Exhaustive Testing for Stuck-Open and Delay Faults

Chih-Ang Chen and Sandeep K. Gupta

CENG Technical Report 92-09

Department of Electrical Engineering - Systems
University of Southern California
Los Angeles, California 90089-2562
(213)740-2251

August 1992

Exhaustive Testing for Stuck-Open and Delay Faults*

Chih-Ang Chen
Electrical Engineering – Systems
University of Southern California
Los Angeles CA 90089-2562

Sandeep K. Gupta
Electrical Engineering – Systems
University of Southern California
Los Angeles CA 90089-2562

Abstract

Desire for higher quality tests is causing increasing interest in testing for delay and CMOS stuck-open faults. These require two pattern tests and tests sets are usually large. Built-in self-test (BIST) schemes are attractive for such comprehensive testing. The BIST test pattern generators (TPGs) should be designed to ensure high pattern pair coverage. In this paper, necessary and sufficient conditions to ensure complete pattern pair coverage for linear feedback shift register (LFSR) and cellular automata (CA) have been studied. The TPGs designed by using these conditions guarantee 100% coverage of all single and two pattern testable faults. It is shown that LFSRs with primitive feedback polynomials, with large number of terms of the form x^{2i}, are better for two pattern testing. CAs are good TPGs for two pattern testing, independent of their feedback rules.

If the test sequence length is unacceptable, then there is a need to trade-off pattern pair coverage to lower test sequence length. A procedure has been developed which helps design TPGs which maximize pattern pair coverage for a given test length constraint. Robust path delay fault coverage on some benchmark circuits indicate the TPGs designed using the procedures outlined in this paper provide much higher fault coverage than other TPGs.

Keywords: BIST, test pattern generators, exhaustive testing, two pattern testing, LFSR, cellular automata.

*This research was funded by NSF Research Initiation Award no. MIP-9210871

1 Introduction

Traditional focus of VLSI testing in general, and BIST in particular, has been to maximize the coverage of stuck-at faults. (Also, most BIST techniques design test pattern generators for each combinational logic block separately. In this paper, the circuit under test (CUT) will be assumed to be combinational.) Some techniques, such as pseudo-exhaustive testing, go beyond single stuck-at faults and implicitly target all multiple stuck-at faults and some bridging faults (those that do not convert the combinational circuit into a sequential one). However, a large class of physical defects do not map into these categories. Some physical defects, such as transistor stuck-open faults in CMOS, convert the combinational CUT into a sequential one. This is due to the fact that, for a given vector, output of the faulty circuit can take different values depending on the *state* of the circuit. (The *state* of the combinational circuit is determined by the charge stored at various parasitic, diffusion, and gate capacitances in the circuit due to the previously applied vector.) Hence, stuck-open faults require two pattern tests. Similarly, delay faults require two pattern tests since the delays in the circuit depend on the previous *state* of the CUT.

There is growing interest in coverage of these faults. Firstly, increasing quality level requirements constantly require better testing methodologies. Furthermore, increasing clock rates and use of aggressive statistical timing can cause a circuit to malfunction even if each device in a fabricated chip performs within its worst case delay tolerance limits. Hence, chip manufacturers are starting to augment their testing methodologies to test for delay faults.

One important consideration in design of BIST TPGs for two-pattern testing is to ensure that adequate number of pattern pairs are applied to the combinational CUT. This is the main focus of this paper. Necessary and sufficient conditions for a TPG to achieve complete pattern pair coverage have been derived. It has been shown that LFSRs with primitive feedback polynomials which contain many terms of the form x^{2i} are better for two pattern testing. Cellular automata have been shown to have large number of ways in which complete transition coverage can be achieved. A TPG design procedure which maximizes transition coverage under a given test time constraint has been developed and validated with

robust path delay fault simulation on synthesis benchmark circuits.

The paper is organized into five main sections. Section 2 reviews the related research. Rules for designing TPGs to achieve complete transition count are discussed in Section 3. Section 4 provides a procedure to design TPG to maximize transition count for a given test sequence length constraint. Robust path delay simulation results discussed in Section 5 show that TPGs designed using the procedures developed in this paper are superior to other TPGs. Finally, concluding remarks and future research directions are presented in Section 6.

2 Review of Previous Results

Two-pattern exhaustive testing is defined as testing where the test sequence contains all possible (V_1, V_2) pattern pairs $(V_1 \neq V_2)$. Advantages of two-pattern exhaustive testing are many. Firstly, two-pattern exhaustive testing is very comprehensive. Such testing will test for all faults detectable by one or two patterns. These include all detectable stuck-at, stuck-open, delay, and bridging faults. In case of stuck-open and delay faults, all faults which have robust tests are tested robustly. Those which are not robustly testable are tested in all possible ways in which they can be tested. Besides, no elaborate fault-simulation, estimation of fault coverage (for a given test length), or test length estimation (for desired fault-coverage) is required. However, the biggest drawback of exhaustive testing is the high test sequence length.

Two-pattern pseudo-exhaustive testing for an n-input circuit requires that all possible 2^n vectors are applied as V_1, each followed by all possible $2^n - 1$ $V_2(\neq V_1)$. Hence the number of vector pairs required is $2^n(2^n - 1)$. As the pairs may be suitably ordered, the number of vectors required is bounded by $2^n(2^n - 1) = 2^{2n} - 2^n$. Since $2^{2n-1} < 2^{2n} - 2^n$, a finite state machine with $2n$-states is necessary to generate these test patterns.

It has been shown in [Smi85] that, for path delay faults, all robustly detectable path delay faults can be detected by using (V_1, V_2) vector pairs where V_1 and V_2 differ in only one bit. This is called adjacency testing and it cuts down the number of (V_1, V_2) pairs required for exhaustive testing. Design of circuits which generate such (V_1, V_2) pattern pairs exhaustively

are presented in [CK85]. In [CK85], circuits for pseudo-exhaustive adjacent testing have also been proposed. Note that there are many delay faults which are not robustly testable. The coverage of these faults is not guaranteed by adjacency testing. Exhaustive two pattern testing is studied in [Sta84]. It is shown that it is necessary and sufficient to have a $2n$-bit LFSR to test an n-input circuit two-pattern exhaustively.

