Two-Dimensional Linear Hybrid Cellular Automata as Test Pattern Generators in VLSI Testing

by

Jie Xia Zhu (Cathy)
B.EE, Jinan University, China, 1995

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

We accept this thesis as conforming
to the required standard

[redacted]

Dr. Jon C. Muzio, Supervisor
(Department of Computer Science)

[redacted]

Dr. Micaela Serra, Departmental Member
(Department of Computer Science)

[redacted]

Dr. Kin F. Li, Outside Member
(Department of Electrical & Computer Engineering)

[redacted]

Dr. Vijay K. Bhargava, External Examiner
(Department of Electrical & Computer Engineering)

Supervisor: Dr. Jon C. Muzio

ABSTRACT

This thesis discusses two-dimensional linear hybrid cellular automata (2D LHCA) and their use as test pattern generators in VLSI testing. It gives the transition matrix of the n-by-m LHCA. As it is desirable for the characteristic polynomial of the transition matrix to be primitive since it generates all possible non-zero test vectors, starting from any non-zero state, two ways to speed up the calculation of the characteristic polynomial of 3-by-m LHCA are developed. And a table of minimal-cost n-by-m LHCA with maximum length cycle for small values of n and m is listed followed by some analysis.

Transition properties of LHCA are important for detecting delay faults because the latter require a pair of vectors to stimulate. The thesis concentrates on analyzing 2-by-m LHCA. We discover an error in a published theorem, formulate and prove a correct result, allowing for the accurate calculation of the number of substate vectors which produced maximum numbers of the transition pairs for 2-by-m LHCA. Results show that 2-by-m two-dimensional LHCA have a larger set of transitions than one-dimensional LHCA. When applying 2-by-m LHCA as test pattern generators in testing experiments, we demonstrate that 2-by-m LHCA perform slightly better than one-dimensional LHCA for the detection of delay faults in the ISCAS85 benchmark circuits.

Examiners:

[redacted]

Dr. Jon C. Muzio, Supervisor
(Department of Computer Science)

[redacted]

Dr. Micaela Serra, Departmental Member
(Department of Computer Science)

[redacted]

Dr. Kin F. Li, Outside Member
(Department of Electrical & Computer Engineering)

[redacted]

Dr. Vijay K. Bhargava, External Examiner
(Department of Electrical & Computer Engineering)

Table of Contents

List of Tables

List of Figures

List of Abbreviations

LHCA or CA	Linear Hybrid Cellular Automata
LFSM	Linear Finite State Machine
LFSR	Linear Feedback Shift Register
CP	Characteristic Polynomial
TPG	Test Pattern Generator
PPSFP	Parallel Pattern Single Fault Propagation
VLSI	Very Large Scale Integration
BIST	Built-In Self-Test
DFT	Design For Test

Acknowledgement

I would like to express my greatest appreciation to my supervisor, Dr. Jon C. Muzio, for his encouragement, support, guidance throughout my graduate studies and thesis work.

I also would like to thank my parents, all my family members for their consistent support, understanding and immense love to me.

Thanks to professors Dr. Michael Miller, Dr. Micaela Serra, Dr. Kin Li and all of the VLSI group members for their advice and the talks during my graduate studies.

Finally, thanks to all my friends both in China and here for their invaluable friendship and concern for me.

Dedication

To my family and my friends.

Chapter 1

Introduction

Linear Hybrid Cellular Automata (LHCA or CA) have been studied in the past few years. They are a type of linear finite state machines (LFSMs), composed of an array of n cells. Each cell consists of a single memory element capable of storing a state and being updated synchronously by its next state function. The function is called the cell's rule. LHCA have been considered as an alternative to linear feedback shift registers (LFSRs) in VLSI design and testing, and in particular for Built-In Self-Test (BIST). The applications include test pattern generation, pesudorandom number generation, cryptography, and signature analysis.

A considerable amount of work has been done to investigate the properties of CA. When used as a test vector generator, an n-degree LHCA with a primitive characteristic polynomial over $GF(2)$ is desired since it yields a maximum length cycle, which means it traverses all possible $(2^n - 1)$ non-zero states before returning to its initial non-zero state. Based on maximum length CA, their pseudorandom properties and transition properties have been studied.

Two-dimensional linear hybrid cellular automata (2D LHCA) and their properties have first been studied by Hassan Janoowalla [12]. The results show that 1D CA and 2D CA perform equally well and much better than LFSRs regarding pseudorandom test and fault simulation. Cattell et al. [4] present 2-by-m LHCA in detail their two vector transition properties and undertake some application experiments. The results also show that 2D LHCA, on average, perform better than 1D LHCA for testing delay faults.

The purpose of this thesis is to present some properties of more complex 2D LHCA, especially their transition properties. Based on the theoretical analysis, their applications in VLSI testing to detect faults are discussed. The thesis is mainly divided into two parts. The first part focuses on a theoretical analysis of more complicated 2D LHCA with regular configuration. They are n-by-m ($n, m \geq 3$) LHCA. We investigate their transition function and characteristic polynomial (CP). When working on developing the transition properties of 2D LHCA, we discover that a theorem about calculating the number of substate vectors for 2-by-m LHCA is incorrect in [4]. In the second part of the thesis, a careful analysis of

the theorem followed by the derivation of a new theorem is presented. We then compare the transition properties of 2-by-m LHCA with 1D LHCA. The results show that 2-by-m 2D LHCA have a lot more transitions than 1D LHCA particularly as the number of cells increases. After that, we undertake some initial simulation studies using a set of standard benchmark circuits, which confirm the results of the theoretical analysis, that 2-by-m 2D LHCA perform slightly better than 1D LHCA for the detection of delay faults.

The thesis is organized as follows.

Chapter 2 provides the necessary theoretical background for the thesis. This includes the definitions of finite fields and linear finite state machines (LFSMs). Two typical LFSMs, linear feedback shift register (LFSR) and linear hybrid cellular automaton (LHCA), are introduced followed by the difference of their transition properties. Based on the existing LHCA, which is 1D LHCA, another class of LHCA, 2D LHCA is presented briefly and some previous work relating to the applications and the theory of CA are discussed.

Chapter 3 investigates n-by-m 2D LHCA. We deduce the transition matrix structure of n-by-m LHCA at the beginning. As it is important to evaluate the characteristic polynomial (CP) of a LHCA, we introduce two ways to speed up the calculation of the CP because of the huge matrix structure. This is based on 3-by-m LHCA. Finally, a table of minimal-cost n-by-m LHCA with maximum length cycle for small values of n and m is listed.

Chapter 4 focuses on the transition properties of 2D LHCA in the 2-by-m case. Because some previous work is found to be inaccurate, we derive several new theorems followed by some examples to help explain the theorems. We also discuss the idea of "partner sets" which help calculate the number of substate vectors from which we generate upper bound of distinct transitions. Some conclusions are given at the end with some discussion of the results compared with 1D LHCA.

Chapter 5 describes testing application, built-in self-test, especially in one critical area - test pattern generator (TPG). Then we introduce the delay fault model in combinational circuits. In fault simulation, fault equivalence rules for delay faults are presented followed by

an efficient way to simulate delay faults, which is a parallel pattern single fault propagation (PPSFP) [18].

Chapter 6 shows the simulation experiments on the ISCAS85 benchmark circuits using 2-by-m 2D LHCA. The experiments are developed based on some previous work done by the VLSI group at the University of Victoria. We illustrate two experiments and observe the results of fault coverage regarding delay faults compared with 1D LHCA. As we anticipated, 2D LHCA perform better than 1D LHCA.

Chapter 7 summarizes the conclusions of the thesis and raises several relevant thoughts as future work.

There are four appendices included in the thesis. Appendix 1 shows the Maple program for generating the Characteristic polynomial of a particular 3-by-m LHCA using its recurrence relation. Appendix 2 shows the Maple program for generating minimal-cost maximum-length cycle 2D LHCA. These two programs are written by me. Appendix 3 shows all the experimental results and appendix 4 contains the user manual for fault simulation software 'sim3' that we use in the thesis.

Chapter 2

Background

This chapter presents the necessary background material for the thesis. Finite Fields are introduced first and followed by several kinds of linear finite state machines (LFSM). Previous work relating to the theory are discussed.

