's present state and next state, respectively. Linear equations over GF(2) governing the operation of the circuit can be expressed as follows:

$$\begin{aligned} z_0 &= x_4 \\ z_1 &= x_0 \\ z_2 &= x_1 + x_2 \\ z_3 &= x_2 \\ z_4 &= x_0 + x_3 \end{aligned} \tag{8}$$

The set of linear equations can be further described by:

$$\boldsymbol{Z = M * X} \tag{9}$$

or

$$\begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{10}$$

where matrix $\boldsymbol{M}$ is simply a **companion matrix** [8] whose characteristic polynomial $f(x)$ is defined as the **determinant** of $\boldsymbol{M - Ix}$, or symbolically:

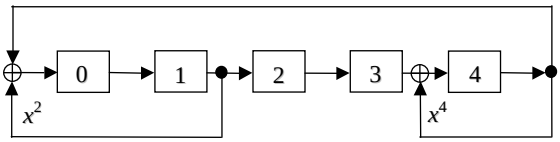
$$f(x) = | \boldsymbol{M} - \boldsymbol{I}x | \qquad (11)$$

Then, Eq. 11 can be rewritten as:

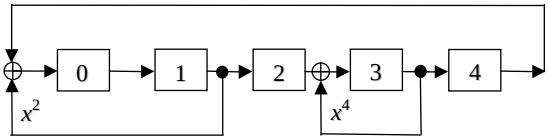
$$f(x) = \begin{vmatrix} x & 0 & 0 & 0 & 1 \\ 1 & x & 0 & 0 & 0 \\ 0 & 1 & 1+x & 0 & 0 \\ 0 & 0 & 1 & x & 0 \\ 1 & 0 & 0 & 1 & x \end{vmatrix} \qquad (12)$$

This yields $f(x) = x^4(1+x) + x^2(1+x) + 1 = 1 + x^2 + x^3 + x^4 + x^5$, which is a primitive polynomial used to construct the three circuits shown in Figs. 4, 5, and 7. This finding implies that *given f(x), if a top-bottom LFSR can be constructed, then a top-bottom ring generator can also be constructed with the same f(x).*

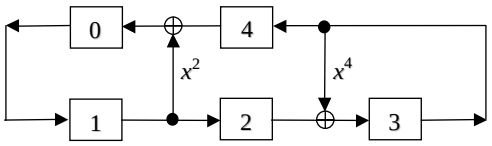
Consider the circuits shown in Figs. 8a to 8c. Fig. 8a is an equivalent circuit of Fig. 5; Fig. 8c is an equivalent circuit of Fig. 8b. Figs. 8a and 8b are represented in a one-dimensional view to reflect their feedback tap relationship. Fig. 8a is transformed to Fig. 8b, according to the transformations given in [6], by shifting the x^4 arc in Fig. 8a to the left by one bit without crossing the x^2 arc, while keeping the x^2 arc of Fig. 8a intact. One may now find Fig. 7 is isomorphic to Fig. 8c with only one difference in flip-flop labeling. This proof confirms our finding above.



(a) Equivalent circuit of Fig. 5



(b) Circuit by shifting the x^4 arc in (a) to the left by 1 bit



(c) Equivalent circuit of (b)

Figure 8. Equivalent circuits of Figs. 5 and 7.

3.2 Bottom-Top Ring Generator Design

Consider the 5-stage bottom-top ring generator shown in Fig. 9. The characteristic polynomial, $1 + x + x^2 + x^3 + x^5$, chosen to construct the hybrid circuit is the same reciprocal polynomial used to realize the bottom-top LFSR shown in Fig. 6.

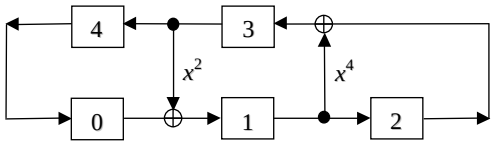


Figure 9. A 5-stage bottom-top ring generator constructed by $s(x) = 1+x^2+\wedge x^4+x^5$ given in Fig. 6.

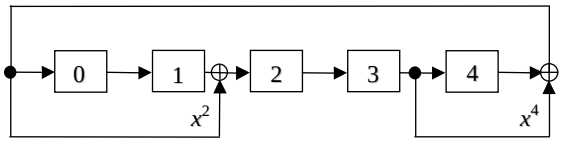
Looking into Fig. 9, the operation of the circuit relating next state $\boldsymbol{Z}$ to present state $\boldsymbol{X}$ can be expressed as:

$$\begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad (13)$$

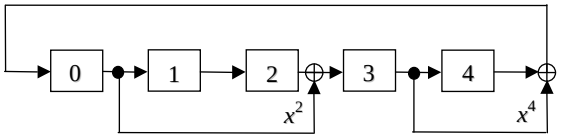
Then, by Eq. 11, $f(x)$ can be rewritten as:

$$f(x) = \begin{vmatrix} x & 0 & 0 & 0 & 1 \\ 1 & x & 0 & 1 & 0 \\ 0 & 1 & x & 0 & 0 \\ 0 & 1 & 1 & x & 0 \\ 0 & 0 & 0 & 1 & x \end{vmatrix} \qquad (14)$$

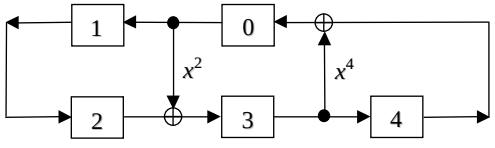
This yields $f(x) = (1+x^2) + x(1+x^2) + x^5 = 1 + x + x^2 + x^3 + x^5$, which is the primitive polynomial used to construct the bottom-top LFSR given in Fig. 6. According to Eq. 5, $s(x) = 1 + x^2 + \wedge x^4 + x^5$. The successive transformations of the circuit of Fig. 6 into that of Fig. 9 are shown in Figs. 10a to 10c. Fig. 10a is an equivalent circuit of Fig. 6. Fig. 10b was obtained by shifting the x^2 arc in Fig. 10a to the right by one bit. Fig. 10c is an equivalent circuit of Fig. 10b, and is isomorphic to Fig. 9 with different labeling in flip-flops. This proves that *given f(x), if a bottom-top LFSR can be constructed, then a bottom-top ring generator can also be constructed with the same f(x).*



(a) Equivalent circuit of Fig. 6



(b) Circuit by shifting the x^2 arc in (a) to the right by 1 bit



(c) Equivalent circuit of (b)

Figure 10. Equivalent circuits of Figs. 6 and 9.

3.3 Properties

Recall that the output of the rightmost flip-flop in a top-bottom LFSR must be strong enough to drive k+1 fanout nodes; whereas the output signal of the rightmost flip-flop in a bottom-top LFSR must propagate through k 2-inout XOR gates. A hybrid ring generator constructed either in a top-bottom or bottom-top form, however, will exhibit the same properties:

1. Every output of a flip-flop in the hybrid design will drive at most 2 fanout nodes.
2. There will be at most one 2-input XOR gate placed between any two flip-flops, and thus each output signal of any flip-flop will only have to propagate through at most one 2-input XOR gate.
3. There will be no long feedback path, as the circuit is implemented in a ring structure.
4. Its regular and modular structure will result in simplified layout and routing, making the circuit timing and layout friendly.
5. The number of 2-input XOR gates used in the hybrid ring generator will be (k+1)/2.