The notion of AC strength is introduced in [SB91] to quantify the limiting pattern coverage of a test pattern generator. (The notion is general and applies to scan chains as well, where it represents the fraction of all possible two pattern tests which can be applied via a given chain.) It is defined as

$$\text{AC strength} = \frac{\text{maximum two pattern count}}{2^n(2^n - 1)} \tag{1}$$

The transition coverage of some autonomous linear pattern generators was studied in [FM91]. The transfer function of an n-stage autonomous linear sequential TPG is represented by a transition matrix T given by

$$T = \begin{pmatrix} g_{11} & g_{12} & g_{13} & \cdots & g_{1n} \\ g_{21} & g_{22} & g_{23} & \cdots & g_{2n} \\ g_{31} & g_{32} & g_{33} & \cdots & g_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{n1} & g_{n2} & g_{n3} & \cdots & g_{nn} \end{pmatrix} \tag{2}$$

where $g_{ij} \in \{0, 1\}$ is the (i, j)-entry of matrix T. In general, the next state x' and current state x of a TPG are related by

$$x' = Tx \tag{3}$$

In [FM91], a metric called transition count has been proposed. It is a measure of pattern pair coverage obtained by using test sequence generate by a TPG. Transition count for an n input circuit is $\leq 2^{2n} - 1$. If a test sequence contains all possible pattern pairs, then it is said to have *complete transition count (coverage).*

Typically, the number of stages in TPG, $m > n$, the number of stages in the CUT. Hence, only a subset of TPG stage outputs are connected to the CUT inputs. Let $v =$

$\{v_1, v_2, \ldots, v_m\}$ be the flip-flops whose outputs are connected to the CUT inputs. These shall be called the *tapped* variables. Similarly, let $u = \{u_1, u_2, \ldots, u_{n-m}\}$ denote the *untapped* variables. Let $X_v = \{x_{v_1}, x_{v_2}, \ldots, x_{v_m}\}$ and $X_u = \{x_{u_1}, x_{u_2}, \ldots, x_{u_{n-m}}\}$ be the states of the tapped and untapped stages of the TPG, respectively. Then, the next state function of the tapped variables can be represented by

$$X_v{}' = T_v X_v + T_u X_u \tag{4}$$

where T_v and T_u are submatrices of T of the sizes $m \times m$ and $m \times (n-m)$. The submatrix T_u, given by

$$T_u = \begin{pmatrix} g_{v_1 u_1} & g_{v_1 u_2} & g_{v_1 u_3} & \cdots & g_{v_1 u_{(n-m)}} \\ g_{v_2 u_1} & g_{v_2 u_2} & g_{v_2 u_3} & \cdots & g_{v_2 u_{(n-m)}} \\ g_{v_3 u_1} & g_{v_3 u_2} & g_{v_3 u_3} & \cdots & g_{v_3 u_{(n-m)}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{v_m u_1} & g_{v_m u_2} & g_{v_m u_3} & \cdots & g_{v_m u_{(n-m)}} \end{pmatrix} \tag{5}$$

is constructed from the m rows $\{v_1, v_2, \ldots, v_m\}$ of T, with the corresponding m columns removed. It has been shown that if r is the rank of T_u, then there are 2^r distinct transitions from each v-state. The rank of T_u is important in determining the two-pattern transition coverage of a TPG tap selection. To obtain maximum transition coverage, the submatrix T_u must have full rank, i.e. $r = \min\{m, n-m\}$. The average transition coverage

$$\widehat{2^r} = \frac{1}{\binom{n}{m}} \sum_{\text{all } v\text{-spaces}} 2^r \tag{6}$$

is then used as a metric of the two-pattern test capabilities of a TPG.

In the following transition count shall be used to evaluate the quality of a TPG. When complete transition count is achievable, then detection of all single and two-pattern testable faults is guaranteed. However, to reduce test length and TPG hardware cost, test sequences with lower transition count might have to be used. In such cases, the TPG is designed to maximize transition count with the notion that maximizing transition count would indeed maximize fault coverage.

Recently, in [ZBM92], further study on transition coverage has been reported. Linear test pattern generators (Linear Feedback Shift Registers and Linear Hybrid Cellular Automata) are studied and the transition count has been computed. Two new test pattern generators, XLFSRs and XLHCA, are defined. (An XLFSR is an LFSR whose odd stage outputs and even stage outputs are grouped together. XLHCA is obtained similarly using LHCA.) This maximizes transition coverage if k CUT inputs are connected to k contiguous outputs of these pattern generators. In the following, we shall show that there are a large number of other TPGs with maximum transition count.

3 Design of TPGs for Complete Transition Count

As has been seen above, for an n input CUT, a $2n$ stage TPG is required to achieve complete transition count. In the following discussion, two types of TPGs shall be studied. These are the common TPGs used in most BIST implementations.

Linear feedback shift register (LFSR): External XOR (Type 1) as well as internal XOR (Type 2) LFSRs shall be studied.

Linear cellular automata (CA): Rule 90/150 CA with null boundary conditions [Bar90, SSMM90] are studied.

Necessary and sufficient conditions to obtain complete transition coverage for a CUT shall be derived.

3.1 Linear Feedback Shift Registers

LFSRs are a class of linear sequential logic network constructed from D flip-flops and modulo-2 adders (XOR gates). An n-stage LFSR is characterized by its feedback polynomial given by

$$P(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_n x^n \tag{7}$$

The value of c_0 and c_n must always be 1 for an LFSR. If the feedback polynomial is primitive [LC83], then the LFSR (initialized to any non-zero state) generates a sequence of length $2^n - 1$. Such LFSRs are called maximal length LFSRs.

The transition matrices T_{LFSR1} and T_{LFSR2} of type 1 and type 2 LFSRs are given by

$$T_{LFSR1} = \begin{pmatrix} c_1 & c_2 & c_3 & \cdots & c_{n-1} & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{pmatrix} T_{LFSR2} = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & c_1 \\ 0 & 1 & 0 & \cdots & 0 & c_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & c_{n-2} \\ 0 & 0 & 0 & \cdots & 1 & c_{n-1} \end{pmatrix} \quad (8)$$

As has been shown above, at least $2n$-stage LFSR/CA TPGs are required for generating exhaustive pattern pairs for an n input circuit. It has been shown in [Sta84] that $2n$-stage external XOR LFSR can generate two-pattern exhaustive test set for any n-input circuit. The n circuit inputs can be connected to all odd/even stage outputs of the LFSR. The following results shows that this is true even for internal XOR LFSRs. First, we present a Lemma used in the Theorem.