2.1 Finite Fields

A field, in abstract algebra, is an algebraic system of elements in which the operations of addition and multiplication may be performed and the associative, commutative, and distributive rules hold, which are familiar from the arithmetic of ordinary numbers. Fields are important objects of study in algebra since they provide the proper generalization of number domains, such as the sets of rational numbers or real numbers. Fields used to be called rational domains. [20]

A finite field, also called a Galois field, is a field with a finite number of elements. All finite fields have prime characteristic, which means the number of elements in the field is always a prime or a power of a prime. It is usually denoted as GF(q).

The finite field with two elements, GF(2), is of particular importance. GF(2) consists of the two elements 0 and 1, which satisfy the following addition and multiplication table shown in 2.1. In this thesis, all arithmetic is performed over GF(2) and the LFSMs discussed are all defined over GF(2).

Table 2.1. *$GF(2)$ Arithmetic.*

+	0	1
0	0	1
1	1	0

×	0	1
0	0	0
1	0	1

2.2 Linear Finite State Machines

A finite state machine(FSM) is one of the most important kinds of sequential circuit. The sequential logic can be implemented within only a fixed number of possible states.

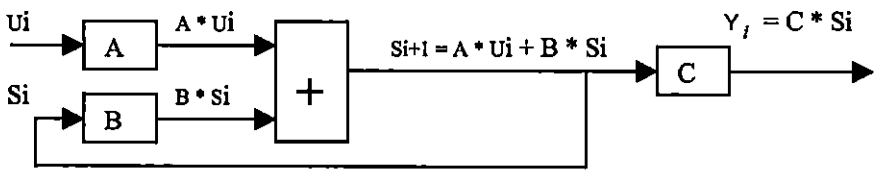


Figure 2.1. *A Finite State Machine.*

Consider the machine M in figure 2.1. Let s_i be the state vector, u_i be the input vector and y_i be the output vector at time i, and assume the dimension of the vectors be n, p, r respectively. The next state and output functions are of the forms

$$s_{i+1} = f(s_i, u_i) = A \cdot u_i + B \cdot s_i$$
$$y_i = \delta(s_i) = C \cdot s_i$$

where A, B and C are matrices over GF(2); s_i, u_i, and y_i are column vectors; and the matrix operations are the usual matrix operations with the arithmetic performed in GF(2). The dimensions of the matrices A, B and C are $n \times n, n \times p$ and $r \times n$ respectively and consistent with the dimensions of the three vector spaces.

We say this machine is a linear finite state machine(LFSM) [15].

For a machine to be linear means that f is a linear function from n-bit vectors to n-bit vectors; that is,

$$f(a + b) = f(a) + f(b)$$

for any states a and b.

If a machine has no input, the LFSM described above is "autonomous". Then its next state function is

$$s_{i+1} = B \cdot s_i.$$

All the LFSMs discussed in this thesis are autonomous. They are initialized with a start state, and then proceed according to their next-state function. Autonomous machines can be used as test pattern generators or pseudorandom number generators. Two common LFSMs are introduced in the following section.

2.2.1 Linear Feedback Shift Register

A Linear Feedback Shift Register (LFSR) is an LFSM defined as a collection of storage elements and XOR gates which perform addition over GF(2), chained together and controlled by a synchronous clock.

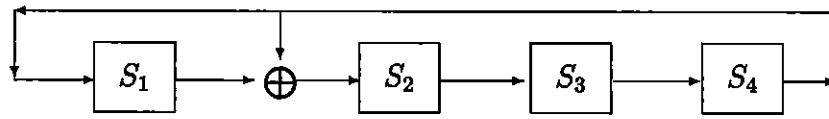


Figure 2.2. *A Linear Feedback Shift Register.*

Example 2.1. Figure 2.2 is an example of LFSR, which has an exclusive-OR gate between two cells. Each cell is a D flip-flop connected in a shift-register structure. A feedback line from the rightmost element can be "tapped" back into the EXOR gate. A set of next-state transition equations for the LFSR above can be written as follows:

$$s_1^+ = s_4$$
$$s_2^+ = s_1 + s_4$$
$$s_3^+ = s_2$$
$$s_4^+ = s_3$$

With all operations being carried out over GF(2), we can also present the next state function as:

$$[s_1^+,\ s_2^+,\ s_3^+,\ s_4^+]^T = T \cdot [s_1,\ s_2,\ s_3,\ s_4]^T$$

where T is a transition matrix for the LFSR.

$$T = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

We define the characteristic polynomial(CP) of a matrix as

$$\Delta = |xI + T|$$

So the CP of the above transition matrix is

$$\Delta(x) = x^4 + x + 1$$

If we start from a non-zero state at time 0, it is easy to obtain a cycle structure of the LFSR. Table 2.2 shows all states, each state denoted as $s_1\ s_2\ s_3\ s_4$.

time	state	time	state	time	state	time	state
0	0001	4	1101	8	0111	12	1000
1	1100	5	1010	9	1111	13	0100
2	0110	6	0101	10	1011	14	0010
3	0011	7	1110	11	1001	15	0001

Table 2.2. *States for LFSR.*

In this example, we find that 2^4 -1 = 15 non-zero states are generated within the same cycle, called a maximum length cycle. The characteristic polynomial associated with a maximum length LFSR is "primitive" [2]. Sequences generated by an LFSR are called pseudorandom sequences and the length grows exponentially with the number of stages in the register. LFSR can be implemented as a pseudorandom number generator in VLSI testing, which we are going to discuss in chapter 5.

2.2.2 Linear Hybrid Cellular Automaton

Besides LFSR, Linear Hybrid Cellular Automaton (LHCA) is also an LFSM composed of one-dimensional array of cells. They have been proposed as an alternative to LFSR, in applications such as pseudorandom number generators in recent years since their patterns have better transition properties.

The characteristics of LHCA is that its cells receive input solely from their nearest neighbors. The configuration of each cell is defined by a rule which defines the next-state function of the cell. Since each cell depends on itself and its two neighbors, the rule is a boolean function of 3 variables, which means there are 2^{2^3} or 256 possible rules. As each cell can obey different rules, LHCA is said to be "hybrid". It has been proved that only LHCA using rules 90 and 150 has maximum length cycle.

Example 2.2. Figure 2.3 is an example of an LHCA. Each cell, labelled s_i, $1 \leq i \leq n$, can hold either rule 90 or 150, and at every clock cycle, it receives an input from its nearest neighbors, s_{i-1} and s_{i+1}. The cells at the boundary of the array always receive a 0.

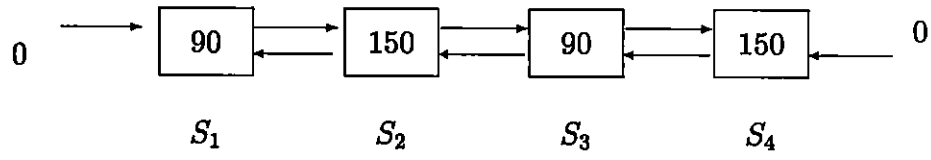


Figure 2.3. *A Linear Hybrid Cellular Automaton.*

The computation rules 90 and 150 are defined as follows:

$$\text{Rule 90: } s_i^+ = s_{i-1} + s_{i+1}$$

$$\text{Rule 150: } s_i^+ = s_{i-1} + s_i + s_{i+1}$$

We can also write the set of next-state equations of figure 2.3 as below:

$$s_1^+ = s_2$$

$$s_2^+ = s_1 + s_2 + s_3$$

$$s_3^+ = s_2 + s_4$$

$$s_4^+ = s_3 + s_4$$

And the corresponding matrix is

$$T = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

The CP of the matrix is $\Delta(x) = x^4 + x + 1$. We find that it is the same as that of the LFSR described in the previous section.