The hybrid ring generator is able to preserve the first 4 benefits given in [5, 6]. This has enabled the circuit to run at a higher speed than its standard, modular, and hybrid LFSR counterparts. As the goal of the paper is to design a modified (maximum-length) LFSR that has the least hardware cost, it is beyond the scope of the paper to discuss techniques that will meet a minimum *interchannel separation* criterion, say 4,096 or 10,000 bits, between any two scan chains [5, 7, 13]. Instead, we will prove that *any modified LFSR (such as a hybrid LFSR, ring generator, or hybrid ring generator) implementing the same f(x) as a standard or modular LFSR using k 2-input XOR gates cannot use fewer than (k+1)/2 XOR gates, when k = 1, 3, or 5*.

Before the proof, consider the two circuits given in Figs. 11 and 12 first. Both circuits were taken from FIGS. 9 and 14 in [14], respectively. Fig. 11 is to illustrate a particular situation where it is required to add an *extra* 2-input XOR gate in a modular LFSR when *a source tap crossing a destination tap while moving to the left* (**SDL**) transformation is used to construct a modified LFSR. Fig. 12 is to illustrate another situation where the inserted extra gate can cancel an available XOR gate, thereby reducing the number of XOR gates in the circuit by one.

In Fig. 11a, two feedback connections 58 and 59 are arranged in such a way that an XOR gate 60 at the destination tap of the first feedback connection is separated from a source tap 62 of the second feedback connection by a single flip-flop. An *elementary shift left* (**EL**) transformation described in [6, 14] is applied to the circuit so the source tap 62 shifts across this flip-flop (see Fig. 11b). The XOR gate 64 at the destination tap of the second feedback connection also shifts to the left accordingly. This operation preserves the m-sequence property of the LFSR as described in [6, 14]. Next, the source tap 62 moves to cross the XOR gate 60 of the first feedback connection 58 (see Fig. 11c). Logic value on the second feedback connection 59 is now no longer equivalent to a mod b; instead, it is now equal to just b. To maintain the same functionality on the output of the destination XOR gate 64, logic value a must be provided by the source tap 66 of the first feedback connection 58 to the XOR gate 64. This is accomplished by adding a feedback connection line 68 between the source tap 66 and the XOR gate 64 at the shifted destination tap. One can see now an extra XOR gate is added to the modified LFSR to preserve the same m-sequence property.

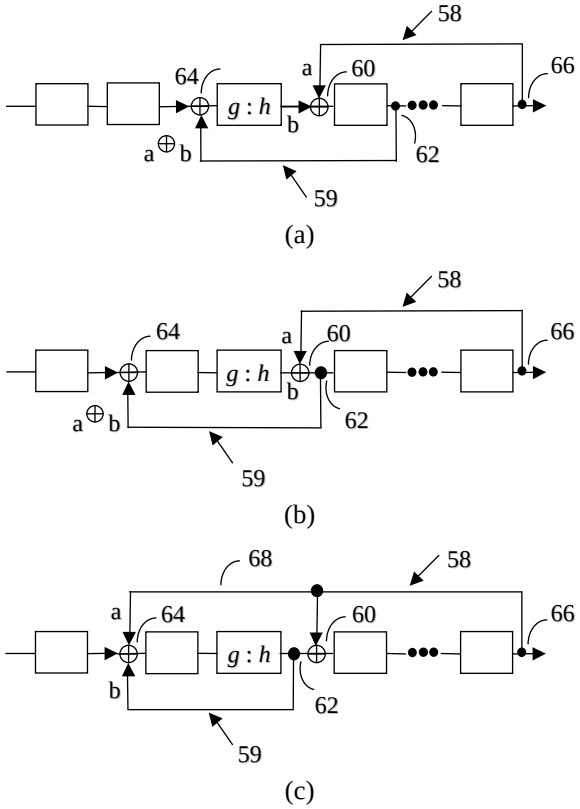


Figure 11. A circuit to illustrate an SDL transformation can lead to insertion of an extra XOR gate.

Fig. 12a shows a modular LFSR implementing $f(x) = 1 + x^2 + x^3 + x^7 + x^8$. First, transformation EL is applied 4 times to the feedback connection represented by coefficient x^7 (feedback connection 30 with source tap 32 and destination gate 34). This leads to the circuit shown in Fig. 12b. Next, transformation SDL is applied to shift the feedback connection 30 further to the left by one flip-flop and adds a feedback connection line 36 at the input to the XOR gate 34 as shown in Fig. 12c. Because another XOR gate 38 with the same connectivity already exists at the output of flip-flop 1, the XOR gate 34 and connection 36

can be discarded. This reduces the number of XOR gates in the LFSR from 3 to 2. To reduce the load of flip-flop 2 that drives XOR gates 40 and 34 in Fig. 12c, an additional transformation EL is applied in Fig. 12d that shifts the feedback connection 30 further to the left. As a result, the modified LFSR uses only 2 XOR gates and every flip-flop output drives at most two fanout nodes.

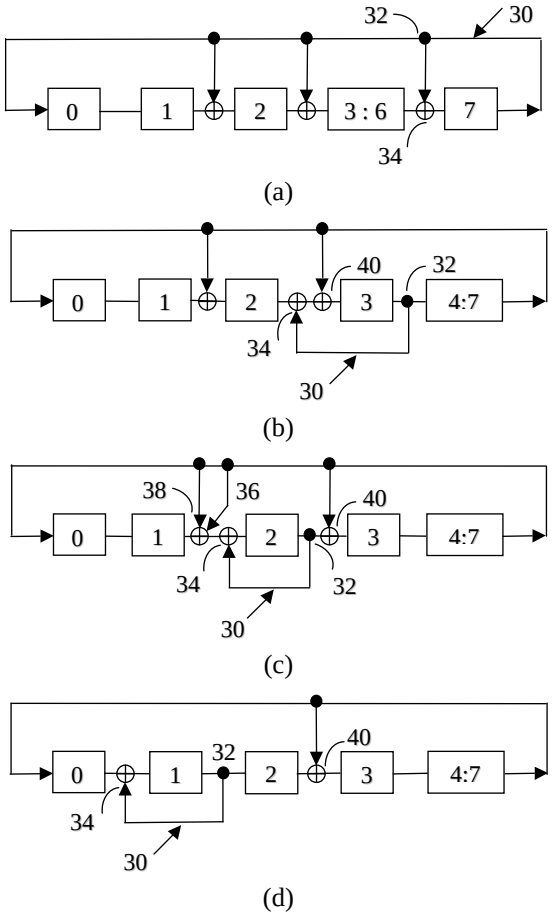


Figure 12. An 8-stage modified LFSR constructed using the transformations given in [14] for $f(x) = 1+x^2+x^3+x^7+x^8$.