Lemma 1 *Given a $2n$-stage LFSR and an n-input CUT, if any two consecutive stages i and $i+1$, for $1 \leq i \leq 2n-1$, are (not) connected to CUT inputs for a type 1 (type 2) LFSR, then the LFSR can not generate exhaustive two-pattern tests.*

Proof: The transition matrices T_{LFSR1} and T_{LFSR2} with stage i and $i+1$ highlighted are given

$$T_{LFSR1} = \left(\begin{array}{cc|cc|cc} \cdots & \cdots & c_i & c_{i+1} & \cdots & \cdots \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \hline \cdots & 1 & 0 & 0 & \cdots & 0 \\ \cdots & 0 & 1 & 0 & \cdots & 0 \\ \hline \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \end{array}\right) T_{LFSR2} = \left(\begin{array}{cc|cc|cc} \cdots & \cdots & 0 & 0 & \cdots & \cdots \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \hline \cdots & 1 & 0 & 0 & \cdots & c_{i-1} \\ \cdots & 0 & 1 & 0 & \cdots & c_i \\ \hline \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \end{array}\right) \quad (9)$$

Type 1 LFSR: If stage i and $i+1$ are selected, then rows i and $i+1$ (with columns i and $i+1$ removed) are included in T_u. Since $g_{(i+1)i}$ is the only non-zero entry in row $i+1$, the row in T_u that corresponds to row $i+1$ in T_{LFSR1} must be zero.

Type 2 LFSR: If stage i and $i+1$ are not selected, then the n taps must be selected from the remaining $2n-2$ stages. Since $g_{(i+1)i}$ is the only non-zero entry in column i, no matter how the n taps are selected, T_u always has a zero column.

In both cases, the rank of T_u is less than n and the LFSR will not generate exhaustive two-pattern tests. **Q.E.D.**

Lemma 2 *Any $2n$-stage LFSR (external or internal XOR) generates exhaustive two-pattern tests for an n-input CUT, whose inputs are connected to all its odd or even stage outputs.*

Proof: As discussed in previous section, for a TPG with the transition matrix T, the rank of the submatrix T_u of T for a given choice of tapped variables determines the two-pattern transition coverage of the TPG. For a $2n$-stage LFSR with n outputs connected to the n inputs of a CUT, the rank of T_u must be n to achieve exhaustive two-pattern transition coverage. Using the construction in Eq. (5), the submatrices T_u's of the transition T_{LFSR1} (T_{LFSR2}) for a type 1 (type 2) LFSR with its odd or even stages connected to the inputs of the CUT are

$$T_{LFSR1}^{odd} = \begin{pmatrix} c_1 & c_3 & c_5 & \cdots & c_{2n-3} & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{pmatrix} T_{LFSR2}^{odd} = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & c_2 \\ 0 & 1 & 0 & \cdots & 0 & c_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & c_{2n-2} \end{pmatrix} \quad (10)$$

$$T_{LFSR1}^{even} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} T_{LFSR2}^{even} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \quad (11)$$

The submatrices T^{even}_{LFSR1} and T^{even}_{LFSR2} are simply the $n \times n$ identity matrix which has rank n. By moving the last column of T^{odd}_{LFSR1} or T^{odd}_{LFSR2} to the first column, an upper or lower triangular matrix with unity diagonal elements is formed. The triangular matrices also have rank n. In all cases, the rank of T_u has the value n. Hence, an $2n$-stage LFSR with its odd/even outputs connected to the n inputs of a CUT will generate $2^n(2^n - 1)$ exhaustive pattern pairs irrespective of the LFSR types and feedback polynomials. **Q.E.D.**

Theorem 1 *Given an n-input CUT and 2n-stage LFSR with feedback polynomial* $P(x) = c_0 + c_1x + c_2x^2 + \cdots + c_{2n}x^{2n}$, *n out-of 2n TPG state outputs must be selected by using one of the following rules to achieve exhaustive two pattern testing.*

Type 1 LFSR:

Rule 1: *Select all odd (or all even) stage outputs, or*

Rule 2a: *Select stages*

$$\underbrace{1, \ldots, 2i-1}_{odd}, \underbrace{2i+2, \ldots, 2n}_{even}$$

for any i such that $c_{2i} = 1$.

Type 2 LFSR:

Rule 1: *Select all odd (or all even) stage outputs, or*

Rule 2b: *Select stages*

$$\underbrace{2, \ldots, 2i}_{even}, \underbrace{2i+1, \ldots, 2n-1}_{odd}$$

for any i such that $c_{2i} = 1$.

Proof: Rule (1) has been shown to be sufficient in the Lemma 2.

Type 1 LFSR: If CUT is not connected to all odd (or all even) stages, then two consecutive stages must be selected. From Lemma 1 no two consecutive stages $(i, i+1)$ can be selected for any $0 \leq i \leq 2n-1$. Hence, any other choice of stages (which are not all even or all odd), to obtain complete transition count, must include selecting stages 1 and $2n$. If both stage 1 and stage $2n$ are chosen, there exist two consecutive stages k and $k+1$, for $1 < k < 2n-1$, which are untapped.

If the number k is odd, then all even stages before stage k and all odd stages after stage $k+1$ must be selected. This implies that stage 2 and stage $2n-1$ are tapped. Since both stage 1 and 2 and stage $2n-1$ and $2n$ are selected, Lemma 1 is violated. Hence, the number k must be even. As shown earlier, two consecutive untapped stages k and $k+1$ will void the non-zero entry $g_{(k+1)k}$ in column k of the transition matrix T_{LFSR1}. In order that the matrix in Eq. (4) still has full rank, the feedback coefficient c_k must be 1 (i.e. there exists a feedback XOR gate between stage k and $k+1$). If $c_k = 1$, then the resulting T_u can be shown to have rank n. Since c_0 and c_{2n} are 1 for LFSR, they account for the all-odd and all-even tap selections. Therefore, the number of possible tap selection to achieve exhaustive two pattern tests is simply the number of nonzero coefficients c_{2i}, for $0 \leq i \leq n$.

Type 2 LFSR: Similar arguments can be used to show that, the only ways to connect n input to CUT to $2n$ stage type 2 LFSR are given by Rules 1 and 2b, above. **Q.E.D.**

Corollary 1 *Given an n-input CUT and a 2n-stage LFSR with feedback polynomial* $P(x) = c_0 + c_1x + c_2x^2 + \cdots + c_{2n}x^{2n}$, *the number of possible ways to connect the n inputs of CUT to n outputs of LFSR for two-pattern exhaustive testing is given by* $\sum_{i=0}^{n} c_{2i}$, *for* $0 \leq i \leq n$.

Proof: Proof follows directly from the above theorem.

