Multiple-Output Low-Power Linear Feedback Shift Register Design

Rajendra S. Katti, *Member*, *IEEE*, and Xiaoyu Ruan, *Student Member*, *IEEE*

Abstract—In this paper we present a new architecture for linear feedback shift registers (LFSR) that produce the output of several clock cycles at once. This enables the frequency of operation to be reduced by a factor equal to the number of outputs produced at a time. This in turn enables the reduction of the power-supply voltage. These two factors dramatically reduce power dissipation in the new architecture of the LFSR. Furthermore the hardware needed is far less than previous low-power implementations of both single and multiple-output LFSRs. Our method is better for BIST applications because it results in more distinct patterns than previously known architectures.

Index Terms—Built-in self-test (BIST), dynamic power dissipation, linear feedback shift registers (LFSR), low power design, multiple output.

Manuscript received _____; revised _____. The work was partially supported by National Science Foundation Grant CCR-0429523. The paper was recommended by Associated Editor _____.

The authors are with the Department of Electrical and Computer Engineering, North Dakota State University, Fargo, ND 58105-5285, USA (e-mail: rajendra.katti@ndsu.edu; xiaoyu.ruan@ndsu.edu).

Digital Object Identifier _____.

1. Introduction

Linear feedback shift registers (LFSR) are used widely in data compression circuitry, encryption circuitry, Built-in self-test (BIST), communication circuitry and error correction circuitry [1,2,3,4,5,6,7,8]. In this paper we consider only Type I LFSRs (defined in [4]), which consist of a bank of D-flip-flops connected serially. The output of some of these is XORed together and fed back to the first flip-flop. The conventional serial architecture of an LFSR with characteristic polynomial, $F(x) = 1+ x^2 + x^5$, is shown in Fig. 1. Here the length of the LFSR (the number of flip-flops), which is denoted by N, is 5 and the number of taps or number of terms XORed, which is denoted by M, is 2. Power consumption in the serial architecture is high as all the flip-flops are clocked in every clock cycle and only one bit of information is generated per clock cycle. The output can be taken from the input or output of any flip-flop. When the output of i successive cycles are generated in one cycle then the LFSR is an i-output (or multiple output) LFSR.

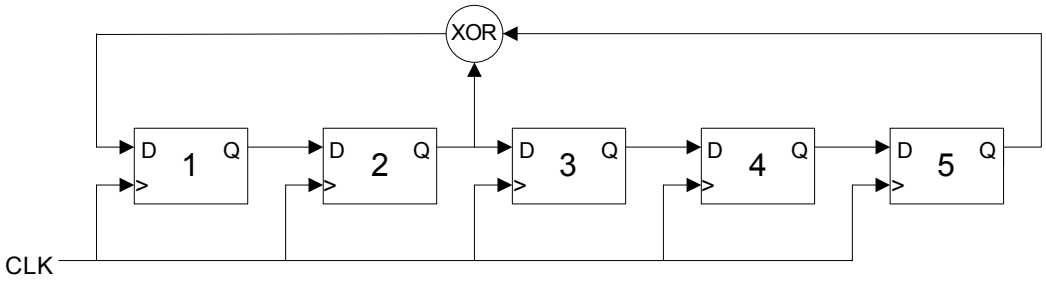

Fig. 1. Serial implementation of the LFSR with $1 + x^2 + x^5$.

The best-known low-power architecture of an LFSR is the one presented in [9]. In [9] only one flip-flop's output changes every clock cycle thereby reducing power dissipation.

However, extra circuitry has to be added to enable this. The architecture in [9] can be described by Fig. 2. In Fig. 2 signal T_i (i = 1, 2,…, N) is obtained from an N-phase generator (a Johnson counter along with some AND gates). T_i is logic-1 in clock cycle i mod N and logic-0 in all other clock cycles. From Fig. 2 we see that the outputs of flip-flops 2 and 5 are XORed together in cycle 1 and the result is stored in flip-flop 5 at the clock edge of cycle 2. This happens because the switches at the output of flip-flops 2 and 5 are turned on by T_1. In cycles 2, 3, 4, and 5 respectively, the outputs of flip-flops (1,4), (5,3), (4,2), and (3,1) are XORed together and stored in flip-flops 4, 3, 2, and 1 respectively, at the clock edges of cycles 3, 4, 5, and 1 respectively. Note that the output of the XOR gate is the output of the LFSR. The operation of the LFSR can be described aptly by Table I.