The above two examples (Figs. 11 and 12) illustrate that applying transformations to a modular LFSR can lead to insertion or deletion of one or more XOR gates. The number of 2-input XOR gates used in the resultant modified LFSR, however, will be at least $(k+1)/2$, *when k = 1, 3, or 5*. The same results apply to transformations of a standard LFSR too. We now provide the proof below:

Theorem 1*: given a maximum-length standard or modular LFSR using k 2-input XOR gates, a modified LFSR implementing the same f(x) as the standard or modular LFSR cannot use fewer than (k+1)/2 2-input XOR gates, when k = 1, 3, or 5.*

Proof: We will prove the theorem by contradiction. When $k = 1$, the condition follows immediately; otherwise, the modified LFSR would not contain any XOR gates and would have implemented $1 + x^n$, which is different from the **primitive trinomial** (a primitive polynomial with 3 terms) used as $f(x)$ to construct the maximum-length standard or modular LFSR.

Next, we show that if $k = 3$, then the condition will still hold. A maximum-length standard or modular LFSR constructed to implement $f(x)$ with $k = 3$ implies that the LFSR uses 3 2-input XOR gates and $f(x)$ is a **primitive pentanomial** (a primitive polynomial with 5 terms). For instance, a modular LFSR is constructed to implement $f_1(x) = p(x) = 1 + x^a + x^b + x^c + x^n$, where $1 \le a < b < c < n$. According to [14], when a source tap of one arc in $\{x^a, x^b, x^c\}$ and a destination tap of another arc in $\{x^a, x^b, x^c\}$ cross each other, it will be required to add a proper feedback connection (a 2-input XOR gate) in the modified LFSR to preserve the *m*-sequence property in the standard or modular LFSR. If the extra gate is to be cancelled, then there must exist an available XOR gate at the position where the extra gate will be added. For instance, the x^b and x^c arcs have a distance of n-b and n-c to the rightmost stage of the modular LFSR, respectively; the x^a arc must be in the same position as the to-be-added feedback connection. That is, distance n-a must be equal to (n-b) + (n-c), or $a + n = b + c$. When this condition holds, the x^a arc will be cancelled. This also implies that $1 + f_1(x)$ is *fully decomposable*. The modified LFSR will now have only 2 XOR gates (representing the original x^b arc and the transformed x^c arc) left. If the number of XOR gates used in this modified LFSR could be reduced to 1 (instead of 2), this means there must exist transformation(s) that can cause the transformed x^c arc to cancel the original x^b arc, or vice versa. If this were possible, then the modified LFSR would have implemented $f_2(x) = 1 + x^c + x^n$ or $1 + x^b + x^n$, which becomes a primitive trinomial. This will contradict the condition that the modified LFSR must implement the same characteristic polynomial $f_1(x)$ as the maximum-length standard or modular LFSR.

We now prove a modified LFSR that implements the same $f_3(x)$ as a maximum-length standard or modular LFSR using 5 2-input XOR gates will use no fewer than 3 2-input XOR gates. As shown in Fig. 12, to reduce the number of XOR gates used in a modified LFSR by one, a feedback connection at the same flip-flop output of the source or destination tap must already exist in the original LFSR to cancel the added XOR gate; otherwise, the XOR gate count would be increased. Let the modular LFSR implement $f_3(x) = p(x) = 1 + x^a + x^b + x^c + x^d + x^e + x^n$, where $1 \le a < b < c < d < e < n$, with $k = 5$ feedback taps $\{x^a, x^b, x^c, x^d, x^e\}$. For instance, $f_3(x) = p(x) = 1 + x^5 + x^{10} + x^{14} + x^{19} + x^{24} + x^{29}$. Only when $c + n = d + e$ and $a + n = b + e$, can the combined x^e and x^d arcs as well as the combined x^e and x^b arcs cancel the x^c and x^a taps, respectively. This also implies that $1 + f_3(x)$ is *fully*

decomposable. The modified LFSR now has 3 arcs {x^b, x^d, x^e} left. The only chance to cancel one more feedback connection (the x^b tap) would be when the condition $b + n = d + e$ holds. This condition cannot hold because $c + n = d + e$. One scenario that needs to consider is whether creating an intermediate XOR gate could lead to other reductions in later steps when $k = 5$. If there were such transformations that could further reduce the circuit to one that contains only 2 arcs, then the 2 arcs in the transformed circuit would take on one of the two following structures: 1) in a *disjoint* form where both destination taps point to the same direction (left or right), similar to Fig. 8b or 10b; or 2) in a *closed* form where one arc is included in another arc and both destination taps point to the same direction (left or right). A disjoint circuit structure with both source or destination taps pointed to each other is isomorphic to Structure 2) when one arc rotates across the feedback path. Similarly, a closed circuit structure with both source or destination taps pointed to different directions is isomorphic to Structure 1) when one arc rotates across the feedback path. By *retransforming* the circuit back to a standard or modular LFSR, Structure 1 will yield an LFSR that uses 3 2-input XOR gates or $k = 3$; whereas Structure 2 will yield an LFSR that uses only 2 2-input XOR gates or $k = 2$. Structure 2 cannot exit because k must be odd for realizing a maximum-length LFSR. Structure 1 cannot exist either, because the retransformed circuit would have implemented a primitive pentanomial instead. Both circuit structures also contradict the condition that the modified LFSR must implement the same characteristic polynomial $f_3(x)$ as the standard or modular LFSR with $k = 5$. Hence, any modified LFSR that implements the same $f(x)$ as the maximum-length standard or modular LFSR with k 2-input XOR gates will use at least $(k+1)/2$ 2-input XOR gates, when $k = 1$, 3, or 5. This concludes the proof. □

Note that while Theorem 1 is mainly provided for construction of hybrid ring generators that use primitive polynomials as characteristic polynomials to yield the lowest hardware cost and guarantee the m-sequence property, the theorem can also be applied to construction of any modified LFSR from a standard or modular LFSR whose characteristic polynomial does not necessarily implement a primitive polynomial, when $1 \le k \le 5$.

4. Construction Method

To better understand how a hybrid ring generator can be designed via visual inspection or by a construction method, consider the 8-stage top-bottom ring generator illustrated in Fig. 13 for implementing $f(x) = p(x) = 1 + x^2 + x^3 + x^7 + x^8$. This primitive polynomial, $p(x)$, is the reciprocal polynomial, $r(x)$, of the primitive polynomial $1 + x + x^5 + x^6 + x^8$ listed in [1]. Also, the same $f(x)$ has been used to construct the modified LFSR in Fig. 12. Because $f(x) = 1 + (x^2+x^3) + x^5(x^2+x^3)$, this means $s(x) = 1 + \wedge x^5 + x^7 + x^8$. A corresponding 8-stage bottom-top ring generator implementing $r(x)$ is shown in Fig. 14. Since $r(x) = (1+x) + x^5(1+x) + x^8$, this yields $s(x) = 1+ x + \wedge x^3 + x^8$.