Generating its cycle structure using the LHCA in figure 2.3, we obtain table 2.3. It is easy to see that this is also a maximum length cycle as it goes through all 15 non-zero states. Comparing it with the cycle structure in LFSR, the vectors occur in a different order.

time	state	time	state	time	state	time	state
0	0001	4	0010	8	0111	12	1111
1	0011	5	0101	9	1000	13	1100
2	0110	6	1101	10	0100	14	1010
3	1011	7	1001	11	1110	15	0001

Table 2.3. *States for LHCA.*

2.2.3 Transition Properties

In table 2.2 and 2.3, each state is denoted as s_1, s_2, s_3, s_4. We count the number of transitions for $\langle (s_1, s_2, s_3, s_4), (s_1^+, s_2^+, s_3^+, s_4^+) \rangle$. Obviously, the number of transitions is $2^4 - 1 = 15$ because they are both maximum length cycle. But if we only select 2-cell vectors, for example, $\langle (s_2, s_3), (s_2^+, s_3^+) \rangle$, the number of transitions are different. For the LFSR in figure 2.2, we have the following transitions:

$$\langle (00), (10) \rangle, \quad \langle (10), (11) \rangle, \quad \langle (11), (01) \rangle, \quad \langle (01), (10) \rangle,$$
$$\langle (10), (01) \rangle, \quad \langle (11), (11) \rangle, \quad \langle (01), (00) \rangle, \quad \langle (00), (00) \rangle,$$

the total number is 8. While for the LHCA in figure 2.3, we have the transitions as follows:

$$\begin{array}{llll} \langle(00),(01)\rangle, & \langle(01),(11)\rangle, & \langle(11),(01)\rangle, & \langle(01),(01)\rangle, \\ \langle(01),(10)\rangle, & \langle(10),(10)\rangle, & \langle(10),(00)\rangle, & \langle(00),(11)\rangle, \\ \langle(00),(01)\rangle, & \langle(01),(11)\rangle, & \langle(10),(11)\rangle, & \langle(11),(11)\rangle, \\ \langle(11),(10)\rangle, & \langle(10),(01)\rangle, & \langle(01),(00)\rangle, & \end{array}$$

the number is 15. This is an illustration that LHCA have a larger number of transitions than LFSRs.

2.2.4 Two Dimensional LHCA

Two-dimensional LHCA have been studied in recent years. A particular structure discussed in [4] is based on a 2-by-n array of cells. Each cell is connected to all of its nearest neighbors, which means that there are three such neighbors for each cell.

Example 2.3. An example of a 2-by-3 LHCA is shown in figure 2.4. Here we define that there are two kinds of cell rules which are generalizations of the rules used in section 2.2.2. One is rule-zero, which means that the cell only receives input from its nearest neighbors. The other one is rule-one meaning the cell receiving input from its nearest neighbors and itself as well. Now we can show the next-state equations, transition matrix and characteristic polynomial.

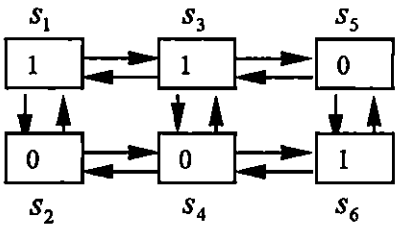


Figure 2.4. *A 2-by-3 LHCA example.*

The next-state equations are:

$$s_1^+ = s_1 + s_2 + s_3$$
$$s_2^+ = s_1 + s_4$$
$$s_3^+ = s_1 + s_3 + s_4 + s_5$$
$$s_4^+ = s_2 + s_3 + s_6$$
$$s_5^+ = s_3 + s_6$$
$$s_6^+ = s_4 + s_5 + s_6$$

The transition matrix is:

$$T = \left(\begin{array}{cc|cc|cc} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ \hline 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ \hline 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right)$$

The resulting CP is:

$$\Delta(x) = x^6 + x^5 + x^3 + x^2 + 1.$$

The transition properties of two-dimensional LHCA are discussed in chapter 4.

2.2.5 Previous Work

As we discussed in the previous sections, the better transition properties of LHCA as against LFSR have made it increasingly common in VLSI testing and other applications for their use as pseudorandom pattern generation. This section describes some previous work done relating to LHCA and their applications, most of which are VLSI circuit testing. Much work has been done about one-dimensional LHCA.

The resurgence of interest in CA started with the work of Wolfram. Aspects such as chaotic behavior, self-organization and computational complexity are explored in [21] and

[22]. In [23], Wolfram studies the randomness properties of CA. The CA considered by Wolfram are uniform and usually non-linear, and use various boundary configurations.

One of the early works by VLSI researchers is [14]. In this paper, Pries *et al.* study CA that are one-dimensional, linear and uniform. The authors study conditions under which the CA is reversible, Hortensius in [9] and Hortensius *et al.* in [10] and [11] compare CA with LFSRs for pseudorandom number generation and signature analysis. The authors determine that CA have less cross correlation in their output stream, and hence may be better as test pattern generators. Further comparative work in this vein, for hybrid CA based on rules 90 and 150, is in [16] and [17]. The authors argue that CA have better randomness properties and VLSI layout advantages when compared to LFSRs.

Many papers cited in this thesis were pursued by the VLSI group in the University of Victoria. A tutorial paper [5] explains the background for CA, following by describing a synthesis algorithm with low complexity, which solves the problem of finding a particular LHCA. LHCA have powerful concatenation and partitioning properties allowing several smaller maximal length LHCA to be combined into a much larger maximal length LHCA. Their performance in this regard is compared with LFSRs, which do not have quite as much flexibility. The basis for LHCA being better generators than LFSRs for testing delay faults is explained by showing the richer nature of the transition pairs generated by the LHCA.

In [3], Cattell and Muzio present theoretical aspects of LHCA over $GF(q)$ focusing the general results concerning the characteristic polynomials of such automata. A probabilistic synthesis algorithm for determining such a LHCA with a specific characteristic polynomial is given, along with empirical results and a theoretical analysis. Cyclic-boundary CA are defined and related to the more common null-boundary CA. An explicit similarity transform between a CA and its corresponding LFSR is derived.

Zhang *et al.* in [25] present a combinatorial method of evaluating the effectiveness of LHCA and LFSR as generators for stimulating faults requiring a pair of vectors. A theoretical analysis and empirical comparisons to see why the LHCA are better than the LFSRs as generators for sequential type faults in a built-in self-test environment are provided. Based

on the concept of a partner set, the method derives the number of distinct k-cell substate vectors which have $2^{2k}, 1 \leq k \leq \lfloor n/2 \rfloor$, transition capability for an n-cell LHCA and an n-cell LFSR with maximum length cycles. Simulation studies of the ISCAS85 benchmark circuits provide evidence of the effectiveness of the theoretical metrics.

All papers mentioned above are based on one-dimensional LHCA. Janoowalla starts the study of the behavior of maximum length linear 2D CA in [12]. The results show that 1D CA and 2D CA perform equally well and much better than LFSRs regarding to their pseudorandom test and fault simulation. Cattell *et al.* study 2-by-n LHCA with regular configuration regarding their theory and application in [4]. The paper introduces their concept and a number of their properties. A recursive relation is proved which enables the characteristic polynomial to be efficiently calculated, and minimal cost, maximal length generators of this type are listed for sizes up to 500. A theoretical analysis of the two vector transition properties of the CA is given and it is shown that, for testing sequential faults over a set of standard benchmarks, the two-dimensional CA perform, on average, better than one-dimensional LHCA, and much better than LFSR.

In [4], Cattell *et al.* mainly present the properties of 2-by-n LHCA. In this thesis, we discuss an error in one of the results in [4] and provide a correct statement and prove the result, as well as deriving properties for more general n-by-m LHCA, including 3-by-n LHCA.

Chapter 3

Two-Dimensional Linear Hybrid Cellular Automata

In the previous chapter we have given a very simple two-dimensional LHCA, which is a 2-by-3 LHCA, and showed its next state equations. In this chapter, we introduce more complicated two-dimensional LHCA and provide their transition matrix using a general structure for all n-by-m LHCA. As the characteristic polynomial (CP) is very complex to be calculated because of their huge matrix structure, two ways to speed up the calculation of CP are introduced for 3-by-m LHCA. We conclude the chapter with a table of minimal-cost n-by-m LHCA with maximum length cycle for small values of n and m.

3.1 Transition Matrix

We first start to take a look at an example of n-by-m LHCA ($n, m \geq 3$). Figure 3.1 shows a 3-by-4 LHCA.