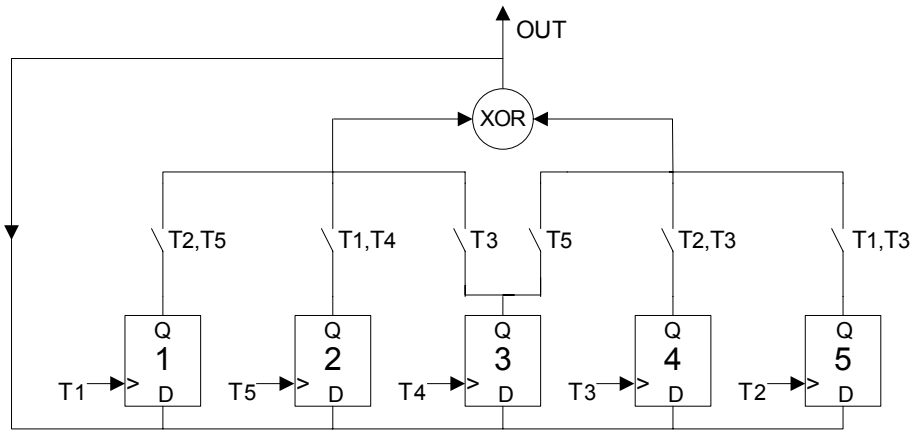

Fig. 2. Low-power implementation of the LFSR with $1 + x^2 + x^5$.

TABLE I
THE OPERATION OF THE LFSR WITH CHARACTERISTIC POLYNOMIAL $1+x^2+x^5$

Cycle number	1	2	3	4	5
Flip-flop outputs that are XORed	2,5	1,4	5,3	4,2	3,1
Flip-flop into which the XORed output is stored	5	4	3	2	1

Note that Table I shows that the result of XORing the outputs of flip-flops 2 and 5 in cycle 1 is stored in flip-flop 5. However, this result is stored in flip-flop 5 in the beginning of clock cycle 2 and not in cycle 1 as shown in the table. This applies to all the clock cycles. The switches that are turned on by more than one T_i are controlled by the ORing of these T_i. Thus a bank of OR gates may be necessary for controlling the switches. In Fig. 2, 4 switches are controlled by 2 T_i's each and hence 4, 2-input OR gates are required. Thus the complete single output LFSR described in [9] consists of an N-phase generator, a maximum of $(N + M)$ switches, N flip-flops, $(M - 1)$ 2-input XOR gates, and a maximum of $(N + M)$, M-input OR gates.

A 2-output LFSR with characteristic polynomial $1 + x^2 + x^5$, is also described in [9]. This circuit consists of $(N + M) + 2N$ more switches than the single output case and each flip-flop is clocked by 2 clocks. Obtaining more than 2 outputs makes the number of switches too high and also the number of clocks, clocking each flip-flop is equal to the number of outputs.

Our new architecture for an LFSR with characteristic polynomial, $1+x^{k_1}+x^{k_2}+\cdots+x^{k_{M-1}}+x^N$, with $k_1 < k_2 < \cdots < k_{M-1} < N$, can generate k_1 outputs in each clock cycle. Therefore the number of M-input XOR gates needed is k_1. The phase

generator needs to generate only $\left\lceil \frac{N}{k_1} \right\rceil$ phases instead of N. The number of switches needed is less than $(N + M)$. Since k_1 outputs are available at a time, the clock frequency and hence the power dissipated reduces by a factor of k_1. Therefore our new architecture is better in terms of hardware and power dissipation than [9]. The multiple output LFSR can be easily converted to a single output LFSR using a multiplexer and latches, which operate at k_1 times the frequency of the multiple output LFSR. However since the multiplexer and latches are the only components operating at the higher frequency and they dissipate very little power, converting to a single output LFSR is very easy.

In the next section we describe our new LFSR architecture, followed by a comparison of hardware complexity and power dissipation of the new architecture with the architecture in [9]. The section that follows this describes an example of our new architecture and quantifies the gain obtained in terms of hardware and power for this example. In Section 4 we perform a comparison of power and the number of distinct patterns generated. Finally the last section concludes the paper.

2. The New LFSR Architecture

We will first describe our architecture for the LFSR with characteristic polynomial $1 + x^2 + x^5$. We refer to the contents of the flip-flops at cycle i as the state in cycle i. From Table I we can see that in cycle 1 the outputs of flip-flops 2 and 5 are XORed together and stored in flip-flop 5 at the clock edge of cycle 2. This new state of flip-flop 5 is used

in cycle 3 when it is input to an XOR gate. Similarly in cycle 2 flip-flops 1 and 4 are XORed together and stored in flip-flop 4 at the clock edge of cycle 3. This new state of flip-flop 4 is used again in cycle 4. This implies that both cycles 1 and 2 can be performed simultaneously because in both these cycles only the initial state of the flip-flops is used. Similarly cycles 3 and 4 can be performed simultaneously and cycle 5 has to be performed by itself. Cycles 1 and 2, change the state of flip-flops 5 and 4, which are then used in cycles 3 and 4. Cycles 3 and 4 change the state of flip-flops 3 and 2, which are then used in cycle 5. This new operation is summarized in Table II below. In the table, 2 clock cycles of Table I are shown as 1 clock cycle.

TABLE II
THE OPERATION OF THE MULTIPLE-OUTPUT LFSR WITH POLYNOMIAL $1+x^2+x^5$

New cycle number	1	2	3
Flip-flops XORed → destination flip-flop	(2,5)→5 (1,4)→4	(5,3)→3 (4,2)→2	(3,1)→1

The output of this LFSR is the output of the XOR gates. Note that in cycle 1 and 2, two outputs are obtained and in cycle 3 only one output is obtained. Our new implementation is shown in Fig. 3.