By **visual inspection** of the hybrid ring generators shown in Figs. 7, 9, 13, and 14, one may find the feedback connections in each circuit are exactly arranged in the same way as that described in [5]: *given tap x^i, create a feedback connection by encompassing i adjacent flip-flops, always beginning with the leftmost ones*. The difference is only the numbers labeled in the flip-flops. We decide to label the flip-flop numbers from 0 to n–1 *counterclockwise* starting with the leftmost bottom flip-flop in the hybrid ring generator design because its circuit structure will be more in line with the standard and modular LFSR designs.

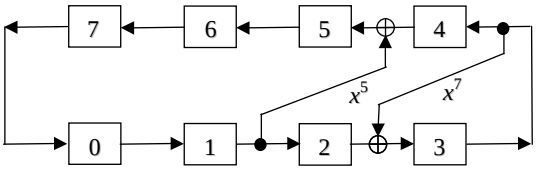


Figure 13. An 8-stage top-bottom ring generator using $s(x) = 1+\wedge x^5+x^7+x^8$ to implement $f(x) = 1+x^2+x^3+x^7+x^8$.

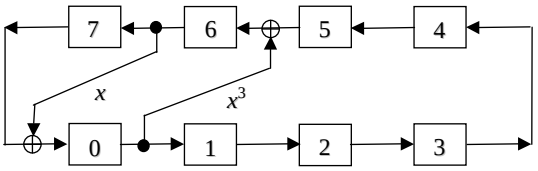


Figure 14. An 8-stage bottom-top ring generator using $s(x) = 1+x+\wedge x^3+x^8$ to implement $r(x) = 1+x+x^5+x^6+x^8$.

A **construction method** following the definitions in [6] for designing a top-bottom or bottom-top ring generator from a hybrid LFSR is now given below [15]:

Step 1: Let T_i represent the span (coefficient c) of the ith tap (x^c); S_i and D_i indicate the locations of the source and destination taps (as inputs to a 2-input XOR gate) in the resultant hybrid ring generator, respectively; and L be the number of flip-flops in a hybrid LFSR. If L is an odd number, let $L = L + 1$; next, label 0 to $L - 1$ on each flip-flop *counterclockwise*, starting with an entry 0 on the leftmost bottom flip-flop; then, calculate locations of the source and destination taps according to the following formulas:

$$S_i = (L - T_i) / 2 + L / 2 - 1 \quad (15)$$

$$D_i = (S_i + T_i) \bmod L. \quad (16)$$

Consider Fig. 13 again. $L = 8$. The two x^5 and x^7 taps in $s(x) = 1 + \wedge x^5 + x^7 + x^8$ is represented by a sequence $T_1 = 5$, $T_2 = 7$ (entries 0 and 8 do not have to be processed as they are not subject to transformations). Thus, applying Eqs. 15 and 16 will yield the following feedback connections: $S_1 = (8–5)/2 + 8/2 – 1 =$ **4**, $D_1 = (4+5) \bmod 8$

= **1**; S_2 = (8–7)/2 + 8/2 – 1 = **3**, D_2 = (3+7) mod 8 = **2**. The two taps can be expressed as a list of pairs: (4,1), (3,2).

Step 2: Reverse the direction of the leftmost (or rightmost) tap to create the $\wedge x^c$ tap on the top row for a top-bottom (or bottom-top) ring generator.

Step 3: (Required only when the circuit has an odd number of stages) Delete the entry $L/2$ from the label and decrement all entries on the top row by 1.

For example, Fig. 7 has 5 flip-flops. The circuit will be first labeled with {0, 1, 2, **3**, **4**, **5**} for L = 6 (not 5). Then, delete the entry 3 and renumber the rest to {0, 1, 2, **3, 4**}.

A set of minimum-weight primitive polynomials (each consisting of 3 or 5 terms [*a.k.a*. weights, exponents, or coefficients]) of degree up to 100 that can be used to construct hybrid LFSRs has been listed in [2, 12]. Stahnke was the first to report a list of minimum-weight primitive polynomials of degree up to 168 that satisfies the *full decomposable* requirement [16]. A new list of minimum-weight primitive polynomials of degree up to 800 is now given in the Appendix generated using modified NTL [17] and Magma [18] programs with prime factors provided in [19]. For every primitive polynomial of degree up to 800, we found a primitive pentanomial that meets the *fully decomposable* requirement always exists when a primitive trinomial does not exist.

Quite a few tables have been reported earlier for different objectives, including minimum-weight primitive polynomials of degree up to 300 in [20]; minimum-weight primitive polynomials of degree 310 through 500 in [21]; and primitive polynomials of degree 9 through 660 with uniformly distributed coefficients in [22].

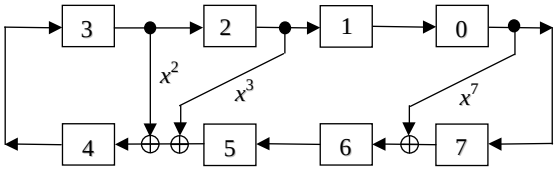
Based on the construction method, each polynomial listed in the Appendix can now be used to construct hybrid ring generators. It is interesting to note that for any n-stage hybrid ring generator, $n \leq 800$, only one or two 2-input XOR gates are required to generate an m-sequence.

5. Comparative Analysis

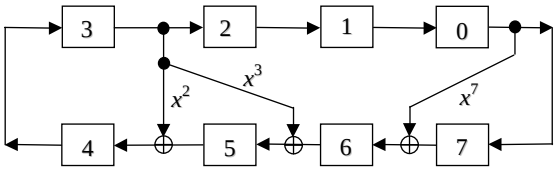
Here, we first make two observations on how the design of ring generators is related to hybrid ring generator design. The benefits of the proposed hybrid ring generator design over other types of MLSGs are then discussed.

Fig. 15a shows an original ring generator design using the synthesis method given in [6] to implement $f(x) = p(x) = 1 + x^2 + x^3 + x^7 + x^8$. The same $f(x)$ has been used to construct the hybrid ring generator shown in Fig. 13.

Comparing the structures of both Figs. 13 and 15a, one can find that Fig. 15a has 2 levels of 2-input XOR gates placed between flip-flops 4 and 5, and uses one more XOR gate than Fig. 13. Conversely, one may construct a ring generator as shown in Fig. 15b so the output of flip-flop 3 drives 3 fanout nodes, instead of 2 nodes [5].



(a) An 8-stage ring generator based on [6]



(b) Another 8-stage ring generator based on [5]

Figure 15. An 8-stage ring generator constructed using the synthesis method given in [5, 6] for $f(x) = 1+x^2+x^3+x^7+x^8$.

This problem was mainly caused by the chosen primitive polynomial that contains consecutive terms (*i.e*., x^2 and x^3; 1 and x as well as x^{n-1} and x^n do not count). If the chosen primitive polynomial does not contain consecutive terms, then the ring generator will always have only one-level of a 2-input XOR gate placed between any pair of flip-flops and enable any flip-flop output to drive at most 2 fanout nodes.