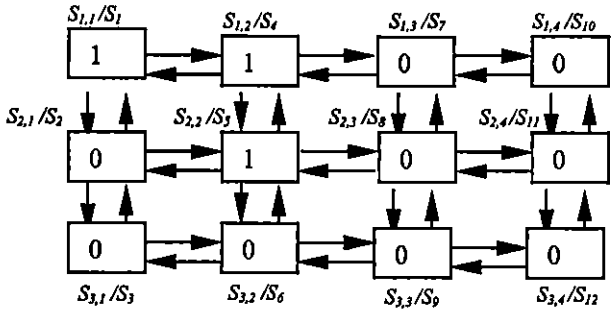


Figure 3.1. *A 3-by-4 LHCA example.*

Let $b_{i,j}$ be the cell rule for each cell, where i is the row number and j is the column number. Cell rules are defined in section (2.2.4). It can be cell-zero, which means it receives inputs only from its nearest neighbors, or cell-one, which means it receives inputs from its nearest neighbors and itself. We use a rule vector column by column to indicate the cell rules. For example, [[1, 0, 0], [1, 1, 0], [0, 0, 0], [0, 0, 0]] has the meaning as below:

$$[b_{1,1}, b_{2,1}, b_{3,1}] = [1, 0, 0]$$

$$[b_{1,2}, b_{2,2}, b_{3,2}] = [1, 1, 0]$$

$$[b_{1,3}, b_{2,3}, b_{3,3}] = [0, 0, 0]$$

$$[b_{1,4}, b_{2,4}, b_{3,4}] = [0, 0, 0]$$

For convenience to explain the relationship between the LHCA structure and transition matrix, we define two different notations here. In figure 3.1, for each cell, there are two notations. The first one is the one on the left using the double subscripts to identify each cell, the second one is the one on the right using only one single subscript with a specific sequence, which goes from top to bottom then from left to right. We can extend the notation to n-by-m LHCA discussed in the later sections. The two notations have the relation as below:

$$S_{i,j} = S_\alpha, \text{ where } \alpha = n \times i + j.$$

We use the first notation to derive the next state equations.

Let $s_{i,j}$ be the present state of cell (i, j) and $s^+_{i,j}$ be its next state. Then the next state equations for each cell are as follows:

$$\begin{aligned}
s^+_{1,1} &= s_{1,1} + s_{1,2} + s_{2,1} \\
s^+_{2,1} &= s_{1,1} + s_{2,2} + s_{3,1} \\
s^+_{3,1} &= s_{2,1} + s_{3,2} \\
s^+_{1,2} &= s_{1,1} + s_{1,2} + s_{1,3} + s_{2,2} \\
s^+_{2,2} &= s_{1,2} + s_{2,1} + s_{2,2} + s_{2,3} + s_{3,2} \\
s^+_{3,2} &= s_{2,2} + s_{3,1} + s_{3,3} \\
s^+_{1,3} &= s_{1,2} + s_{1,4} + s_{2,3} \\
s^+_{2,3} &= s_{1,3} + s_{2,2} + s_{2,4} + s_{3,3} \\
s^+_{3,3} &= s_{2,3} + s_{3,2} + s_{3,4} \\
s^+_{1,4} &= s_{1,3} + s_{2,4} \\
s^+_{2,4} &= s_{1,4} + s_{2,3} + s_{3,4} \\
s^+_{3,4} &= s_{2,4} + s_{3,3}
\end{aligned}$$

And the corresponding transition matrix is

$$\left(\begin{array}{ccc|ccc|ccc|ccc}
\mathbf{1} & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & \mathbf{0} & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & \mathbf{0} & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
1 & 0 & 0 & \mathbf{1} & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & \mathbf{1} & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & \mathbf{0} & 0 & 0 & 1 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 1 & 0 & 0 & \mathbf{0} & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & \mathbf{0} & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \mathbf{0} & 0 & 0 & 1 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \mathbf{0} & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \mathbf{0} & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \mathbf{0}
\end{array}\right).$$

The bold numbers in the matrix represent the rule vectors. The matrix has been divided into $4 \times 4 = 16$ blocks, each of which is of dimension 3×3. We denote T_1, T_2, T_3, T_4 as 3-b-3 tridiagonal matrices and I is 3-by-3 identity matrix. Thus we can simplify it as a block structure as below:

$$\begin{pmatrix}
T_1 & I & 0 & 0 \\
I & T_2 & I & 0 \\
0 & I & T_3 & I \\
0 & 0 & I & T_4
\end{pmatrix}$$

In general, for a 2D LHCA with regular configuration shown in figure 3.2, the leftmost and rightmost cells are assumed to have a constant 0 input as their left and right inputs, respectively. The same assumption applies to the topmost and bottommost cells. We only consider the LHCA to be fully connected in this thesis, which means that each inside cell receives inputs from all four of its neighbors. As it is a 2D structure, for the state of each cell, we define it as $s_{i,j}$. And the next state function, also called the transition function, is $s^+ = f_{i,j}(s)$. Thus we have f_i for each cell s_i as follows:

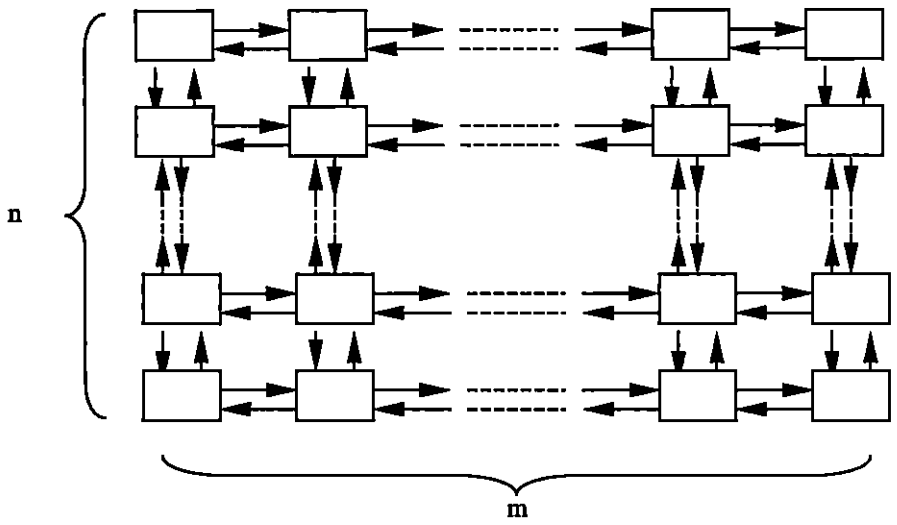


Figure 3.2. *An n-by-m LHCA.*

$$s^+_{i,j} = f_{i,j}(s) =$$

$$\begin{cases} s_{i-1,j} + s_{i+1,j} + s_{i,j-1} + s_{i,j+1} & (\text{if } b_{i,j} = 0) \\ s_{i-1,j} + s_{i+1,j} + s_{i,j} + s_{i,j-1} + s_{i,j+1} & (\text{if } b_{i,j} = 1) \end{cases}$$

where $b_{i,j}$ is the cell rule.

And we define

$$s_{i,j} = 0 \quad (\text{if } i = 0 \text{ or } j = 0),$$

which means for all cells on the left (right) edge, we assume they have input '0' from the left (right). Likewise for all cells on the top (bottom) row.

We can also use the matrix T to illustrate the transition function. Define S to be an array of the present state for all cells, then the next state of the array is defined using the transition matrix T by $S^+ = T \cdot S$. For any n-by-m LHCA shown in figure 3.2, the transition matrix has a block structure of $m \times m$ as below:

$$T = \begin{pmatrix} T_1 & I & 0 & 0 & \cdots & \cdots & 0 & 0 \\ I & T_2 & I & 0 & \cdots & \cdots & 0 & 0 \\ 0 & I & T_3 & I & \cdots & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cdots & I & 0 \\ 0 & 0 & 0 & 0 & \cdots & \cdots & T_{m-1} & I \\ 0 & 0 & 0 & 0 & \cdots & \cdots & I & T_m \end{pmatrix} \tag{3.1}$$

where each $T_1, T_2, \cdots, T_m$ is n-by-n tridiagonal and I is an n-by-n identity matrix. For each T_i with $1 \leq i \leq m$,

$$T_i = \begin{pmatrix} b_{1,i} & 1 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 1 & b_{2,i} & 1 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 1 & b_{3,i} & 1 & \cdots & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cdots & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & \cdots & b_{n-1,i} & 1 \\ 0 & 0 & 0 & 0 & \cdots & \cdots & 1 & b_{n,i} \end{pmatrix}$$

$b_{j,i}$ indicates one of the two rules (rule-zero and rule-one) used by the cells and $1 \leq j \leq n$. It is easy to see that the transition matrix of any n-by-m LHCA is symmetric.

For convenience, to explain the relationship between the LHCA structure and transition matrix, we use the second notation, which we define in the early part of this section. Here, we can easily find that in each row i of the matrix, the elements that are '1's are those cells that input to the cell i. And whether (i, i) is '1' or '0', depends on whether the cell rule is rule 1 or rule 0. For example, in figure 3.1, the cell s_5 connects with s_2, s_4, s_6, s_8 and cell rule is '1'. So in row 5, columns 2, 4, 5, 6, 8 are '1's.