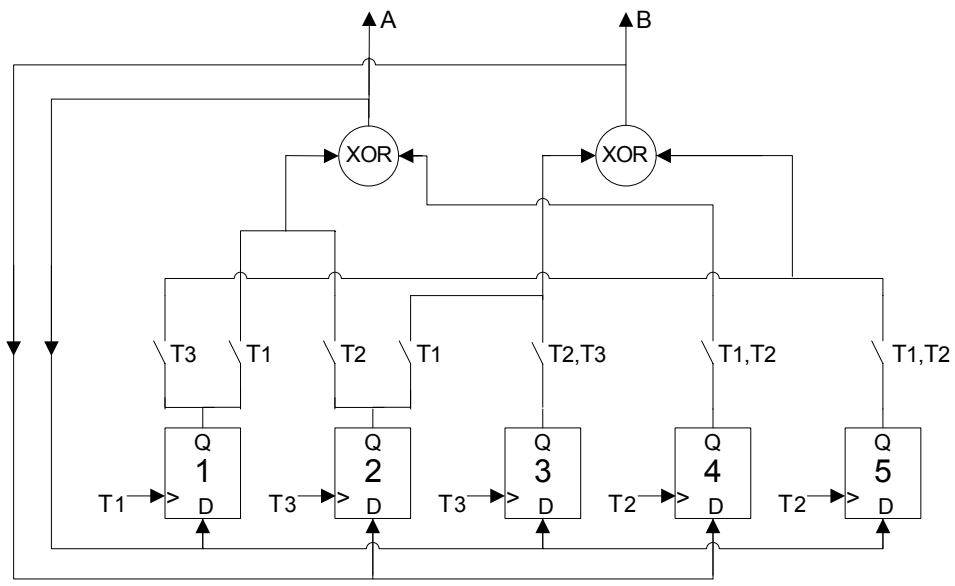

Fig. 3. New multiple-output implementation of the LFSR with $1 + x^2 + x^5$.

From the previous example we see that the number of outputs that can be obtained is 2, which is the exponent of x^2 in $1 + x^2 + x^5$. The total number of phases the phase-generator has to generate is $\lceil 5/2 \rceil$, where 5 is the degree of the characteristic polynomial and 2 is the lowest, non-zero exponent of a term in the characteristic polynomial. The number of switches required in our implementation is always less than ($N + M$) times the number of outputs. In the above example this is 14. The actual implementation has only 7 switches. The implementation in [9] requires 22 switches, which is always less than ($2N + M$) times the number of outputs.

We now describe the general method for the design of an LFSR with characteristic polynomial $1 + x^{k_1} + x^{k_2} + \cdots + x^{k_{M-1}} + x^N$, with $k_1 < k_2 < \cdots < k_{M-1} < N$.

1. Obtain a table similar to Table I for the operation of the LFSR.

2. Combine every k_1 clock cycles into one clock cycle. Therefore each cycle produces k_1 outputs.
3. Form the switch network for each of the k_1, M-input XOR gates. Note that the i^{th} XOR gate produces the i^{th} output, amongst the k_1 outputs produced in each cycle.

The hardware required by our new architecture is,

1. $\left\lceil \frac{N}{k_1} \right\rceil$ phase generator.

2. Approximately $\left\lceil \frac{N}{k_1} \right\rceil$ OR gates, with each OR gate having an average number of inputs equal to $\frac{M}{D_{av}} \left\lceil \frac{N}{k_1} \right\rceil$. Here D_i is the number of distinct inputs, that are input to XOR gate i in $\left\lceil \frac{N}{k_1} \right\rceil$ clock cycles. For example in Table II we see that one of the XOR gates receives inputs from flip-flops (2,5), (5,3), and (3,1) in cycles 1, 2, and 3 respectively. Therefore the number of distinct inputs it receives in these three cycles is, $D_1 = 4$, and they are the outputs of flip-flops 2, 5, 3, and 1. Note that the i^{th} entry in row 2 of all columns refer to inputs to XOR gate i. Therefore from Table II we see that the second entry in row 2 of all columns are (1,4) and (4,2) making $D_2 = 3$. D_{av} is the average of all the D's i.e. $D_{av} = \frac{1}{k_1} \sum_{i=1}^{k_1} D_i$. Each XOR gate has M inputs in every clock cycle. Every $\left\lceil \frac{N}{k_1} \right\rceil$ clock cycles these inputs repeat. Therefore $M \left\lceil \frac{N}{k_1} \right\rceil$ inputs are applied to an XOR gate over $\left\lceil \frac{N}{k_1} \right\rceil$

clock cycles. However only D_{av} distinct inputs exist over $\left\lceil \frac{N}{k_1} \right\rceil$ cycles. This implies that over $\left\lceil \frac{N}{k_1} \right\rceil$ cycles each XOR gate input has a flip-flop output connected to it $\frac{M}{D_{av}}\left\lceil \frac{N}{k_1} \right\rceil$ times. Therefore the switch connected to this input must be turned on $\frac{M}{D_{av}}\left\lceil \frac{N}{k_1} \right\rceil$ times. Thus the OR gate controlling the switch must have $\frac{M}{D_{av}}\left\lceil \frac{N}{k_1} \right\rceil$ phases (or T_i's) as its input.