Fortunately, we were able to find a primitive polynomial of degree 8 that does not contain consecutive terms, $1 + x + x^3 + x^5 + x^8$ [17]. This leads to our first observation: *when designing a ring generator, it is important to choose a primitive polynomial, p(x) as characteristic polynomial, f(x), which does not contain consecutive terms; however, choosing such a primitive polynomial may not be an issue for designing a hybrid ring generator, as long as these consecutive terms can be factored out.*

Our second observation is associated with the ring generator design: *the ring generator does not implement the chosen characteristic polynomial, f(x), but the reciprocal polynomial, r(x), of the chosen f(x).* For instance, in Figs. 15a-b, while an m-sequence is always generated, neither circuit implements $f(x) = 1 + x^2 + x^3 + x^7 + x^8$, but the reciprocal polynomial of $f(x)$, or $r(x) = 1 + x + x^5 + x^6 + x^8$. One can verify the resultant polynomial by building a companion matrix using the approach we discussed in Section 3.

This problem was caused by an incorrect design for placing a wrong order of feedback taps on the modular LFSR which was referred to as a **Galois LFSR** in [5]. To correct this error, one can simply renumber the flip-flops and construct the feedback taps by Eqs. 15 and 16. The correct **modified ring generator** is shown in Fig. 16, where the direction of the feedback path is reversed from Fig. 15a and the flip-flops are labeled differently.

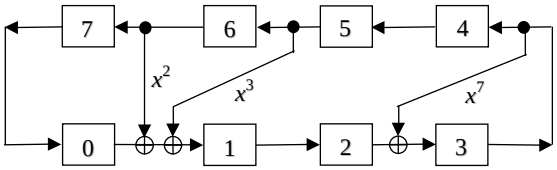


Figure 16. A correct 8-stage modified ring generator implementing $f(x) = 1+x^2+x^3+x^7+x^8$.

Table 1 now summarizes the design features of various MLSGs. The table provides a more accurate measure than Table 1 given in [6] on the top-bottom and bottom-top LFSR design features. The top-bottom (or bottom-top) LFSR will have one level (or two levels) of XOR logic because it is constructed to have *only* one 2-input XOR gate connected to the feedback path according to Eq. 5 (or Eq. 7). On the other hand, the feedback path in each top-bottom or bottom-top LFSR will always drive (k+1)/2 fanout nodes due to the nature of the design. As to *cellular automaton* (CA), in general, the total number of 2-input XOR gates used in a CA design will be equal to 2n-2 for providing better randomness [23, 24].

Table 1. Features of LFSR-Based MLSG Designs

	XOR Gates	Levels of Logic	Fanout
Standard LFSR	k	$\log_2 k$	2
Modular LFSR	k	1	k + 1
Top-Bottom LFSR	(k + 1) / 2	1	(k + 1) / 2
Bottom-Top LFSR	(k + 1) / 2	2	(k + 1) / 2
Cellular Automaton	2n – 2	2	3
Ring Generator	k	1	2
Hybrid Ring Generator	(k + 1) / 2	1	2

Note that the Level of Logic and Fanout columns given in the ring generator row assume that the chosen primitive polynomial as $f(x)$ to design the ring generator does not contain consecutive terms. If one chooses a primitive polynomial that contains consecutive terms, then the Level of Logic or Fanout would have to be increased by one. Similar assumption also applies to hybrid ring generator design: the chosen primitive polynomial must be the one such that its corresponding feedback connection notation, $s(x)$, does not contain consecutive terms. Fortunately, such primitive polynomials for the degree (not every degree) up to 660 listed in [25] and every degree up to 800 listed in the Appendix always exist.

The researchers in [14] have shown an example (as depicted in Fig. 12) using a series of transformations to reduce the number of XOR gates to 2 for Fig. 15a. Interestingly, the **transformed LFSR** (t-LFSR) converges to a hybrid ring generator. However, one major difference between a transformed LFSR and a hybrid ring generator is that the proposed hybrid design approach does not need to go through any transformations once a proper primitive polynomial is found. As we have proved in Theorem 1 that *given a maximum-length standard or modular LFSR using k 2-input XOR gates, a modified LFSR implementing the same f(x) as the standard or modular LFSR cannot use fewer than (k+1)/2 XOR gates, when k = 1, 3, or 5*, the proposed hybrid ring generator will be able to match or outperform all other LFSR-based designs having the lowest hardware cost.

6. Conclusion

This paper described a high-speed design of hybrid ring generators that has yielded the lowest hardware cost among all LFSR-based designs practiced today. It provides quick visual inspection rule of thumb and a simple construction method to design the circuit without going through any transformations. We found that for each n-stage hybrid ring generator, $n \leq 800$, only one or two 2-input XOR gates are required to generate an m-sequence. This enables the circuit to be deployed to generate pseudorandom bit sequences for high-performance applications.

In future work, we plan to extend Theorem 1 to find the true minimum number of 2-input XOR gates required to construct a modified LFSR out of a standard or modular LFSR using k 2-input XOR gates. The characteristic polynomial does not have to be primitive. We also plan to explore the implications of the proposed hybrid ring generators on the design of dense ring generators [7], phase shifters [13], and event counters [26, 27], and seek minimum-weight primitive polynomials of degree 801 through 1200 using the prime factors provided in [19].

7. Acknowledgments

The authors sincerely express our gratitude to Professor Samuel S. Wagstaff, Jr. in the Departments of Computer Sciences and Mathematics at Purdue University for providing the needed prime factors so we can use NTL for computations to generate desired primitive polynomials and check the results with those generated by Magma, or vice versa. We also would like to thank Alice Yu of the University of California at San Diego and Teresa Chang of SynTest Technologies for drawing all figures. This research was supported in part by the National Science Foundation under Grant No. CCF-0916837.

References

[1] M.L. Bushnell and V.D. Agrawal, *Essentials of Electronic Testing for Digital, Memory & Mixed-Signal VLSI Circuits*, Springer, New York, 2000.

[2] L.-T. Wang, C.-W. Wu, and X. Wen, editors, *VLSI Test Principles and Architectures: Design for Testability*, Morgan Kaufmann, San Francisco, 2006.

[3] W.W. Peterson and E.J. Weldon, Jr., *Error-Correcting Codes*, MIT Press, Cambridge, Massachusetts, 1972.

[4] W. Trappe and L.C. Washington, *Introduction to Cryptography with Coding Theory*, Second Edition, Prentice Hall, Upper Saddle River, New Jersey, 2005.

[5] N. Mukherjee, J. Rajski, G. Mrugalski, A. Pogiel, and J. Tyszer, "Ring Generator: An Ultimate Linear Feedback Shift Register," *IEEE Computer*, pp. 64-71, June 2011.

[6] G. Mrugalski, J. Rajski, and J. Tyszer, "High Speed Ring Generators and Compactors of Test Data," *IEEE VLSI Test Symp.*, pp. 57-62, 2003.

[7] G. Mrugalski, N. Mukherjee, J. Rajski, and J. Tyszer, "High-Performance Dense Ring Generators," *IEEE Trans. on Computers*, vol. 55, no. 1, pp. 83-87, Jan. 2006.

[8] S.W. Golomb, *Shift Register Sequence*, Aegean Park Press, Laguna Hills, California, 1982.