3.2 Characteristic Polynomials

The characteristic polynomial(CP) of a LHCA (or any LFSM) determines some properties of the state space of a LHCA. For example, the CP is primitive if and only if the LHCA has a maximal length cycle [13]. In the later part of the chapter, we only investigate a simple case of n-by-m LHCA which is 3-by-m LHCA followed by some discussion about n-by-m LHCA. Two ways to efficiently compute the CPs are presented. One is deriving the recurrence relation and the recurrence, the other is using a matrix formulation.

3.2.1 Recurrence Relation

In this section, we derive the recurrence relation which allows for both efficient computation of the CP and analysis of its properties.

To derive the recurrence relation and the recurrence, we first define a submachine of a LHCA. Similarly to the 2-by-m CA discussed in [4], if M is a 3-by-m CA, the submachine of M is obtained by removing some cells from the end on each row. If we remove $(n-i)$ cells in the first row, $(n-j)$ cells in the second row, $(n-k)$ in the third row, then the submachine is denoted as $M\left|\begin{smallmatrix} i \\ j \\ k \end{smallmatrix}\right|$, and its CP is denoted as $\Delta\left|\begin{smallmatrix} i \\ j \\ k \end{smallmatrix}\right|$. For example, the submachine in figure 3.3 below is denoted as $M\left|\begin{smallmatrix} k-1 \\ k \\ k \end{smallmatrix}\right|$. Note that in this thesis, the submachine is not a CA unless $i=j=k$. But any submachine is still an LFSM. The original 3-by-m CA can be denoted by $M\left|\begin{smallmatrix} m \\ m \\ m \end{smallmatrix}\right|$ and its CP is $\Delta\left|\begin{smallmatrix} m \\ m \\ m \end{smallmatrix}\right|$.

The following lemmata are utilized to derive the recurrence relation when working with the CP of a symmetric 3-by-m LHCA in this thesis. Lemma 3.1 is proved in [4] for any n-cell LFSM. We present two more lemmata specifically for 3-by-m LHCA.

Lemma 3.1 [4]. Consider an k-cell LFSM M with a symmetric transition matrix T. Then for any $1 \leq i \leq k$, the CP of M equals

$$(x+t_{i,i})|\,xI+T_{\{i\}}| + \sum_{j=1}^{k} t_{i,j}|\,xI+T_{\{i,j\}}|,$$

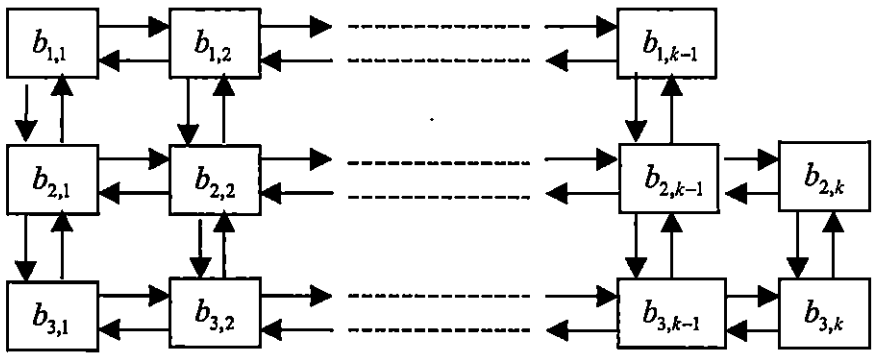


Figure 3.3. *A 3-by-m LHCA Submachine Structure.*

where $t_{i,j}$ is the element in matrix T and T_s denotes T with rows and columns contained in S deleted. And $|\, xI + T_{\{i\}}|$ is the CP of M without cell S_i.

For example, in Figure 3.1, we use the second notation in the figure to explain it. As it is a 3-by-4 LHCA, $k = 3 \times 4 = 12$. Let $i = 10$, then $t_{10,10} = 0$ and

$$\Delta \begin{vmatrix} 4 \\ 4 \\ 4 \end{vmatrix} = x\,|xI + T_{\{10\}}| + \sum_{j=1}^{12} t_{10,j}\,|xI + T_{\{10,j\}}|$$

$$= x\,|\,xI + T_{\{10\}}| + \,|\,xI + T_{\{10,7\}}| + \,|\,xI + T_{\{10,11\}}|\,,$$

because cell 7 and cell 11 are the neighborhood of cell 10.

Lemma 3.2. For any 3-by-m LHCA, if we remove only one cell from the end on any two rows, three different submachines are generated, which are

$$M \begin{vmatrix} m \\ m-1 \\ m-1 \end{vmatrix},\ M \begin{vmatrix} m-1 \\ m \\ m-1 \end{vmatrix},\ M \begin{vmatrix} m-1 \\ m-1 \\ m \end{vmatrix}$$

Their CPs are as below:

$$\Delta \begin{vmatrix} m \\ m-1 \\ m-1 \end{vmatrix} = (x + b_{1,m})\Delta \begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} + \Delta \begin{vmatrix} m-2 \\ m-1 \\ m-1 \end{vmatrix} \tag{3.2}$$

$$\Delta \begin{vmatrix} m-1 \\ m \\ m-1 \end{vmatrix} = (x + b_{2,m})\Delta \begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} + \Delta \begin{vmatrix} m-1 \\ m-2 \\ m-1 \end{vmatrix} \tag{3.3}$$

$$\Delta \begin{vmatrix} m-1 \\ m-1 \\ m \end{vmatrix} = (x + b_{3,m})\Delta \begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} + \Delta \begin{vmatrix} m-1 \\ m-1 \\ m-2 \end{vmatrix} \tag{3.4}$$

By applying lemma 3.1, we can easily prove the above lemma.

Lemma 3.3. For any 3-by-m LHCA, if we remove one cell from the end on one of the three rows, three different submachines are generated, which are

$$M\begin{vmatrix} m-1 \\ m \\ m \end{vmatrix},\ M\begin{vmatrix} m \\ m-1 \\ m \end{vmatrix},\ M\begin{vmatrix} m \\ m \\ m-1 \end{vmatrix}$$

Their CPs are as below:

$$\begin{aligned}\Delta\begin{vmatrix} m-1 \\ m \\ m \end{vmatrix} &= ((x+b_{2,m})(x+b_{3,m})+1)\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} + (x+b_{2,m})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-2 \end{vmatrix} \\ &\quad +(x+b_{3,m})\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-1 \end{vmatrix} + \Delta\begin{vmatrix} m-1 \\ m-2 \\ m-2 \end{vmatrix} \qquad (3.5)\\ \Delta\begin{vmatrix} m \\ m-1 \\ m \end{vmatrix} &= (x+b_{1,m})(x+b_{3,m})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} + (x+b_{1,m})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-2 \end{vmatrix} \\ &\quad +(x+b_{3,m})\Delta\begin{vmatrix} m-2 \\ m-1 \\ m-1 \end{vmatrix} + \Delta\begin{vmatrix} m-2 \\ m-1 \\ m-2 \end{vmatrix} \qquad (3.6)\\ \Delta\begin{vmatrix} m \\ m \\ m-1 \end{vmatrix} &= ((x+b_{1,m})(x+b_{2,m})+1)\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} + (x+b_{1,m})\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-1 \end{vmatrix} \\ &\quad +(x+b_{2,m})\Delta\begin{vmatrix} m-2 \\ m-1 \\ m-1 \end{vmatrix} + \Delta\begin{vmatrix} m-2 \\ m-2 \\ m-1 \end{vmatrix} \qquad (3.7)\end{aligned}$$

Proof. Applying lemma (3.1) and substituting lemma (3.2) into formula (3.5), we have

$$\begin{aligned}\Delta\begin{vmatrix} m-1 \\ m \\ m \end{vmatrix} &= (x+b_{2,m})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m \end{vmatrix} + \Delta\begin{vmatrix} m-1 \\ m-2 \\ m \end{vmatrix} + \Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} \\ &= (x+b_{2,m})((x+b_{3,m})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} + \Delta\begin{vmatrix} m-1 \\ m-1 \\ m-2 \end{vmatrix}) \\ &\quad + (x+b_{3,m})\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-1 \end{vmatrix} + \Delta\begin{vmatrix} m-1 \\ m-2 \\ m-2 \end{vmatrix} + \Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} \\ &= ((x+b_{2,m})(x+b_{3,m})+1)\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} + (x+b_{2,m})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-2 \end{vmatrix} \\ &\quad +(x+b_{3,m})\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-1 \end{vmatrix} + \Delta\begin{vmatrix} m-1 \\ m-2 \\ m-2 \end{vmatrix}\end{aligned}$$

Similar deduction can be used to prove formula (3.6) and (3.7).