Example 1 *Consider a 5 input CUT. As described earlier, a 10-stage LFSR is necessary to test the circuit exhaustively with all two-pattern vector pairs. Two possible connections (independent of the feedback polynomial) from LFSR outputs to CUT inputs are shown in Figure 1.*

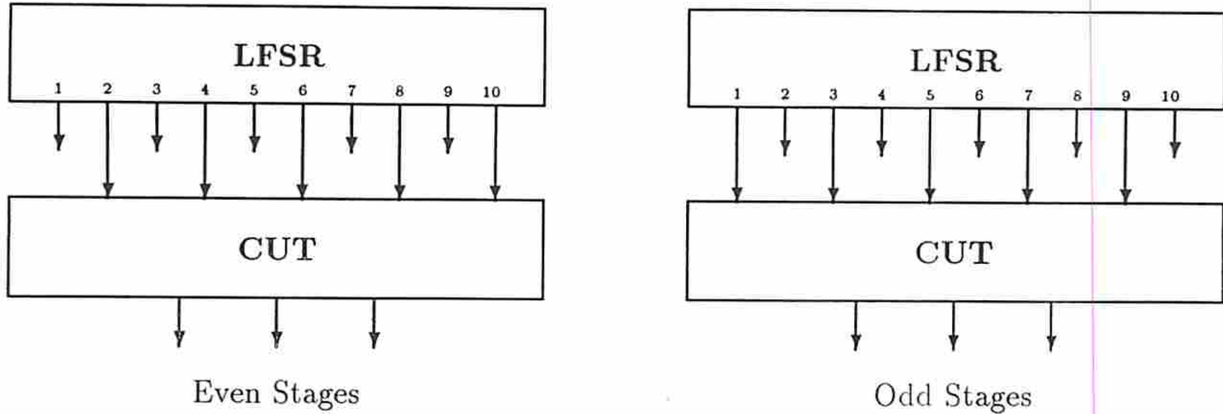

Figure 1: LFSR for 5 input CUT

Example 2 *Consider a 5 input CUT and a 10 stage LFSR with generator polynomial given by* $P(x) = 1 + x^2 + x^3 + x^4 + x^8$. *Since the coefficient* $c_0, c_2, c_6,$ *and* c_8 *are 1, the number of possible tap selections to achieve maximal transition coverage is 4. For type 1 LFSR they are* $(1,3,5,7)$, $(2,4,6,8)$, $(1,4,6,8)$, *and* $(1,3,6,8)$. *For type 2 LFSRs, the four selections are* $(1,3,5,7)$, $(2,4,6,8)$, $(2,3,5,7)$, *and* $(2,4,5,7)$.

If the feedback polynomial is primitive, then the LFSR generates maximal length sequence and $2^{2n} - 1$ distinct vector pairs are applied to the CUT. On the other hand, the test generation is more involved for LFSRs with non-primitive polynomials. The LFSR is initialized to a non-zero state. Since the feedback polynomial is not primitive, the LFSR state repeats in less than $2^{2n} - 1$ clock cycles. When the initial state repeats, the LFSR is reinitialized to a non-zero state that has not been previously covered. If this reseeding process is repeated till all states have been covered, then the CUT is tested two-pattern exhaustively. Hence, the control of LFSRs with non-primitive feedback is much harder and LFSRs with primitive feedback polynomials are recommended. Further, the above results show that LFSRs with feedback polynomials with large number of even powers of x, provide larger number of ways of connecting n CUT inputs to the TPG stages for complete transition coverage. If, in a multi-output circuit, each output function depends on a subset of the inputs, then, logic feeding many of the outputs may be two pattern exhaustively tested. Hence, LFSRs with primitive feedback polynomials with large number of even powers of x

should be used as TPGs for two pattern testing.

3.2 Cellular Automata

The cellular automata considered in the following are linear, rule 90/150 with null boundary conditions [Bar90, SSMM90]. The property of the state transition matrix of a 90/150 CA is that the value at stage-i is always a function of the outputs of the $(i-1)$-th and $(i+1)$-th stages. The only exceptions are the first and last stages which are both connected to only one neighbor (null boundary condition).

The transition matrix T_{CA} of a CA with null boundary conditions is given by

$$T_{CA} = \begin{pmatrix} c_1 & 1 & 0 & \cdots & 0 & 0 \\ 1 & c_2 & 1 & \cdots & 0 & 0 \\ 0 & 1 & c_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & c_{n-1} & 1 \\ 0 & 0 & 0 & \cdots & 1 & c_n \end{pmatrix} \tag{12}$$

where $c_i = 1(0)$ if stage i is rule 150(90).

Theorem 2 *Given any $2n$-stage rule 90/150 linear CA with null boundary conditions. Let n CA outputs be selected (tapped) by using the following construction.*

1. *Pair successive (odd, even) stages together.*
2. *Select one output from each pair.*

The n input CUT is two-pattern exhaustively tested, if and only if, the n CUT inputs are connected to the n outputs of CA selected by using the above construction.

Proof: Before starting, several rules which are essential in the proof of the theorem are given. These rules must be satisfied by the desired tap selection.

Rule 1: Either stage 1 or stage 2, but not both, must be included.

Rule 2: Either stage $(2n-1)$ or stage $2n$, but not both, must be included

Rule 3: No three consecutive stages should be all selected or unselected.

These rules can be easily verified by the reduced transition matrix for the selected variables in Eq. (4). In order to generate exhaustive two-pattern tests, T_u in Eq. (4) must be of full rank n.

$$T_{CA} = \left(\begin{array}{cc|cccc|cc} c_1 & 1 & 0 & \cdots & 0 & & 0 & 0 \\ 1 & c_2 & 1 & \cdots & 0 & & 0 & 0 \\ \hline 0 & 1 & c_3 & \cdots & 0 & & 0 & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & c_{2n-2} & & 1 & 0 \\ \hline 0 & 0 & 0 & \cdots & 1 & & c_{2n-1} & 1 \\ 0 & 0 & 0 & \cdots & 0 & & 1 & c_{2n} \end{array}\right) \quad (13)$$

Rule 1: If both stage 1 and 2 are selected, then the submatrix T_u must include row 1 and 2 of T_{CA} with column 1 and 2 removed. There are $n-2$ other taps to be selected. Once a tapped stage i is determined, row i of T_{CA} is added to T_u and the entries in column i of T_u are removed. Since row 1 of T_u is all 0's after tapping stage 1 and 2, no matter what the other $n-2$ taps are, the submatrix T_u always have a row with all 0 entries and the rank of T_u must be less than n. A CA that generates two-patterns exhaustive can not have both stage 1 and 2 tapped.

If both stage 1 and 2 are not selected, then the n taps must be selected from the remaining $2n-2$ rows of T_{CA} with column 1 and 2 included in T_u. However, column 1 of T_u is all 0's if both stage 1 and 2 are not tapped. No matter how the n taps are selected, column 1 of T_u must be all 0's. Therefore, the tap selection that generates two-pattern exhaustive tests must choose one of stage 1 or 2, but not both.