[9] C. Arvillias and D.G. Maritsas, "Toggle-Registers Generating in Parallel k kth Decimations of m-sequences $X^p + X^k + 1$ Design Tables," *IEEE Trans. on Computers*, vol. C-28, no. 2, pp. 89-101, Feb. 1979.

[10] W.W. Warlick and J.E. Hershey, "High-Speed m-Sequence Generators," *IEEE Trans. on Computers*, vol. C-29, no. 5, pp. 398-400, May 1980.

[11] L.-T. Wang and E.J. McCluskey, "A Hybrid Design of Maximum-Length Sequence Generators," *Proc. IEEE Int. Test Conf.*, pp. 38-47, 1986.

[12] L.-T. Wang and E.J. McCluskey, "Hybrid Designs Generating Maximum-Length Sequences," *IEEE Trans. on Computer-Aided Design*, vol. 7, no. 1, pp. 91-99, Jan. 1988.

[13] J. Rajski and J. Tyszer, "Automated Synthesis of Phase Shifters for Built-In Self-Test Applications," *IEEE Trans. on Computer-Aided Design*, vol. 19, no. 10, pp. 1175-1188, Oct. 2000.

[14] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee, "Method for Synthesizing Linear Finite State Machines," United States Patent No. 6,353,842, March 5, 2002.

[15] L.-T. Wang and N.A. Touba, "Method and Apparatus for Hybrid Ring Generator Design," United States Patent Application No. 13/195,524, August 1, 2011.

[16] W. Stahnke, "Primitive Binary Polynomials," Mathematics of Computation, vol. 27, no. 124, pp. 977-980, Oct. 1973.

[17] NTL: http://www.shoup.net/ntl/.

[18] Magma: http://www.math.ufl.edu/help/magma/MAGMA.html.

[19] J. Brillhart, D.H. Lehmer, J.L. Selfridge, B. Tuckerman, and S. S. Wagstaff, Jr., *Contemporary Mathematics - Factorizations of bn±1, b = 2, 3, 5, 6, 7,10, 11, 12 up to High Powers*, Third Edition, American Mathematical Society, vol. 22, 2002; also available in http://www.ams.org/publications/online-books/conm22-index.

[20] P.H. Bardell, W.H. McAnney, and J. Savir, *Built-In Test for VLSI: Pseudorandom Techniques*, Somerset, New Jersey: John Wiley & Sons, 1987.

[21] P.H. Bardell, "Primitive Polynomials of Degree 301 through 500," *J. Electronic Testing: Theory and Applications*, vol. 3, no. 2, pp. 175-176, May 1992.

[22] J. Rajski and J. Tyszer, "Primitive Polynomials over GF(2) of Degree up to 660 with Uniformly Distributed Coefficients," *J. Electronic Testing: Theory and Applications*, vol. 19, no. 6, pp. 645-657, Dec. 2003.

[23] P.D. Hortensius, R.D. McLeod, W. Pries, D.M. Miller, and H.C. Card, "Cellular Automata-Based Pseudorandom Number Generators for Built-In Self-Test," *IEEE Trans. on Computer-Aided Design*, vol. 8, no. 8, pp. 842-859, Aug. 1989.

[24] G. Mrugalski, J. Rajski, and J. Tyszer, "Cellular Automata-Based Test Pattern Generators with Phase Shifters," *IEEE Trans. on Computer-Aided Design*, vol. 19, no. 8, pp. 878-893, Aug. 2000.

[25] G. Mrugalski, N. Mukherjee, J. Rajski, and J. Tyszer, "Planar High Performance Ring Generators," *IEEE VLSI Test Symp.*, pp. 193-198, 2004.

[26] N. Mukherjee, A. Pogiel, J. Rajski, and J. Tyszer, "High-Speed On-Chip Event Counters for Embedded Systems," *Proc. IEEE Int. Conf. on VLSI Design*, pp. 275-280, 2009.

[27] D.W. Clark and L.-J. Weng, "Maximal and Near-Maximal Shift Register Sequences: Efficient Event Counters and Easy Discrete Logarithms," *IEEE Trans. on Computers*, vol. 43, no. 5, pp. 560-568, May 1994.

Appendix: Minimum-Weight Primitive Polynomials of Degree up to 800 over GF(2)