3. Approximately $k_1 D_{av} = \sum_{i=1}^{k_1} D_i$ switches. At XOR gate i's inputs, D_i distinct flip-flop outputs arrive over $\left\lceil \frac{N}{k_1} \right\rceil$ cycles. For each distinct flip-flop output there must be a switch that connects it to the XOR gate input. Since there are k_1 XOR gates the total number of switches is $k_1 D_{av} = \sum_{i=1}^{k_1} D_i$.

4. N flip-flops.

5. k_1, M-input XOR gates.

As opposed to this the implementation in [9] requires the following hardware.

1. N phase generator.

2. A maximum of $k_1(N + M)$, M-input OR gates.

3. A maximum of $k_1(2N + M)$ switches. A maximum of $(N + M)$ switches are needed for each XOR gates' inputs and since there are k_1 XOR gates the total number of

switches at the inputs of the XOR gates is $k_1(N + M)$. The output of each XOR gate is connected to the N flip-flops via N switches implying that k_1N switches are connected to the outputs of all the XOR gates. Thus the total number of switches required is $k_1(2N + M)$.

4. N flip-flops.
5. k_1, M-input XOR gates.

By choosing the appropriate characteristic polynomial, our method leads to a major reduction in the number of OR gates and switches. Since an N phase generator has $N/2$ flip-flops with N, 2-input NAND gates, our implementation of a $\left\lceil \frac{N}{k_1} \right\rceil$ phase generator requires only $\frac{1}{2}\left\lceil \frac{N}{k_1} \right\rceil$ flip-flops and $\left\lceil \frac{N}{k_1} \right\rceil$, 2-input NAND gates. The number of LFSR flip-flops and XOR gates however still remains the same. The following lemma states that relationship between the number of switches in our architecture and the design in [9].

Lemma 1: The maximum number of switches in our architecture is less than the maximum number of switches in Lowy's architecture [9].

Proof: The number of switches in our architecture is $k_1 D_{av} = \sum_{i=1}^{k_1} D_i$. D_{av} is the average number of distinct flip-flop outputs connected to an XOR gate over $\left\lceil \frac{N}{k_1} \right\rceil$ cycles. Since there are only N flip-flops the maximum value of D_{av} is N. Therefore the maximum number of switches in our architecture is k_1N. This is less than the maximum number of switches required by the architecture in [9] which is $k_1(2N + M)$. □

We now develop the power dissipation equation for our architecture.

Power Dissipation Comparison

We will now compare the power dissipated by our architecture compared to the one in [9]. We will derive equations for the power dissipated for the phase generator, OR gates, flip-flops, and XOR gates. For dynamic power calculation, the same notations and assumptions presented by Hamid and Chen [10] are used to make the comparison simple. The worst-case dynamic power is given by

$$P = \frac{1}{t_p} \times C_{total} \times V_{dd}^2 \times (\text{percentage activity}),$$

where t_p is the clock period, C_{total} is the total capacitance driven by the gate outputs, V_{dd} is the supply voltage, and the percentage activity is 50%.

Other notations used in the calculations are

P_{FF} = power dissipation of the D flip-flop with 1 output capacitance.

P_{clock} = the clock power dissipated by each flip-flop.

P_{XOR} = power dissipation of an XOR gate with 1 output capacitance.

P_{OR} = power dissipation of an OR gate with 1 output capacitance.

P_{AND} = power dissipation of an AND gate with 1 output capacitance.

$P_{\min}$ = power dissipation due to load of the source capacitance of a minimum size transistor.

P_{INV} = power dissipation of an inverter with 1 output capacitance.

N = Number of stages.

M = Number of taps.

Phase Generator Power

The power dissipation in the phase generator is (obtained along similar lines as in [9])

$$P_1 = 2P_{FF} + 2P_{AND}\left(\frac{M}{D_{av}}\left\lceil\frac{N}{k_1}\right\rceil\right) + \frac{1}{2}\left\lceil\frac{N}{k_1}\right\rceil P_{clock}\ .$$

In the above expression the term $\frac{M}{D_{av}}\left\lceil\frac{N}{k_1}\right\rceil$ is the load on the AND gates, which is the number of OR gates the output of an AND gate is connected to. In order to simplify calculations we choose to include the clock power dissipation in the flip-flop power dissipation just as in [10]. The phase generator power dissipation is now given by

$$P_1 = 2P_{FF} + 2P_{AND}\left(\frac{M}{D_{av}}\left\lceil\frac{N}{k_1}\right\rceil\right).$$

The power dissipated by the phase generator in [9] is,

$$P_{1L} = 2P_{FF} + 2P_{AND}(k_1M + k_1)\ .$$

The term $(k_1M + k_1)$ is the load on an AND gate which is k_1M OR gates and k_1 switches providing inputs to the flip-flops.