Rule 2: Symmetrically, the same argument applies if none or both stage $n-1$ and n are selected. In these cases, column n or row n of T_u will be null respectively.

$$T_{CA} = \left(\begin{array}{cc|ccc|cc}
\ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \\
\cdots & c_{i-2} & 1 & 0 & 0 & 0 & \cdots \\
\hline
\cdots & 1 & c_{i-1} & 1 & 0 & 0 & \cdots \\
\cdots & 0 & 1 & c_i & 1 & 0 & \cdots \\
\cdots & 0 & 0 & 1 & c_{i+1} & 1 & \cdots \\
\hline
\cdots & 0 & 0 & 0 & 1 & c_{i+2} & \cdots \\
\ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}\right) \tag{14}$$

Rule 3: Let $(i-1)$, i, and $(i+1)$ be three consecutive stages of a CA. If all or none of the three stages are selected, then the row or column in T_u that corresponds to stage i will be all 0's as shown in Eq. (14).

The necessity of the above construction is now proven by contradiction. Consider n pairs of a CA states,

$$(x_1, x_2), (x_3, x_4), \ldots, (x_{2i-3}, x_{2i-2}), (x_{2i-1}, x_{2i}), \ldots, (x_{2n-1}, x_{2n}) \tag{15}$$

Since half of the $2n$ stages must be tapped, if there is an (odd, even) pair with both stages tapped, then there exists another pair with both stages untapped. Therefore, if the theorem is false, it is always possible to find a pair i with both stages tapped (untapped) and every pair either to the left or right of the i-th pair has one and only one tapped stage.

. **Case 1: Both stages in the i-th pair are tapped** Without loss of generality, assume every pair to the left of the i-th pair has one and only one tapped stage, then x_{2i-3} in the $(i-1)$-th pair must be selected. Otherwise, three consecutive stages are selected and Rule 3 is violated. If x_{2i-4} is selected in the $(i-2)$-th pair, then rows in T_u that corresponds to x_{2i-1} and x_{2i-3} are equivalent. The dependency in rows of T_u occurs whenever two consecutive stages are selected in the first $2i-2$ stages if both x_{2i-1} and x_{2i} are chosen. Hence, choosing both stages in the i-th pair requires choosing all odd stages in the first $i-1$ pairs. From the

transition matrix T

$$T = \begin{pmatrix} c_1 & 1 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & c_2 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \\ 0 & 0 & \cdots & c_{2i-5} & 1 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \cdots & 1 & c_{2i-4} & 1 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \cdots & 0 & 1 & c_{2i-3} & 1 & 0 & 0 & \cdots \\ 0 & 0 & \cdots & 0 & 0 & 1 & c_{2i-2} & 1 & 0 & \cdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & 1 & c_{2i-1} & 1 & \cdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 1 & c_{2i} & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \quad (16)$$

the $n \times n$ submatrix T_u is derived

$$T_u = \begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & 1 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \cdots & 1 & 0 & 0 & 0 & \cdots \\ \cdots & \cdots & \cdots & 1 & 1 & 0 & 0 & \cdots \\ \cdots & \cdots & \cdots & 0 & 1 & 1 & 0 & \cdots \\ \cdots & \cdots & \cdots & 0 & 0 & 1 & 0 & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \quad (17)$$

The first i rows of T_u sum up to 0. Therefore, selecting both stages of the i-th pair is illegal.

Similarly, choosing none of the stages in the i-th pair requires choosing even stages in the first $i-1$ pairs, in which case the first i columns of T_u sum up to 0. Selecting none of the two stages in the i-th pair also leads to contradiction. Hence, only one of the two stages in the i-th pair must be included in the selections.

To prove that the conditions of the construction are sufficient to ensure complete transition count, consider the i-th pair (x_{2i-1}, x_{2i}) in Eq. (15). Define

$$s_i = \begin{cases} 0 & \text{if stage } 2i-1 \text{ is selected} \\ 1 & \text{if stage } 2i \text{ is selected} \end{cases} \quad (18)$$

It can be shown that the matrix T_u has the following form

$$T_u = \begin{pmatrix} 1 & s_1s_2 & 0 & 0 & 0 & 0 & \cdots & \cdots & \cdots \\ \overline{s_1}\,\overline{s_2} & 1 & s_2s_3 & 0 & 0 & 0 & \cdots & \cdots & \cdots \\ 0 & \overline{s_2}\,\overline{s_3} & 1 & s_3s_4 & 0 & 0 & \cdots & \cdots & \cdots \\ 0 & 0 & \overline{s_3}\,\overline{s_4} & 1 & s_4s_5 & 0 & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \overline{s_3}\,\overline{s_4} & 1 & s_4s_5 & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & \overline{s_4}\,\overline{s_5} & 1 & \cdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 1 & s_{n-1}s_n \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \overline{s_{n-1}}\,\overline{s_n} & 1 \end{pmatrix} \qquad (19)$$

The entries $g_{(i-1)i}$ and $g_{i(i-1)}$ cannot both be 1 for any selection outlined in the Theorem. If the entry $g_{i(i-1)}$ is 1, then row $i-1$ is added to row i. By doing series of fundamental operation on T_u starting from row 1, it is always possible to change T_u into an upper triangular matrix with unity diagonal elements. The rank of T_u is n. Hence, the tap selection according to the theorem guarantees that exhaustive two-pattern tests are generated for a CA. **Q.E.D.**

Corollary 2 *The* $2n$ *stage CA can be connected to the* n *inputs of a CUT in* 2^n *possible ways.*

Proof: Since there are two choices for each of the n pairs in a $2n$ stage CA, the number of possible connection to achieve exhaustive two-pattern tests is 2^n. **Q.E.D.**

Note that the above result is independent of the exact rules used in the CA. However, if the CA generates maximal sequence, then single seeding is required for exhaustive testing. Tables of maximal length 90/150 CA can be obtained from [SS90].

The number of ways in which n-out-of-$2n$ CA outputs can be selected is in agreement with observation made in [FM91]. Note that a CA provides many more ways (2^n) of selecting n-out-of-$2n$ stages compared to an LFSR. This makes CA more suitable TPGs for two pattern testing.