□

Based on lemma (3.1), (3.2), (3.3), we can obtain the formulas to calculate the CP of a 3-by-m LHCA in theorem (3.4).

Theorem 3.4. The CP Δ of a 3-by-m LHCA M is calculated using the recurrence relation:

$$\begin{aligned}
\Delta\begin{vmatrix} m \\ m \\ m \end{vmatrix} &= ((x+b_{1,m})(x+b_{2,m})(x+b_{3,m})+b_{1,m}+b_{3,m})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} \\
&+((x+b_{1,m})(x+b_{2,m})+1)\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-2 \end{vmatrix}+((x+b_{2,m})(x+b_{3,m})+1)\,\Delta\begin{vmatrix} m-2 \\ m-1 \\ m-1 \end{vmatrix} \\
&+((x+b_{1,m})(x+b_{3,m}))\,\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-1 \end{vmatrix}+(x+b_{1,m})\,\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-2 \end{vmatrix} \\
&+(x+b_{2,m})\,\Delta\begin{vmatrix} m-2 \\ m-1 \\ m-2 \end{vmatrix}+(x+b_{3,m})\,\Delta\begin{vmatrix} m-2 \\ m-2 \\ m-1 \end{vmatrix}+\Delta\begin{vmatrix} m-2 \\ m-2 \\ m-2 \end{vmatrix}
\end{aligned} \tag{3.8}$$

where $b_{i,j}$ is the cell rule of the machine ($i \le 3, j \le m$).

The base cases are defined by

$$\begin{cases} \Delta\begin{vmatrix} k_1 \\ k_2 \\ k_3 \end{vmatrix} = 0 & \text{if } k_1 = -1 \text{ or } k_2 = -1 \text{ or } k_3 = -1; \\ \Delta\begin{vmatrix} k \\ k \\ k \end{vmatrix} = 1 & \text{if } k = 0. \end{cases}$$

Proof. Applying lemma 3.1, 3.2 and 3.3, we have

$$\begin{aligned}
\Delta\begin{vmatrix} m \\ m \\ m \end{vmatrix} &= (x+b_{1,m})\Delta\begin{vmatrix} m-1 \\ m \\ m \end{vmatrix}+\Delta\begin{vmatrix} m-2 \\ m \\ m \end{vmatrix}+\Delta\begin{vmatrix} m-1 \\ m-1 \\ m \end{vmatrix} \\
&= (x+b_{1,m})(((x+b_{2,m})(x+b_{3,m})+1)\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix}+(x+b_{2,m})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-2 \end{vmatrix} \\
&+(x+b_{3,m})\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-1 \end{vmatrix}\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-2 \end{vmatrix})+(x+t_{2,k})\Delta\begin{vmatrix} m-2 \\ m-1 \\ m \end{vmatrix}+\Delta\begin{vmatrix} m-2 \\ m-2 \\ m \end{vmatrix} \\
&+\Delta\begin{vmatrix} m-2 \\ m-1 \\ m-1 \end{vmatrix}+(x+t_{3,k})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix}+\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-2 \end{vmatrix} \\
&= ((x+b_{1,m})(x+b_{2,m})(x+b_{3,m})+b_{1,m}+b_{3,m})\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix} \\
&+((x+b_{1,m})(x+b_{2,m})+1)\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-2 \end{vmatrix}+((x+b_{2,m})(x+b_{3,m})+1)\,\Delta\begin{vmatrix} m-2 \\ m-1 \\ m-1 \end{vmatrix} \\
&+((x+b_{1,m})(x+b_{3,m}))\,\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-1 \end{vmatrix}+(x+b_{1,m})\,\Delta\begin{vmatrix} m-1 \\ m-2 \\ m-2 \end{vmatrix}
\end{aligned}$$

$$+(x+b_{2,m})\,\Delta\begin{vmatrix} m-2 \\ m-1 \\ m-2 \end{vmatrix}+(x+b_{3,m})\,\Delta\begin{vmatrix} m-2 \\ m-2 \\ m-1 \end{vmatrix}+\Delta\begin{vmatrix} m-2 \\ m-2 \\ m-2 \end{vmatrix}$$

□

As an example, we compute the CP of the 3-by-4 CA shown in figure 3.1 using the recurrence relation. The computation starts from $m = 0$, and sequence is from up to down. In the rightmost and bottommost of the table 3.1, we see that the CP is

$$x^{12}+x^{11}+x^9+x^8+x^7+x^6+x^4+x+1 \tag{3.9}$$

The polynomial is primitive, which means that the CA has a maximum length cycle.

$m=$	0	1	2	3	4
$\Delta\begin{vmatrix} m-1 \\ m-1 \\ m-1 \end{vmatrix}$	0	1	x^3+x^2+1	$x^6+x^5+x^3+x$	$x^9+x^8+x^7$ $+x^4+x+1$
$\Delta\begin{vmatrix} m-1 \\ m-1 \\ m \end{vmatrix}$	0	x	x^4+x^3 $+x^2+1$	$x^7+x^6+x^5+x^2$ $+x+1$	$x^{10}+x^9+x^7+x^6$ $+x^5+x^3+x^2+1$
$\Delta\begin{vmatrix} m-1 \\ m \\ m-1 \end{vmatrix}$	0	x	x^4+1	$x^7+x^6+x^5+x^4$ $+x^3+x+1$	$x^{10}+x^9+x^7+x^5$ $+x^4+x+1$
$\Delta\begin{vmatrix} m \\ m-1 \\ m-1 \end{vmatrix}$	0	$x+1$	x^4+x	$x^7+x^6+x^5$ $+x^4+1$	$x^{10}+x^9+x^7+x^6$ $+x^3+x^2+x$
$\Delta\begin{vmatrix} m-1 \\ m \\ m \end{vmatrix}$	0	x^2+1	x^5+x^2+1	$x^8+x^7+x^6$ $+x^5+x^3$	$x^{11}+x^{10}+x^8+x^7$ $+x^4+x$
$\Delta\begin{vmatrix} m \\ m-1 \\ m \end{vmatrix}$	0	x^2+x	$x^5+x^3+x^2$ $+x+1$	$x^8+x^7+x^4$ $+x^2+1$	$x^{11}+x^{10}+x^9+x^7$ $+x^6+x^5+x^4+1$
$\Delta\begin{vmatrix} m \\ m \\ m-1 \end{vmatrix}$	0	x^2+x+1	x^5+x^4 $+x+1$	$x^8+x^7+x^6$ $+x^3+x+1$	$x^{11}+x^{10}+x^8+x^7$ $+x^6+x$
$\Delta\begin{vmatrix} m \\ m \\ m \end{vmatrix}$	1	x^3+x^2+1	$x^6+x^5+x^3$ $+x$	$x^9+x^8+x^7$ $+x^4+x+1$	$x^{12}+x^{11}+x^9+x^8$ $+x^7+x^6+x^4+x+1$

Table 3.1. *CP Calculation for a 3-by-4 LHCA*

3.2.2 Matrix Formulation of the Characteristic Polynomial

In this section, we are going to show another way that is able to speed up the calculation of the CP for a 3-by-m LHCA. In [4], it shows that the CP of a 2-by-m LHCA can be modelled as the product of 4-by-4 matrices based on its recurrence relation. It enables us to derive a concatenation relation for CA. Similarly, we can find that CP for a 3-by-m can be modelled as the product of 8-by-8 matrices.

Before describing the implementation in detail, we need to extend our notation of a submachine, to allow for structures that do not necessarily include the first column cells. Under this premise, cells in a group of columns are considered as a submachine and can be defined as $M\left|\begin{smallmatrix} i_1 j_1 \\ i_2 j_2 \\ i_3 j_3 \end{smallmatrix}\right|$. And let $\Delta\left|\begin{smallmatrix} i_1 j_1 \\ i_2 j_2 \\ i_3 j_3 \end{smallmatrix}\right|$ denotes the CP of $M\left|\begin{smallmatrix} i_1 j_1 \\ i_2 j_2 \\ i_3 j_3 \end{smallmatrix}\right|$.

To make it simpley, we only consider the submachine with $i_1 = i_2 = i_3 = i$ and $j_1 = j_2 = j_3 = j$, which is denoted as $M\left|\begin{smallmatrix} ij \\ ij \\ ij \end{smallmatrix}\right|$. For example, in Figure 3.4, the left one is submachine $M\left|\begin{smallmatrix} 11 \\ 11 \\ 11 \end{smallmatrix}\right|$ and the right one is submachine $M\left|\begin{smallmatrix} 12 \\ 12 \\ 12 \end{smallmatrix}\right|$ for a 3-by-4 LHCA in Figure 3.1.