OR Gates Power

In both our architecture and the one in [9], during each clock cycle k_1M switches are activated. Therefore the power dissipated by the OR gates is $P_2 = k_1MP_{OR}$.

Flip-flops Power

In our architecture each flip-flop is connected to an average of $\frac{k_1D_{av}}{N}$ switches and only k_1 of the flip-flops change state in a clock cycle. Therefore the power dissipated by

the flip-flops is $P_3 = \frac{1}{2} k_1 P_{FF} + \frac{1}{2} \frac{k_1 D_{av}}{N} P_{FF}$. The "$\frac{1}{2}$" in the power in this and other expressions that follow, accounts for the fact that the flip-flop changes state only 50% of the time (percentage activity of the flip-flop). For the architecture in [9] each flip-flop is connected to approximately k_1 switches (this assumes that $(N + M)/N = 1$ [9]) and k_1 flip-flops change state each cycle, therefore the power dissipated by the flip-flops is $P_{3L} = \frac{1}{2} k_1 P_{FF} + \frac{1}{2} k_1 P_{FF}$.

XOR Gates Power

In our architecture k_1 XOR gates are connected to k_1 flip-flops, therefore the power dissipated by the XOR gates is, $P_4 = \frac{1}{2} k_1^2 P_{XOR}$. If an inverter were to drive each of the N flip-flops, then the power dissipated by the XOR gates would be $P_4 = \frac{1}{2} k_1 P_{XOR} + \frac{1}{2} k_1 P_{INV}$. The architecture of Lowy [9] also requires k_1 XOR gates, but each one of these is connected to the drains of N minimum sized transistors (switches). Therefore the power dissipated by the architecture in [9] is, $P_{4L} = \frac{1}{2} k_1 P_{XOR} + \frac{N}{2} k_1 P_{\min}$.

Total Power

Since k_1 outputs are available each clock cycle, the frequency of operation can be reduced by a factor of k_1. Therefore the total power consumed by our architecture is,

$$P = \frac{1}{k_1} \left[2P_{FF} + 2P_{AND} \left(\frac{M}{D_{av}} \left\lceil \frac{N}{k_1} \right\rceil \right) + k_1 M P_{OR} + \frac{1}{2} k_1 P_{FF} \left(1 + \frac{D_{av}}{N} \right) + \frac{1}{2} k_1 P_{XOR} + \frac{1}{2} k_1 P_{INV} \right].$$

The total power consumed by the architecture in [9] is,

$$P_L = \frac{1}{k_1}\left[2P_{FF} + 2P_{AND}\left(k_1 M + k_1\right) + k_1 M P_{OR} + k_1 P_{FF} + \frac{1}{2}k_1 P_{XOR} + \frac{N}{2}k_1 P_{\min}\right].$$

To simplify calculations we choose to ignore P_{INV} and $P_{\min}$. Thus the final expressions become the following.

$$P_{ours} = \frac{1}{k_1}\left[2P_{FF} + 2P_{AND}\left(\frac{M}{D_{av}}\left\lceil\frac{N}{k_1}\right\rceil\right) + k_1 M P_{OR} + \frac{1}{2}k_1 P_{FF}\left(1 + \frac{D_{av}}{N}\right) + \frac{1}{2}k_1 P_{XOR}\right],$$

$$P_{Lowy} = \frac{1}{k_1}\left[2P_{FF} + 2P_{AND}\left(k_1 M + k_1\right) + k_1 M P_{OR} + k_1 P_{FF} + \frac{1}{2}k_1 P_{XOR}\right].$$

The decrease in power dissipation is given by,

$$P_{Lowy} - P_{ours} = \frac{1}{k_1}\left[2P_{AND}\left(k_1 M + k_1 - \frac{M}{D_{av}}\left\lceil\frac{N}{k_1}\right\rceil\right) + \frac{1}{2}k_1 P_{FF}\left(1 - \frac{D_{av}}{N}\right)\right].$$

In the next section we describe the above design with an example. In the section that follows the next we consider built-in self-test (BIST) applications of our proposed LFSR as considered in [10]. Our results indicate that the proposed LFSR, while having reduced hardware complexity, generates more distinct patterns than the ones proposed in [10].

3. Example

In this section we will construct an LFSR with characteristic polynomial $1 + x^3 + x^4 + x^7 + x^{12}$ that demonstrates our LFSR architecture. Comparisons are also made with [9,10] in terms of power dissipation and hardware complexity. We will construct a table similar to Table II to demonstrate the design. Table III shows the table that describes the LFSR

design. The XOR gates and their inputs in the LFSR circuit obtained from Table III are shown in Fig. 4.