Example 3 *Consider a 10 stage CA (hybrid 90/150, null boundary condition) with the following configuration. Figure 2 shows two different ways of choosing* n*-out-of-*$2n$ *stages*

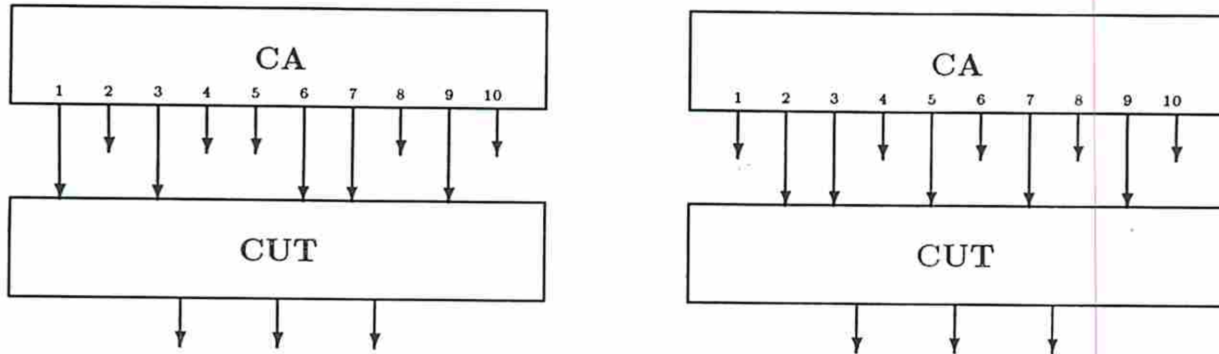

Figure 2: CA for 5 input CUT

for exhaustive testing. Note that there are 30 **other ways** *in which taps may be selected to achieve complete transition coverage.*

Table 1 shows some examples LFSRs and tap selections suitable for two-pattern exhaustive testing. In Table 1 (a), $P(x)$ is the feedback polynomial for the LFSR. Note that the all-odd and all-even taps guarantee complete transition coverage for both type 1 and type 2 LFSRs. On the other hand, there are some taps particular to the type 1 and type 2 LFSRs, if any even power of x (other than 1 and x^{2n}) is present in $P(x)$. In Table 1 (b), $P(x)$ is the characteristic polynomial of the CA. Note that there are a large number of choices in case of CAs.

4 General Tap Selection

Constraints on test application time or TPG hardware may force one to reduce the number of stages in the TPGs. This implies that reduction in test length (or in TPG hardware complexity) is achieved at the cost of reducing pattern coverage (transition count). If the CUT outputs do not depend on all the inputs, then pseudo-exhaustive testing may be used to reduce test time without reducing possible fault coverage. This is the subject of a companion paper [GC93]. However, if the CUT has even one output which depends on all the inputs, then pseudo-exhaustive testing cannot reduce the number of tests required. For such circuits, one has to trade-off transition count (and maybe fault-coverage) to reduce test

Table 1: Examples of Tap Selections for Achieving Complete Transition Coverage

Number of PI's	$P(x)$	Taps Selected
4	$1+x+x^4$	1,3 2,4
6	$1+x+x^6$	1,3,5 2,4,6
8	$1+x^2+x^7+x^8$	1,3,5,7 2,4,6,8 1,4,6,8 (type 1 only) 2,3,5,7 (type 2 only)

(a) Linear Feedback Shift-Registers (LFSRs)

Number of PI's	$P(x)$	Taps Selected
4	$1+x^3+x^4$	1,3 2,4 1,4 2,3
6	$1+x+x^6$	1,3,5 2,4,6 2,3,5 1,4,5 ⋮
8	$1+x+x^6+x^7+x^8$	1,3,5,7 2,4,6,8 2,3,6,7 1,4,6,8

(b) Cellular Automata (CA)

time. In the following, an algorithm shall be presented which will help achieve maximum pattern-pair coverage for a given test length. That is, given the number of stages in the TPG (which determines the test length as well), the following provides a way to connect the n CUT inputs to the TPG in manner that maximizes pattern coverage (transition count).

Lemma 3 *Given an m-stage ($n < m < 2n$) LFSR/CA and an n-input CUT. The maximum achievable transition count is $2^m - 1$.*

Proof: For an m-stage linear sequential circuit (of which LFSR/CA is an example), the maximum sequence length is $2^m - 1$. Hence, the maximum achievable transition count is $2^m - 1$.
Q.E.D.

As has been discussed above, maximal length LFSR/CA are useful because they need only one seed. Also, it has been shown above that LFSRs with feedback polynomials which contain many even powers of x are more suitable for two pattern testing. CAs are, in general, better suited for two pattern testing, independent of the rules. Hence, the next question to be answered for TPG design, for two pattern testing, is to determine which of its outputs should be connected to the CUT inputs. *This problem of choosing appropriate TPG stage outputs to connect to CUT inputs is called tap selection problem.*

Theorem 3 *Consider an n-input CUT and an m-stage linear TPG, where $n < m < 2n$. The following tap selection guarantees that the maximum achievable transition count given by Lemma 3 is achieved.*

1. *Select $\lceil \frac{m}{2} \rceil$ taps using an algorithm which generates optimal transition count for $\lceil \frac{m}{2} \rceil$ input CUT.*
2. *Select remaining $n - \lceil \frac{m}{2} \rceil$ taps arbitrarily.*

Proof: Let $m' = \lceil \frac{m}{2} \rceil$. Since the m' selected taps in Step 1 have optimal transition count, the submatrix T_u for the tapped stages in Eq. (4) must be of full rank. If m is even, the $m' \times m'$ submatrix T_u must have rank $r = m'$. By arbitrarily selecting the next tap, the size

and the rank of T_u become $(m'+1) \times (m'-1)$ and $m'-1$, respectively. After increasing the number of taps to n, the size and rank of T_u become $n \times (m-n)$ and $n-m$, respectively. There are a total of 2^n states, each with 2^{m-n} transitions (except the all-zero state, which has only $2^{m-n}-1$ transitions). The transition count is $2^m - 1$, which is optimal for an m-stage linear TPG. If n is odd, the matrix T_u has size $m' \times (m'-1)$ and rank $m'-1$. By arbitrarily selecting the next tap, the size and rank of T_u become $(m'+1) \times (m'-2)$ and $m'-2$, respectively. After all n taps are selected, the size and the rank of T_u are $n \times (n-m)$ and $n-m$, respectively. Again, the transition count is $2^m - 1$, which is optimal. **Q.E.D.**

Note that the above theorem is applicable to any linear TPG not just LFSR and CA. The main difficulty in using the above theorem to design a general linear TPG for maximal transition count is in the Step 1 of the construction outlined above. That is, a method for selecting $\lceil \frac{m}{2} \rceil$ out of m taps for maximum transition count (if CUT had $\lceil \frac{m}{2} \rceil$ inputs) is required. If m is even, then the results presented in Theorems 1 and 2 can be used to select $\lceil \frac{m}{2} \rceil = m/2$ taps to achieve maximum transition count. The remaining $n - m/2$, taps can then be randomly selected, as shown in Theorem 3 above.

If m is odd, then the selection of the initial $\lceil \frac{m}{2} \rceil$ taps for maximum transition count may be performed as follows. In case of type 2 LFSR, Rule 1 or Rule 2b in Theorem 1 can be used to select $\lfloor \frac{m}{2} \rfloor$ taps. The last tap (stage m of the LFSR), can then be selected. These $\lceil \frac{m}{2} \rceil$ taps can be shown to achieve maximum transition coverage. Similarly, for CA, it can be shown that $\lfloor \frac{m}{2} \rfloor$ taps can be selected from the first $m-1$ stages of CA using rules presented in Theorem 2. The last tap (stage m of the CA), can then be selected. These $\lceil \frac{m}{2} \rceil$ taps can be shown to achieve maximum transition coverage. A number of other ways of selecting $\lceil \frac{m}{2} \rceil$ out of m (m odd) taps have been found and are currently being investigated.