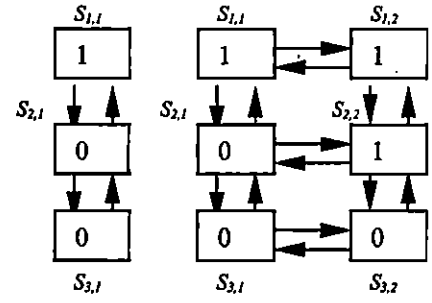


Figure 3.4. *Submachines of a 3-by-4 LHCA*

In a 3-by-m LHCA, for each column of cells with cell-rule $b_{1,i}$, $b_{2,i}$, $b_{3,i}$ and $i \leq m$, we can define that it has a matrix structure $P_{i,i}$, of which the elements are as shown in figure 3.5.

$$
\begin{array}{cccccccc}
(x+b_{1,m})(x+b_{2,m})(x+b_{3,m})+b_{3,m}+b_{1,m} & (x+b_{2,m})(x+b_{3,m})+1 & (x+b_{1,m})(x+b_{3,m}) & (x+b_{1,m})(x+b_{2,m})+1 & x+b_{3,m} & x+b_{2,m} & x+b_{1,m} & 1 \\
(x+b_{2,m})(x+b_{3,m})+1 & 0 & x+b_{3,m} & x+b_{2,m} & 0 & 0 & 1 & 0 \\
(x+b_{1,m})(x+b_{3,m}) & x+b_{3,m} & 0 & x+b_{1,m} & 0 & 1 & 0 & 0 \\
(x+b_{1,m})(x+b_{2,m})+1 & x+b_{2,m} & x+b_{1,m} & 0 & 1 & 0 & 0 & 0 \\
x+b_{3,m} & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
x+b_{2,m} & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
x+b_{1,m} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}
$$

Figure 3.5. *8-by-8 Matrix Structure for Each Column in 3-by-m LHCA.*

Using the recurrence relation, we obtain that the product of each matrix for each column of cells from i through j results in a matrix consisting of the CPs of various submachines of M, with the CP of M as the top left entry. We denote the product

$$P_{i,j} = P_{i,i}\, P_{(i+1),(i+1)} \cdots P_{(j-1),(j-1)}\, P_{j,j}$$

By repeating the multiplication of each matrix, we can get the matrix for $M\left|\begin{smallmatrix} i & j \\ i & j \\ i & j \end{smallmatrix}\right|$. Any CA can be divided into several small CA. After calculating each small CA, they can be concatenated to form the original CA.

For example, consider the LHCA in Figure 3.1 as the concatenation of four CA, which have rule vectors $[1,0,0]$, $[1,1,0]$, $[0,0,0]$, and $[0,0,0]$ by column. As an example, $P_{1,1}$ equals to

$$\begin{pmatrix}
(x+1)x^2+1 & x^2+1 & (x+1)x & (x+1)^2+1 & x & x & x+1 & 1 \\
x^2+1 & 0 & x & x & 0 & 0 & 1 & 0 \\
(x+1)x & x & 0 & x+1 & 0 & 1 & 0 & 0 \\
(x+1)^2+1 & x & x+1 & 0 & 1 & 0 & 0 & 0 \\
x & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
x & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
x+1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}.$$

Then we have

$$\begin{aligned}
P_{1,4} &= P_{1,1} \cdot P_{2,2} \cdot P_{3,3} \cdot P_{4,4} = (P_{1,1} \cdot P_{2,2}) \cdot (P_{3,3} \cdot P_{4,4}) \\
&= P_{1,2} \cdot P_{3,4} \\
&= P_{1,4}
\end{aligned}$$

As the final 8-by-8 matrix structure generated is very big, we don't show the detail values here. When the final matrix is generated, we should be able to see the top left entry in the matrix is

$$x^{12} + x^{11} + x^9 + x^8 + x^7 + x^6 + x^4 + x + 1$$

which is the same as characteristic polynomial (3.9).

In appendix A, we have the program written in Maple language to calculate the characteristic polynomial based on the recurrence relation given in this section. The program convert the recursion into a simple loop structure. In this way, when m increases, we only need to run one more loop. The number of operations for each loop is growing linearly. This is similar to the analysis of the concatenation relation. If we compute the characteristic polynomial directly, the number of operation is growing factorially, which is slower than either way we discuss above.

3.2.3 Primitive Characteristic Polynomials

Usually for any application, we attempt to minimize the hardware cost of the implementation. For this purpose, one way is to minimize the number of cells that use rule-one, since rule-one always has a slightly higher complexity both for evaluation and for implementation. The algorithms to find out the minimal-cost, maximal-length LHCA can be described in pseudo-code as follows and the program is printed in appendix B:

Define a n-by-m LHCA;

Initialization:

Transition Matrix T is defined in formula 3.1

CP: $\Delta = |xI - T|$

let cell-rule $b_{i,j} := 0$, where $i = \{1, \cdots, n\}$ and $j = \{1, \cdots, m\}$;

Start:

substitute $b_{i,j}$ in T

if Δ is primitive

then do:

print "The minimal-cost, maximal-length LHCA is Δ."

exit;

else do:

for k =1 to $n \times m$ do

for all k combinations of $n \times m$ number do

```
            go to Start
        end
    end
    print "There is no primitive CP for the n-by-m LHCA."
end.
```

With the algorithm above, we use a program developed by Cathy written in "Maple" language to find the primitive CP over $GF(2)$ with minimal rule-one n-by-m $LHCA$. With the increase of n and m, the running time for the program increases rapidly especially when more rule-one cells are needed to get the first minimal-cost 2D LHCA. For example, we want to find the minimal cost 6-by-6 LHCA. It takes more than two weeks to finish the program because the minimal cost is 6 cells. Due to the limited computer resource to calculate large-dimensional matrices, so far we only have the results shown in Table 3.2. The entries in the table indicate the number of cell using rule-one. For example, for the entry 3-by-4 LHCA, the entry is '1, 4, 5'. It means cells s_1, s_4, s_5 using rule-one, so the rule vector is

$$\begin{aligned}&[[b_{1,1}, b_{2,1}, b_{3,1}], [b_{1,2}, b_{2,2}, b_{3,2}], [b_{1,3}, b_{2,3}, b_{3,3}], [b_{1,4}, b_{2,4}, b_{3,4}]]\\ &= [[1,0,0], [1,1,0], [0,0,0], [0,0,0]],\end{aligned}$$

This corresponds to the CA shown in figure 3.1.

Note that, for 3-by-3 LHCA, there does not exist any LHCA with a maximum length cycle.

From these limited results, some observations can be found as follows:

1. All the maximal length n-by-m LHCA have at least two rule-one cells for $n, m \geq 3$ except that 3-by-3 LHCA do not exist any maximum length one.
2. There does not exist a maximal length uniform LHCA, which has all cells using rule-zero or rule-one.
3. For any m-by-m LHCA ($m > 3$), to get maximal length LHCA, there are at least m rule-one cells.

n-by-m LHCA	# of cell with rule-one	n-by-m LHCA	# of cell with rule-one
3-by-3	-	3-by-4	1, 4, 5
3-by-5	1, 7, 14	3-by-6	1, 7, 8
3-by-7	1, 4, 5	3-by-8	1, 2
3-by-9	1, 7, 20	3-by-10	1, 4
3-by-11	1, 2, 7	3-by-12	1, 14
3-by-13	1, 7, 34		
4-by-4	1, 5, 7, 9	4-by-5	1, 4, 9
4-by-6	2, 11	4-by-7	2, 6
4-by-8	13, 30	4-by-9	1, 5, 13, 25
4-by-10	1, 2		
5-by-5	1, 6, 9, 16, 21	5-by-6	6, 15
5-by-7	1, 6, 11, 22	5-by-8	1, 2, 11
6-by-6	1, 7, 10, 13, 16, 19	6-by-7	1, 31

Table 3.2. *Minimal Cost n-by-m LHCA, with Maximum Length Cycles*

Observation 1 and 2 are similar with the observations got from [4] for 2-by-n LHCA. We can assume that our observations are quite reasonable though there are not enough information from the table.