TABLE III
THE OPERATION OF THE LFSR WITH POLYNOMIAL $1+X^3+X^4+X^7+X^{12}$

Cycle number	1	2	3	4
Flop-flops XORed	(3,4,7,12)→12	(12,1,4,9)→9	(9,10,1,6)→6	(6,7,10,3)→3
→	(2,3,6,11)→11	(11,12,3,8)→8	(8,9,12,5)→5	(5,6,9,2)→2
destination flip-flop	(1,2,5,10)→10	(10,11,2,7)→7	(7,8,11,4)→4	(4,5,8,1)→1

This LFSR produces 3 outputs every clock cycle. The LFSR needs a 4-phase generator, 4 2-input OR gates, 24 switches, 12 flip-flops, and 3 4-input XOR gates. Note that for this LFSR $D_{av} = 8$, because each row of the "flip-flops-XORed-row" in Table III has 8 distinct flip-flop outputs being XORed. For example there are 8 distinct terms in (3,4,7,12), (12,1,4,9), (9,10,1,6), and (6,7,10,3). Therefore the average number of inputs an OR gate must have is $\frac{M}{D_{av}}\left\lceil \frac{N}{k_1} \right\rceil = \frac{4\times 4}{8} = 2$, and the number of switches required by this LFSR is $k_1 D_{av} = \sum_{i=1}^{k_1} D_i = 8\times 3 = 24$. Lowy's architecture [9] on the other hand requires a 12-phase generator, 24 2-input OR gates, 60 switches, 12 flip-flops, and 3 4-input XOR gates.

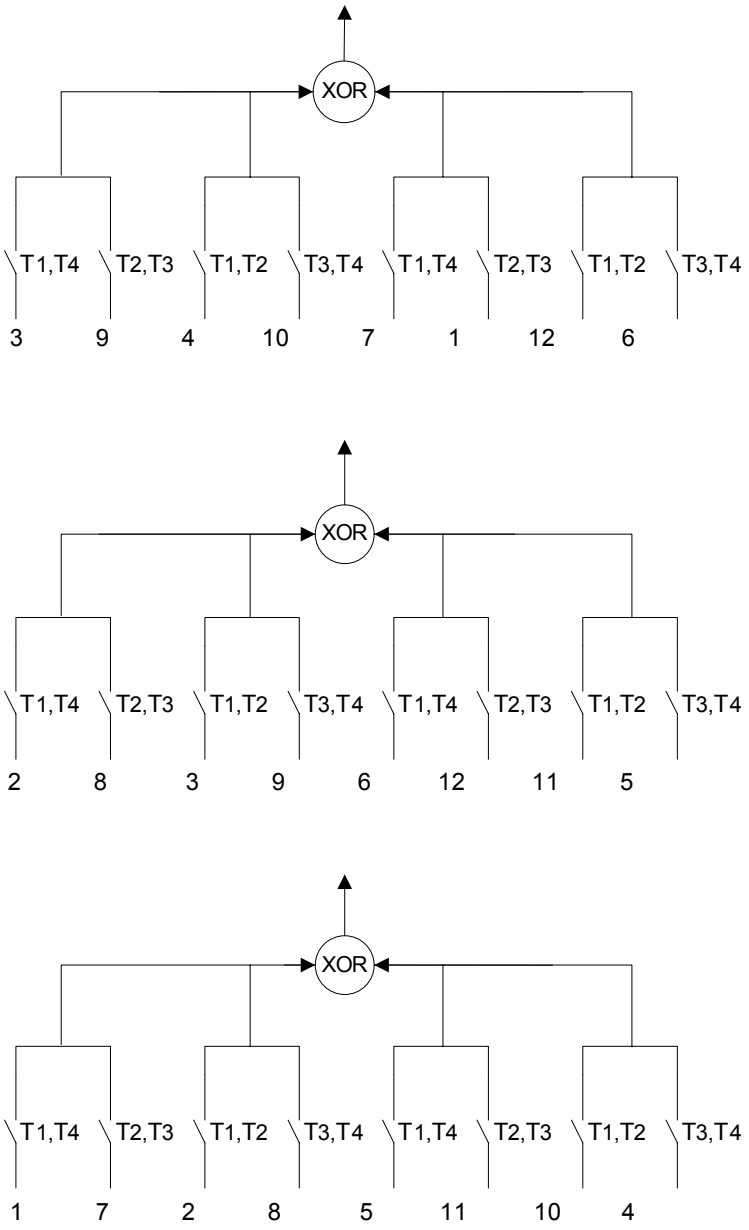

Fig. 4. Implementation of the LFSR with $1 + x^3 + x^4 + x^7 + x^{12}$.