Example 4 *Consider a 5 input CUT and an 8 stage LFSR. The tap selection in Figure 3 achieves the maximum transition count* $2^8 - 1$. *Note that the odd stage outputs are selected first. Then the output of stage-8 is selected randomly (according to the the above Theorem) to connect to the CUT inputs.*

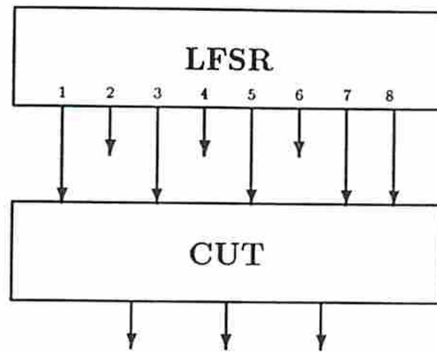

Figure 3: 8-stage LFSR for 5 input CUT

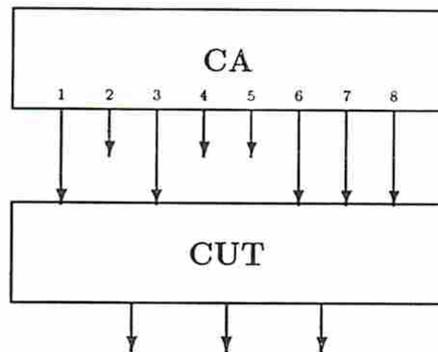

Figure 4: 8-stage CA for 5 input CUT

Example 5 *Consider a 5 input CUT and an 8 stage CA. The tap selection in Figure 4 achieves the maximum transition count* $2^8 - 1$. *In this case, either* $(1,3,6,7)$ *or* $(1,3,6,8)$ *can be viewed as initial tap selection for complete transition count. The last tap is then selected randomly.*

The results presented above provide a way of determining the best ways of connecting the n-inputs of the CUT to the m-outputs of an m-stage LFSR/CA ($n \leq m \leq 2n$). For simplicity, let us assume m to be even in the following discussion.

LFSR: Theorems 1 and 3 provide us with the following strategy for selecting n-out-of-m taps.

1. Select $m/2$ taps out of m using one of the rules provided in Theorem 1.
2. If $n > m/2$, then select the remaining $n - m/2$ taps arbitrarily.

This construction guarantees maximal transition coverage $(2^m - 1)$.

For an XLFSR, there are m ways of selecting n contiguous outputs of the XLFSR to connect the CUT inputs. It can be shown that the above construction generates a large number of tap selections which maximize transition count. The number of ways of selecting n-out-of-m taps by the above procedure has the lower bound ($n \leq m \leq 2n$, m even)

$$2\binom{m/2}{n - m/2}.$$

The number of distinct tap selections identified by the above procedure is upper bounded by

$$c\binom{m/2}{n - m/2}$$

where $c = \sum_{i=0}^{m} c_{2i}$ is the number of non-zero even coefficients of the feedback polynomial of the TPG. For most values of $m > n$, the number of choices identified by the above procedure are much larger than the number of choices offered by the XLFSR.

CA: Theorems 2 and 3 provide the following tap selection procedure.

1. Group consecutive (odd, even) stages of the CA, and select one stage from each (odd, even) stage pair.
2. If $(n > m/2)$ select the remaining taps randomly.

It can be shown that, by the above construction, there are

$$2^{m-n}\binom{m/2}{n-m/2}$$

different ways of selecting n-out-of-m stage outputs of the CA ($n \leq m \leq 2n$, m even).

Hence, it can be seen that the results presented here identify a very large number of ways to select n-out-of-m LFSR/CA stages to connect to n CUT inputs.

5 Experimental Results

The results presented above can be used to design TPGs which maximize transition count for given hardware constraints. If $2n$-stage LFSR/CA is available then the TPGs can be designed to ensure complete transition count for an n input CUT. This guarantees detection of 100% of all two pattern detectable faults.

However, if the number of stages in the TPG, $n \leq m \leq 2n$, then n TPG stage outputs can be selected to connect to the n CUT inputs, to maximize transition count. This construction allows a BIST designer to design the TPG to maximize transition count while meeting the hardware and test sequence length constraints. Since maximizing transition count means maximizing pattern-pair coverage, the TPG designed by using the above results should maximize coverage of faults detectable by pattern pairs. (Note that a TPG which does not provide maximum transition count, will have transition count which is no more than half of the maximum. This substantial reduction in transition count will almost certainly reduce the fault coverage.) A number of PLA benchmarks [BHMSV84] were used in the following experiments. These were synthesized such that 100% of the path delay faults in these circuits were robustly testable [PR91]. Since, the main emphasis of this paper is on exhaustive testing, selected output cones of these circuits were studied one at a time. The outputs which depended on 6-8 inputs and had a large number of paths were selected. This permitted extensive experimentations with a large number of TPG tap selections. Table 2 summarizes some details of the circuit outputs studied.

Table 2: Example Circuits Used to Study TPGs

Circuit Name	Number of Inputs	Number of Outputs	Cone Used in Experiment	No. of I/ps in Cone	No. of Physical Paths
add6	12	7	3	6	68
adr4	8	5	2	8	184
alu2	10	8	3	8	37
alu3	10	8	3	8	39
myadr4_1	8	1	1	8	64
radd	8	5	4	8	184
z4	7	4	1	7	56

LFSRs with primitive feedback polynomials were used in the experiments. For an n-input circuit, LFSRs with, $m \in \{2n, 2n-2, \ldots\}$ stages were used. For each value of m, n out of m TPG taps were selected in many different ways. Firstly, the taps were selected such that they satisfied the conditions derived in the paper to maximize transition count. Taps were also selected in alternate ways which did not maximize transition count. It was observed that (see Tables 3 and 4) the TPGs whose taps were selected by using the results presented (the boldface entries in these tables) produced sequences with maximum transition counts. The vectors generated by all the TPGs were then used to perform robust path delay fault simulation on the circuits. It was found that the TPGs designed using the results derived provided substantially higher coverage of path delay faults for all circuits and TPG sizes (m). Tables 3 and 4 show some representative results. (Note that for each physical path there are two logical paths, rising and falling.) As mentioned above, a tap selection which does not provide maximum transition count, has transition count at most half of the maximum. Hence, in most cases, there is a substantial drop in fault coverage if the tap selection does not guarantee maximum transition count. However, it can also be seen that, in many cases, different tap selections with the same transition count have very different fault coverages. There is a need to study this problem further and incorporate circuit specific information into the TPG design process.