	2 1 0	3 1 0	4 1 0	5 2 0
6 1 0	7 1 0	8 6 5 1 0	9 4 0	10 3 0
11 2 0	12 7 4 3 0	13 4 3 1 0	14 12 11 1 0	15 1 0
16 5 3 2 0	17 3 0	18 7 0	19 6 5 1 0	20 3 0
21 2 0	22 1 0	23 5 0	24 4 3 1 0	25 3 0
26 8 7 1 0	27 8 7 1 0	28 3 0	29 2 0	30 16 15 1 0
31 3 0	32 28 27 1 0	33 13 0	34 15 14 1 0	35 2 0
36 11 0	37 12 10 2 0	38 6 5 1 0	39 4 0	40 21 19 2 0
41 3 0	42 23 22 1 0	43 6 5 1 0	44 27 26 1 0	45 4 3 1 0
46 21 20 1 0	47 5 0	48 28 27 1 0	49 9 0	50 27 26 1 0
51 16 15 1 0	52 3 0	53 16 15 1 0	54 37 36 1 0	55 24 0
56 22 21 1 0	57 7 0	58 19 0	59 22 21 1 0	60 1 0
61 16 15 1 0	62 57 56 1 0	63 1 0	64 4 3 1 0	65 18 0
66 10 9 1 0	67 10 9 1 0	68 9 0	69 29 27 2 0	70 16 15 1 0
71 6 0	72 53 47 6 0	73 25 0	74 16 15 1 0	75 11 10 1 0
76 36 35 1 0	77 31 30 1 0	78 20 19 1 0	79 9 0	80 38 37 1 0
81 4 0	82 38 35 3 0	83 46 45 1 0	84 13 0	85 28 27 1 0
86 13 12 1 0	87 13 0	88 72 71 1 0	89 38 0	90 19 18 1 0
91 84 83 1 0	92 13 12 1 0	93 2 0	94 21 0	95 11 0
96 49 47 2 0	97 6 0	98 11 0	99 47 45 2 0	100 37 0
101 7 6 1 0	102 77 76 1 0	103 9 0	104 11 10 1 0	105 16 0
106 15 0	107 65 63 2 0	108 31 0	109 7 6 1 0	110 13 12 1 0
111 10 0	112 45 43 2 0	113 9 0	114 82 81 1 0	115 15 14 1 0
116 71 70 1 0	117 20 18 2 0	118 33 0	119 8 0	120 118 111 7 0
121 18 0	122 60 59 1 0	123 2 0	124 37 0	125 108 107 1 0
126 37 36 1 0	127 1 0	128 29 27 2 0	129 5 0	130 3 0
131 48 47 1 0	132 29 0	133 52 51 1 0	134 57 0	135 11 0
136 126 125 1 0	137 21 0	138 8 7 1 0	139 8 5 3 0	140 29 0
141 32 31 1 0	142 21 0	143 21 20 1 0	144 70 69 1 0	145 52 0
146 60 59 1 0	147 38 37 1 0	148 27 0	149 110 109 1 0	150 53 0
151 3 0	152 66 65 1 0	153 1 0	154 129 127 2 0	155 32 31 1 0
156 116 115 1 0	157 27 26 1 0	158 27 26 1 0	159 31 0	160 19 18 1 0
161 18 0	162 88 87 1 0	163 60 59 1 0	164 14 13 1 0	165 31 30 1 0
166 39 38 1 0	167 6 0	168 17 15 2 0	169 34 0	170 23 0
171 19 18 1 0	172 7 0	173 100 99 1 0	174 13 0	175 6 0
176 119 118 1 0	177 8 0	178 87 0	179 34 33 1 0	180 37 36 1 0
181 7 6 1 0	182 128 127 1 0	183 56 0	184 102 101 1 0	185 24 0
186 23 22 1 0	187 58 57 1 0	188 74 73 1 0	189 127 126 1 0	190 18 17 1 0
191 9 0	192 28 27 1 0	193 15 0	194 87 0	195 10 9 1 0
196 66 65 1 0	197 62 61 1 0	198 65 0	199 34 0	200 42 41 1 0
201 14 0	202 55 0	203 8 7 1 0	204 74 73 1 0	205 30 29 1 0
206 29 28 1 0	207 43 0	208 62 59 3 0	209 6 0	210 35 32 3 0
211 46 45 1 0	212 105 0	213 8 7 1 0	214 49 48 1 0	215 23 0
216 196 195 1 0	217 45 0	218 11 0	219 19 18 1 0	220 15 14 1 0
221 35 34 1 0	222 92 91 1 0	223 33 0	224 31 30 1 0	225 32 0
226 58 57 1 0	227 46 45 1 0	228 148 147 1 0	229 64 63 1 0	230 46 45 1 0
231 26 0	232 100 99 1 0	233 74 0	234 31 0	235 10 9 1 0
236 5 0	237 26 25 1 0	238 168 167 1 0	239 36 0	240 121 119 2 0
241 70 0	242 132 131 1 0	243 76 75 1 0	244 40 39 1 0	245 168 167 1 0
246 35 34 1 0	247 82 0	248 65 63 2 0	249 86 0	250 103 0
251 228 227 1 0	252 67 0	253 7 6 1 0	254 19 18 1 0	255 52 0
256 100 99 1 0	257 12 0	258 83 0	259 15 14 1 0	260 21 20 1 0
261 64 63 1 0	262 97 96 1 0	263 93 0	264 10 9 1 0	265 42 0
266 47 0	267 86 85 1 0	268 25 0	269 7 6 1 0	270 53 0
271 58 0	272 108 107 1 0	273 23 0	274 67 0	275 23 22 1 0
276 89 88 1 0	277 70 69 1 0	278 5 0	279 5 0	280 42 41 1 0
281 93 0	282 35 0	283 60 59 1 0	284 119 0	285 106 105 1 0
286 69 0	287 71 0	288 11 10 1 0	289 21 0	290 134 133 1 0
291 107 105 2 0	292 97 0	293 96 95 1 0	294 61 0	295 48 0
296 34 33 1 0	297 5 0	298 30 29 1 0	299 47 46 1 0	300 7 0
301 66 65 1 0	302 41 0	303 29 28 1 0	304 196 195 1 0	305 102 0
306 226 225 1 0	307 117 115 2 0	308 297 296 1 0	309 155 154 1 0	310 16 15 1 0
311 31 30 1 0	312 308 305 3 0	313 79 0	314 15 0	315 10 9 1 0
316 135 0	317 96 95 1 0	318 115 114 1 0	319 36 0	320 4 3 1 0
321 31 0	322 67 0	323 204 203 1 0	324 256 255 1 0	325 76 75 1 0
326 90 89 1 0	327 34 0	328 93 91 2 0	329 50 0	330 16 15 1 0
331 324 323 1 0	332 123 0	333 2 0	334 27 26 1 0	335 42 41 1 0
336 212 211 1 0	337 55 0	338 104 103 1 0	339 194 193 1 0	340 93 92 1 0
341 24 23 1 0	342 125 0	343 75 0	344 260 259 1 0	345 22 0
346 180 179 1 0	347 338 337 1 0	348 128 127 1 0	349 12 11 1 0	350 53 0
351 34 0	352 76 75 1 0	353 69 0	354 119 118 1 0	355 6 5 1 0
356 49 48 1 0	357 70 69 1 0	358 333 332 1 0	359 68 0	360 26 25 1 0
361 45 44 1 0	362 63 0	363 8 7 1 0	364 67 0	365 72 71 1 0
366 29 0	367 21 0	368 114 113 1 0	369 91 0	370 139 0
371 16 15 1 0	372 196 195 1 0	373 100 99 1 0	374 64 63 1 0	375 16 0
376 142 141 1 0	377 41 0	378 43 0	379 114 113 1 0	380 47 0
381 185 183 2 0	382 81 0	383 90 0	384 164 163 1 0	385 6 0
386 83 0	387 68 67 1 0	388 69 68 1 0	389 154 153 1 0	390 89 0
391 28 0	392 346 345 1 0	393 7 0	394 135 0	395 270 269 1 0
396 25 0	397 67 66 1 0	398 101 100 1 0	399 86 0	400 118 117 1 0