The power dissipated by our architecture is given by,

$$P_{ours} = \frac{1}{3}\left(2P_{FF} + 4P_{AND} + 12P_{OR} + \frac{5}{2}P_{FF} + \frac{3}{2}P_{XOR} \right).$$

The power dissipated by Lowy's architecture [9] is given by,

$$P_{Lowy} = \frac{1}{3}\left(2P_{FF} + 30P_{AND} + 12P_{OR} + 3P_{FF} + \frac{3}{2}P_{XOR} \right).$$

Therefore our architecture consumes lesser power, which is given by,

$$P_{Lowy} - P_{ours} = \frac{1}{3}\left(26P_{AND} + \frac{1}{2}P_{FF} \right).$$

4. Comparison of Power and Distinct Patterns Generated

In this section we consider polynomials of the type $1 + x^{\lceil N/2 \rceil} + x^N$ or $1 + x^{\lfloor N/2 \rfloor} + x^N$ similar to the ones considered in [10]. It was shown in [10] that such polynomials result in a number of switches of order N instead of order $(N + M)$ and that the number of distinct patterns generated is more than that of Lowy's architecture. We show that we can obtain $N/2$ outputs simultaneously if N is odd with hardware requirement that is less than that in [10]. Since multiple outputs have been generated and the hardware required is less the power consumption of our architecture is considerably less than that in [10]. Our design also results in more distinct patterns than [10] and is therefore more suitable for applications like BIST. Note that for even N our design does not generate as many distinct patterns as in [10], however our design results in more distinct patterns if we use polynomials that are not of the form $1 + x^{\lceil N/2 \rceil} + x^N$ or $1 + x^{\lfloor N/2 \rfloor} + x^N$. For example if N is 10 one could use the polynomial $1 + x^3 + x^{10}$ instead of $1 + x^5 + x^{10}$. Using the data in Table IV we can obtain the hardware complexity

and power dissipation equations for polynomials of the kind $1+x^{\lceil N/2 \rceil}+x^N$ or $1+x^{\lfloor N/2 \rfloor}+x^N$ (N odd).

TABLE IV
THE CHARACTERISTICS OF LFSRS WITH POLYNOMIAL $1+x^{\lceil N/2 \rceil}+x^N$ OR $1+x^{\lfloor N/2 \rfloor}+x^N$

	$k_1 = (N+1)/2$	$k_1 = (N-1)/2$
Number of outputs per cycle = k_1	$(N+1)/2$	$(N-1)/2$
Phases = $\left\lceil \frac{N}{k_1} \right\rceil$	2	3
Number of OR gates = $\left\lceil \frac{N}{k_1} \right\rceil$	1 (since there are only two phases)	3
Average number of inputs to each OR gate = $\frac{M}{D_{av}}\left\lceil \frac{N}{k_1} \right\rceil$	4/3	3/2
Maximum D	3	4
Number of switches = $k_1 D$	$3(N+1)/2$	$4(N-1)/2$
Number of XOR gates = k_1	$(N+1)/2$	$(N-1)/2$
Number of flip-flops = N	N	N

Using the general power equations derived by us and putting in the appropriate value of k_1 and D_{av} we can obtain the power dissipation equations. When k_1 is $\left\lceil \frac{N}{2} \right\rceil = \frac{N+1}{2}$, the power dissipation is given by,

$$P_{ours} = \frac{N+3}{2N} P_{FF} + \frac{P_{XOR}}{2} + MP_{INV} .$$

Note that the phase generator consists only of an inverter in this case because only 2 phases have to be generated. When k_1 is $\left\lfloor \frac{N}{2} \right\rfloor = \frac{N-1}{2}$, the power dissipation is given by,

$$P_{ours} = \frac{2}{N-1}(P_{FF} + P_{INV}) + 2P_{OR} + \frac{N+4}{2N}P_{FF} + \frac{P_{XOR}}{2}.$$

In this case the phase generator consists of one flip-flop and an inverter. We now compare the above hardware complexity and power dissipation equations to the LFSRs in [10]. The architecture in [10], for a single output, requires an N-phase generator, ($N + M$), M-input OR gates, about N switches, N flip-flops and one XOR gate. For obtaining k_1 outputs the architecture in [10] requires k_1 XOR gates, $2k_1N$ switches and $k_1(N + 2)$ OR gates. The power dissipation equation for the architecture in [10] for a single output is $\left(3P_{FF} + 4P_{AND} + 2P_{OR} + \frac{P_{XOR}}{2}\right)$ and for k_1 outputs it is the same as for Lowy's case [10] with $M = 2$ and is given by $\frac{1}{k_1}\left(2P_{FF} + 6k_1P_{AND} + 2k_1P_{OR} + k_1P_{FF} + \frac{1}{2}k_1P_{XOR}\right)$. Thus our architecture is superior both in terms of hardware complexity and power dissipation.