Table 3: Experimental Data on Different TPGs for *add6* (op no. 3)

m # of Stages in LFSR	Taps Selected	Transition Count	# of Logical Paths Tested
12	**(1,3,5,7,9,11)**	**4095**	**136**
	(2,4,6,8,10,12)	**4095**	**136**
	(1,2,5,7,9,11)	2048	94
	(1,3,6,7,9,11)	2048	84
	(1,3,5,7,10,11)	2048	87
	(1,2,6,7,9,11)	1024	57
	(1,2,5,7,10,11)	1024	63
	(1,3,6,7,10,11)	1024	53
	(1,4,5,6,7,10)	512	38
	(1,2,3,4,7,8)	256	21
	(1,2,6,7,8,9)	256	26
10	**(1,3,5,7,9,10)**	**1023**	**59**
	(2,4,6,8,9,10)	**1023**	**57**
	(1,2,3,5,6,7)	256	26
	(1,2,4,5,6,7)	256	26
	(1,2,3,4,5,6)	128	15

Table 4: Experimental Data on Different TPGs for *adr4* (op no. 2)

m # of Stages in LFSR	Taps Selected	Transition Count	# of Logical Paths Tested
16	**(1,3,5,7,9,11,13,15)**	**65535**	**368**
	(2,4,6,8,10,12,14,16)	**65535**	**368**
	(1,4,5,7,9,11,13,15)	32768	270
	(1,3,5,7,10,11,13,15)	32768	200
	(1,2,5,7,10,11,13,15)	16384	108
	(1,3,4,7,9,11,12,15)	16384	188
14	**(1,3,5,7,9,10,11,13)**	**16383**	**151**
	(2,4,6,8,10,11,12,14)	**16383**	**151**
	(1,2,3,5,8,9,11,13)	8192	83
	(1,2,5,6,7,10,11,13)	4096	94
	(1,3,4,5,8,10,11,12)	4096	128
	(1,2,3,8,9,10,11,12)	1024	35
	(1,6,7,8,9,10,11,12)	1024	56
12	**(1,2,3,5,7,9,10,11)**	**4095**	**92**
	(2,4,5,6,8,10,11,12)	**4095**	**128**
	(1,2,3,5,6,7,9,11)	**4095**	**63**
	(1,2,3,6,7,8,9,10)	1024	35
	(2,3,4,5,6,9,10,11)	1024	60

6 Conclusion

Design of LFSR/CA test pattern generators, suitable for two pattern exhaustive testing, is the main subject of this paper. Conditions for TPG tap selection which guarantee maximal transition count have been derived. LFSRs with primitive feedback polynomials, which have many even powers of x, have been shown to be more suitable as TPGs for two pattern testing. Cellular automata make good TPGs for two pattern testing, independent of their feedback rules.

A large class of TPGs that maximize transition count have been identified. Experimental results on some benchmark circuits show that for a given constraint on the TPG size, the TPGs designed by using the results derived in this paper, generate test patterns which provide much higher fault coverages than other TPGs.

A number of issues still remain unaddressed. Firstly, more real circuits have multiple outputs and, in many cases, none of the outputs depend on all the circuit inputs. In such cases, two pattern pseudo-exhaustive testing concepts can help reduce the test sequence length and TPG hardware complexity, without reducing fault coverage. This is the subject of a companion paper [GC93]. Also, the focus of this work has been on designing TPGs to maximize transition count or pattern coverage. However, there is a need to study test application methodologies essential for detection for some of the two pattern testable faults (e.g. robust path delay tests require that test patterns be applied using a combination of slow and fast clocks). These and other such practical issues pertaining to test application need to be studied. Since, the test sequence lengths for two pattern testing are high, there is a need to study efficient pseudo-random test techniques for two pattern testing. This is a subject of ongoing research.

Acknowledgement

The authors wish to thank Dr. A.K. Pramanick (IBM) and Prof. S.M. Reddy (University of Iowa) for providing the path delay fault simulator.

References

[Bar90] Paul H. Bardell. Analysis of Cellular Automata as Pseudorandom Pattern Generators. In *Proceedings IEEE International Test Conference*, pages 762–767, 1990.

[BHMSV84] R. K. Brayton, G. D. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli. *Logic Minimization Algorithms for VLSI Synthesis*. Kluwer Academic Publishers, Boston, MA, 1984.

[CK85] Gary L. Craig and Charles R. Kime. Pseudo-Exhaustive Adjacency Testing: A BIST Approach for Stuck-Open Faults. In *Proceedings IEEE International Test Conference*, pages 126–137, 1985.

[FM91] Kiyoshi Furuya and Edward J. McCluskey. Two-Pattern Test Capabilities of Autonomous TPG Circuits. In *Proceedings IEEE International Test Conference*, pages 704–711, October 1991.

[GC93] Sandeep K. Gupta and Chih-Ang Chen. BIST Test Pattern Generators for Two Pattern Pseudo-Exhaustive Testing. *Submitted to the International Test Conference*, 1993.

[LC83] S. Lin and D. J. Costello. *Error Control Coding: Fundamentals and Applications*. Prentice Hall, Englewood Cliffs, N.J., 1983.

[PR91] A. K. Pramanick and S. M. Reddy. On Multiple Path Propagating Tests for Path Delay Faults. In *Proceedings IEEE International Test Conference*, pages 393–402, 1991.

[SB91] Jacob Savir and Robert Berry. At-Speed Test Is Not Necessarily an AC Test. In *Proceedings IEEE International Test Conference*, pages 722–728, 1991.

[Smi85] Gordon L. Smith. Model for Delay Faults Based on Paths. In *Proceedings IEEE International Test Conference*, pages 342–349, 1985.

[SS90] T. Slater and M. Serra. Tables of Linear Hybrid 90/150 Cellular Automata. Technical Report DCS-105-IR, Department of Computer Science, University of Vicoria, Victoria BC, Canada, 1990.

[SSMM90] M. Serra, T. Slater, J. C. Muzio, and D. M. Miller. The Analysis of One-Dimensional Cellular Automata and Their Aliasing Properties. *IEEE Trans. on CAD*, 9(7):767–778, July 1990.

[Sta84] C. W. Starke. Built-In Test for CMOS Circuit. In *Proceedings IEEE International Test Conference*, pages 309–314, 1984.

[ZBM92] S. Zhang, R. Byrne, and D. M. Miller. BIST Generators for Sequential Faults. In *Proceedings IEEE International Conference on Computer Design*, pages 260–263, 1992.