Note: "24 4 3 1 0" means $p(x) = x^{24} + x^4 + x^3 + x^1 + x^0 = x^{24} + x^4 + x^3 + x + 1$, where 4 = 3 + 1.

```
401 152 0          402 341 339 2 0    403 150 149 1 0    404 189 0          405 340 337 3 0
406 157 0          407 71 0           408 382 381 1 0    409 87 0           410 156 155 1 0
411 136 131 5 0    412 147 0          413 282 281 1 0    414 46 45 1 0      415 102 0
416 144 143 1 0    417 107 0          418 18 17 1 0      419 166 163 3 0    420 131 130 1 0
421 302 297 5 0    422 149 0          423 25 0           424 66 65 1 0      425 12 0
426 59 57 2 0      427 106 105 1 0    428 105 0          429 412 411 1 0    430 39 38 1 0
431 120 0          432 350 345 5 0    433 33 0           434 164 163 1 0    435 302 301 1 0
436 165 0          437 38 37 1 0      438 65 0           439 49 0           440 4 3  1 0
441 31 0           442 7 5 2 0        443 16 15 1 0      444 55 54 1 0      445 58 57 1 0
446 105 0          447 73 0           448 124 123 1 0    449 134 0          450 79 0
451 196 195 1 0    452 35 34 1 0      453 227 225 2 0    454 36 35 1 0      455 38 0
456 328 327 1 0    457 16 0           458 203 0          459 190 189 1 0    460 61 0
461 7 6 1 0        462 73 0           463 93 0           464 187 186 1 0    465 59 0
466 16 15 1 0      467 360 359 1 0    468 193 189 4 0    469 282 281 1 0    470 149 0
471 1 0            472 25 23 2 0      473 126 125 1 0    474 191 0          475 382 381 1 0
476 15 0           477 193 191 2 0    478 121 0          479 104 0          480 121 115 6 0
481 138 0          482 50 49 1 0      483 428 427 1 0    484 105 0          485 64 63 1 0
486 59 58 1 0      487 94 0           488 4 3 1 0        489 83 0           490 219 0
491 15 14 1 0      492 8 7 1 0        493 204 203 1 0    494 137 0          495 76 0
496 186 185 1 0    497 78 0           498 476 475 1 0    499 372 371 1 0    500 249 248 1 0
501 359 357 2 0    502 153 152 1 0    503 3 0            504 364 363 1 0    505 156 0
506 95 0           507 152 146 6 0    508 109 0          509 255 254 1 0    510 49 48 1 0
511 10 0           512 108 105 3 0    513 85 0           514 22 21 1 0      515 240 239 1 0
516 26 25 1 0      517 346 345 1 0    518 33 0           519 79 0           520 224 221 3 0
521 32 0           522 470 469 1 0    523 202 201 1 0    524 167 0          525 199 197 2 0
526 135 134 1 0    527 47 0           528 302 301 1 0    529 42 0           530 132 131 1 0
531 19 18 1 0      532 1 0            533 100 99 1 0     534 89 88 1 0      535 52 51 1 0
536 52 51 1 0      537 94 0           538 271 270 1 0    539 362 361 1 0    540 179 0
541 180 177 3 0    542 18 17 1 0      543 16 0           544 220 217 3 0    545 122 0
546 119 116 3 0    547 247 245 2 0    548 99 98 1 0      549 247 245 2 0    550 193 0
551 135 0          552 88 87 1 0      553 39 0           554 364 363 1 0    555 263 261 2 0
556 153 0          557 240 239 1 0    558 61 60 1 0      559 34 0           560 210 209 1 0
561 71 0           562 76 75 1 0      563 80 79 1 0      564 163 0          565 82 81 1 0
566 153 0          567 143 0          568 218 215 3 0    569 77 0           570 67 0
571 277 275 2 0    572 285 284 1 0    573 568 567 1 0    574 13 0           575 146 0
576 116 115 1 0    577 25 0           578 72 71 1 0      579 466 465 1 0    580 61 60 1 0
581 140 139 1 0    582 85 0           583 130 0          584 74 73 1 0      585 121 0
586 118 117 1 0    587 46 45 1 0      588 151 0          589 520 519 1 0    590 93 0
591 50 49 1 0      592 352 351 1 0    593 86 0           594 19 0           595 10 9 1 0
596 245 244 1 0    597 58 57 1 0      598 7 6 1 0        599 30 0           600 11 10 1 0
601 201 0          602 35 33 2 0      603 20 19 1 0      604 64 63 1 0      605 19 18 1 0
606 133 132 1 0    607 105 0          608 108 107 1 0    609 31 0           610 127 0
611 39 38 1 0      612 82 81 1 0      613 219 217 2 0    614 75 74 1 0      615 211 0
616 21 19 2 0      617 200 0          618 370 369 1 0    619 202 201 1 0    620 29 28 1 0
621 184 183 1 0    622 297 0          623 68 0           624 16 15 1 0      625 133 0
626 298 297 1 0    627 251 250 1 0    628 223 0          629 362 361 1 0    630 427 426 1 0
631 307 0          632 400 399 1 0    633 101 0          634 315 0          635 188 187 1 0
636 88 87 1 0      637 603 599 4 0    638 6 5 1 0        639 16 0           640 17 15 2 0
641 11 0           642 119 0          643 233 231 2 0    644 229 228 1 0    645 596 595 1 0
646 249 0          647 5 0            648 23 22 1 0      649 37 0           650 3 0
651 152 151 1 0    652 93 0           653 176 175 1 0    654 367 366 1 0    655 88 0
656 248 247 1 0    657 38 0           658 55 0           659 112 111 1 0    660 412 411 1 0
661 204 203 1 0    662 297 0          663 257 0          664 40 39 1 0      665 33 0
666 34 31 3 0      667 631 629 2 0    668 171 170 1 0    669 406 405 1 0    670 153 0
671 15 0           672 106 105 1 0    673 28 0           674 81 79 2 0      675 280 279 1 0
676 241 0          677 31 30 1 0      678 367 366 1 0    679 66 0           680 234 231 3 0
681 193 192 1 0    682 78 77 1 0      683 63 62 1 0      684 155 154 1 0    685 4 3 1 0
686 197 0          687 13 0           688 249 247 2 0    689 14 0           690 541 539 2 0
691 90 85 5 0      692 299 0          693 23 22 1 0      694 70 69 1 0      695 212 0
696 550 549 1 0    697 267 0          698 215 0          699 340 339 1 0    700 238 237 1 0
701 118 117 1 0    702 37 0           703 63 62 1 0      704 156 153 3 0    705 19 0
706 133 131 2 0    707 136 135 1 0    708 287 0          709 4 3 1 0        710 15 14 1 0
711 92 0           712 202 201 1 0    713 41 0           714 23 0           715 7 6 1 0
716 183 0          717 271 269 2 0    718 30 29 1 0      719 150 0          720 215 209 6 0
721 9 0            722 231 0          723 32 31 1 0      724 19 18 1 0      725 160 159 1 0
726 5 0            727 180 0          728 336 335 1 0    729 58 0           730 147 0
731 35 34 1 0      732 77 76 1 0      733 96 95 1 0      734 227 226 1 0    735 44 0
736 354 351 3 0    737 5 0            738 347 0          739 24 23 1 0      740 153 0
741 290 289 1 0    742 241 240 1 0    743 90 0           744 110 109 1 0    745 258 0
746 351 0          747 167 166 1 0    748 304 303 1 0    749 7 6 1 0        750 284 283 1 0
751 18 0           752 656 653 3 0    753 158 0          754 19 0           755 274 273 1 0
756 349 0          757 7 6 1 0        758 235 234 1 0    759 98 0           760 61 59 2 0
761 3 0            762 83 0           763 126 125 1 0    764 181 180 1 0    764 181 180 1 0
766 67 66 1 0      767 168 0          768 122 121 1 0    769 120 0          770 191 189 2 0
771 202 201 1 0    772 7 0            773 350 349 1 0    774 185 0          775 367 0
776 208 207 1 0    777 29 0           778 375 0          779 270 269 1 0    780 239 237 2 0
781 52 51 1 0      782 329 0          783 68 0           784 274 273 1 0    785 92 0
786 32 31 1 0      787 232 231 1 0    788 112 111 1 0    789 226 225 1 0    790 63 62 1 0
791 30 0           792 662 661 1 0    793 253 0          794 143 0          795 346 345 1 0
796 228 227 1 0    797 70 69 1 0      798 311 310 1 0    799 25 0           800 248 245 3 0
```

Note: "800 248 245 3 0" means $p(x) = x^{800} + x^{248} + x^{245} + x^3 + x^0 = x^{800} + x^{248} + x^{245} + x^3 + 1$, where 248 = 245 + 3.