Table V compares power dissipation of our architecture with Lowy's architecture [9] for different characteristic polynomials. The data used to obtain the table is for a CMOS 0.18 μ process standard cell library, with capacitances, C_{dff} = 0.0027 pf, C_{XOR} = 0.0042 pf, C_{OR} = 0.0026 pf, C_{INV} = 0.0027 pf, C_{AND} = 00215 pf, and the power supply voltage is V_{dd} = 1.8 V. Frequency is set to 30 MHz. From the table we see that on the average our architecture results in more than 50% improvement in power dissipated if the number of outputs is more than 1. The first column of the table gives the characteristic polynomial of the LFSR, the second column gives the power dissipated by either the architecture in

[9] or [10] whichever is lower and the third column gives the power dissipated by our architecture. Table V also compares the percentage of the maximum of the number of distinct patterns that are generated by our architecture with that generated by single output architectures in [9] and [10] for various polynomials. Columns 5 and 6 give the following quantity: number of distinct outputs in 2^N cycles divided by $2^N - 1$, which is the maximum number of distinct outputs possible. The entries in Column 5 are from [9] or [10] whichever is higher. Column 8 gives the best seeds of our architecture. A seed given in the table is an integer whose N-bit binary equivalent gives the initial values of the flip-flops. Therefore if the seed is $\sum_{i=1}^{N} S_i \times 2^{i-1}$, then S_i ($1 \le i \le N$) is the binary value that the i^{th} flip-flop is initialized to. If several seeds result in the same maximum percentage then the smallest seed is given. We compare our multiple output polynomials with previous single output architectures because the hardware overhead for previous architectures with multiple outputs is too high thus making them unusable. For some of the polynomials in Table V our architecture results in a single output LFSR because $k_1 = 1$. From Column 6 of the table we see that the average percentage of the number of distinct patterns generated is close to 100%, thereby making our architecture suitable for BIST applications.

TABLE V
POWER AND PERCENT DISTINCT OUTPUT COMPARISON

Polynomial	Power from [9,10] (μW)	Power from ours (μW)	% improved	2^N cycles (%) from [9,10]	2^N cycles (%) from ours	% improved	Seed from ours
$1+x+x^3$	2.75	2.28	17.1	71.4	100	28.6	1
$1+x+x^4$	2.75	2.28	17.1	60.0	100	40.0	1
$1+x^2+x^5$	2.33	1.208	48.15	25.8	100	74.2	1
$1+x+x^6$	2.75	2.28	17.1	46.0	62.3	16.3	1
$1+x^4+x^7$	2.33	0.915	60.73	26.8	100	73.2	1
$1+x^4+x^5+x^6+x^7$	4.5	2.14	52.44	36.2	100	63.8	1
$1+x+x^5+x^6+x^8$	4.5	4.08	9.33	38.8	100	61.2	1
$1+x^4+x^9$	2.33	1.03	55.8	40.9	100	59.1	1
$1+x^3+x^{10}$	2.4	1.33	44.6	45.9	100	54.1	1
$1+x^3+x^4+x^7+x^{12}$	3.74	1.89	49.5	N/A	75.9	N/A	4

5. Conclusion

We have presented a new multiple-output LFSR architecture that results in lower hardware complexity and lower power dissipation than previously known architectures. This was proved by deriving expressions for the number of hardware components and for the power dissipated. We also showed that our architecture is very useful for BIST applications because of its ability to generate distinct patterns.

Acknowledgments

We would like to acknowledge the assistance of Abdullah Mamun in obtaining the capacitances of the required circuits.

References

1. Davida, G. I. and Rodrigues, L., "Data Compression Using Linear Feedback Shift Registers", *Proceedings of the IEEE Data Compression Conference*, Los Alamitos, CA, pp. 475, 1994.
2. Moon, T. K. and Veeramachaneni, S., "Linear Feedback Shift Registers as Vector Quantisation Codebooks", *Electronics Letters*, Vol. 35, No. 22, pp. 1919-1920, October 1999.
3. Jamil, T. and Ahmad, A., "An Investigation into the Application of Linear Feedback Shift Registers for Steganography", *Proceedings of IEEE SOUTHEASTCON Conference*, Columbia, SC, pp. 239-244, 2002.
4. Abromovici, M., Breuer, M. A., and Friedman, A. D., "Digital Systems Testing and Testable Design", revised printing, IEEE Press, Piscataway, NJ.
5. Lee, J. S. and Miller, L. E., "CDMA Systems Engineering Handbook", Artech House Publishers, Boston, 1998.
6. Alspector, J., Gannett, J. W., Haber, S., Parker, M. B., and Chu, R., "Generating Multiple Analog Noise Sources from a Single Linear Feedback Shift Register with Neural Network Applications", *Proceedings of IEEE International Symposium on Circuits and Systems*, New Orleans, LA, Vol. 2, pp. 1058-1061, 1990.
7. Golomb, S. W., "Shift Register Sequences", Holden-Day, Inc., San Francisco, CA, 1967.

8. Xilinx, Inc., “Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators”, http://www.xilinx.com/bvdocs/appnotes/xapp052.pdf, date accessed: February 15, 2003, date written: October 11, 2003.
9. Lowy, M., “Parallel Implementation of Linear Feedback Shift Registers for Low Power Applications”, *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, Vol. 43, pp. 458-466, June 1996.
10. Hamid, M. E. and Chen, C. I. H., “A Note to Low-Power Linear Feedback Shift Registers”, *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing,* Vol. 45, No. 9, pp. 1304-1307, September 1998.