3.3 Discussion and Conclusion

In this chapter, we have presented the general transition matrix structure of a n-by-m LHCA. We introduced two ways to speed up the calculation of CP of a 3-by-m transition matrix. One way is to use the recurrence relation. Though the way is still feasible for 3-by-m LHCA, the formulas already turn out to be complicated. So it is not worthwhile to continue this way for n-by-m LHCA ($n \geq 4$). It is the same situation for the second way. For each 2-by-m LHCA, the matrix dimension of submachine is 4-by-4. For each 3-by-m LHCA, the matrix dimension is 8-by-8. Based on this, we can anticipate that the size will be getting bigger when n increases, that makes the calculation much more complicated. For example, for a 4-by-m CA, the matrix dimension is assumed to be 16-by-16, etc.

As primitive CP represents the LHCA is a maximum length cycle one, a table of minimal cost with maximum length cycle n-by-m LHCA is shown followed by some observations.

In next chapter, we present another important characteristic of LHCA, which is a discussion of their transition properties.

Chapter 4

Transition Properties

Two-dimensional (2D) LHCA are investigated especially for their transition properties. Each state is used as a test vector. In VLSI testing applications, for delay faults (we talk about delay faults in the next chapter), the actual test vector sequence produced by a generator is critical since a fault requires a pair of vectors for stimulation. We are thus concerned with the sequence of the transitions produced. Moreover, for a n-cell CA, it is not sufficient to just consider transitions of n-cell vectors, but we also need to look at the transitions of k-cell substate vectors for $k < n$. To make this chapter easier to understand, we start by giving the relevant background of transition properties and necessary notation.

We are focusing on 2-by-m LHCA in this thesis. In [4], it presents two-vector transition property. For those vector generators with maximum length cycle, it derives the theorem using a formula to calculate different substate vectors which producing maximum number of distinct transitions. We do simulation studies to count all the possible combinations in a brute force way and are surprised to find that the resulting numbers we got are different from the numbers obtained from the formula. Thus, we investigate the formula and find that there is a problem with the result. After careful investigation, we formulate a new and corrected statement of the result, and derive a proof of its correctness. This result is completely consistent with our simulation results.

4.1 Background

In this section, we provide some background of the transition properties and corresponding notation. We give the definition of substate vector transition and the next state function of a substate vector. We also show the upper and lower bounds of the number of the distinct transitions for substate vectors as proved in [25]. The concepts of rank and partner set are introduced afterwards to obtain the number of different substate vectors which produce upper bound transitions. Some examples are given to help understand the concepts in the last part of the section.

Let's define a k-cell substate vector, and the corresponding transitions first([25][8]).

Definition 4.1[25]. For a given n-cell LFSM state vector $s = (s_1, s_2, \ldots, s_n)^T, s_p \in \{0,1\}$, $1 \leq p \leq n$, a k-cell substate vector ω of s is defined by

$$\omega = (s_{i_1}, s_{i_2}, \ldots, s_{i_k})^T,$$

and a transition corresponding to ω is defined as

$$\langle (s_{i_1}, s_{i_2}, \ldots, s_{i_k})^T, (s_{i_1}^+, s_{i_2}^+, \ldots, s_{i_k}^+)^T \rangle,$$

where $1 \leq i_1 \leq i_2 \leq \ldots \leq i_k \leq n$.

For notational convenience, $\overline{\omega}$ is used to denote a substate vector of s with cells which are not in ω.

In general, for a given n-cell LFSM with state transition matrix T, the next state s^+ is defined as

$$s^+ = T \cdot s. \tag{4.1}$$

If we only need to consider the ω^+ for a k-cell substate vector ω of s, then we have

$$\omega^+ = T_{[\omega,\overline{\omega}]} \cdot s\,, \tag{4.2}$$

where $T_{[\omega,\overline{\omega}]}$ only includes columns of T, which correspond to the elements in ω.

Moreover, we can partition $T_{[\omega,\overline{\omega}]}$ into T_ω and $T_{\overline{\omega}}$. T_ω includes rows corresponding to ω, and $T_{\overline{\omega}}$ includes rows corresponding to $\overline{\omega}$. The next substate function of ω in (4.2) can also be given as

$$\omega^+ = T_\omega \cdot \omega + T_{\overline{\omega}} \cdot \overline{\omega} \tag{4.3}$$

As we are concerned with pairs of vectors, one primary property is to find out the number of distinct transitions. We count every transition even if $(s_{i_1}, s_{i_2}, \ldots, s_{i_k})^T = (s_{i_1}^+, s_{i_2}^+, \ldots, s_{i_k}^+)^T$ because it simplifies the derivation of a general equation to evaluate the number of transitions for a given substate vector that we are going to discuss in the following section.

In [8], Furuya and McCluskey have proven the theorem below:

Theorem 4.1[8]. For a given k-cell substate vector ω with next state function

$$\omega^+ = T_\omega \cdot \omega + T_{\overline{\omega}} \cdot \overline{\omega},$$

2^r distinct transitions occur from every k-cell substate vector, where $r = rank(T_{\overline{\omega}})$, while the whole state vector s goes through all possible 2^n states.

Also, in [26], the lower and upper bounds of the number of the distinct transitions have been proved in the theorem below:

Theorem 4.2 (Upper and lower bounds)[26]. Consider any n-cell $LFSM$ vector generator with a maximum length cycle. Let $F(k)$ be the maximal number of distinct transitions corresponding to a k-cell substate vector ω. We have

$$\text{upper bound: } F(k) \leq \begin{cases} 2^{2k}, & 1 \leq k < \lceil n/2 \rceil, \\ 2^n - 1, & \lceil n/2 \rceil \leq k \leq n, \end{cases}$$

$$\text{lower bound: } F(k) \geq \begin{cases} 2^k, & 1 \leq k < n, \\ 2^n - 1, & k = n, \end{cases}$$

In our application, we are considering a 2-by-m LHCA. In this case, the total number of cells n is $2m$. And we consider substate vector, which contains half the m cells, so $k = m$.

Here we give some examples to help explain the definitions and the theorems described above.

Example 4.1. For a 2-by-3 LHCA shown in figure 4.1, we name each cell a successive number from left to right (the notation is the second one we introduced in section 3.1). Let $s = (s_1, s_2, s_3, s_4, s_5, s_6)^T$ be the state vector. Assume the LHCA uses a rule vector [[1,0],[1,0],[0,1]].

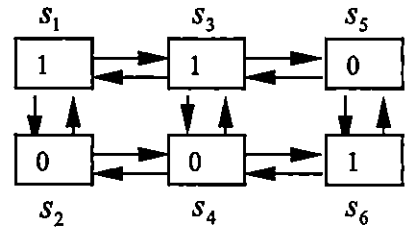


Figure 4.1. *A 2-by-3 LHCA example.*

Then we have the transition matrix

$$T = \left(\begin{array}{cc|cc|cc} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ \hline 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ \hline 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right)$$

If we chose a 3-cell substate vector $\omega = (s_1, s_5, s_6)^T$, then $\overline{\omega} = (s_2, s_3, s_4)^T$. And according to formulas (4.2) and (4.3), we have

$$\omega^+ = \begin{pmatrix} s_1^+ \\ s_5^+ \\ s_6^+ \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_5 \\ s_6 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} s_2 \\ s_3 \\ s_4 \end{pmatrix}$$

Here

$$r = rank(T_{\overline{\omega}}) = rank \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = 3.$$

Based on theorems (4.1) and (4.2), for each value of $(s_1, s_5, s_6)^T$, there are 2^3 distinct next values $(s_1^+, s_5^+, s_6^+)^T$, and the total number of distinct transitions for the given substate vector is $(2^3 2^3 - 1 = 2^6 - 1)$ because the all zeros state is not included in the sequence.

Example 4.2. Using the same CA shown in figure 4.1, but in this example, we may choose another 3-cell substate vector $\omega = (s_1, s_4, s_6)^T$, then $\overline{\omega} = (s_2, s_3, s_5)^T$. We have

$$\omega^+ = \begin{pmatrix} s_1^+ \\ s_4^+ \\ s_6^+ \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_4 \\ s_6 \end{pmatrix} + \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} s_2 \\ s_3 \\ s_5 \end{pmatrix}$$

In this example,

$$r = rank(T_{\overline{\omega}}) = rank \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = 2,$$

and the total number of distinct transitions is $(2^2 2^2 = 2^4)$.

From the two examples above, we can see that if a different k-cell substate vector is chosen, the ranks and the numbers of distinct transitions generated may be different.

For 2-by-m LHCA, we have known that a key to determine whether a given m-cell substate vector ω has $(2^{2m} - 1)$ transition capability is to check if the corresponding $T_{\overline{\omega}}$ has rank m. However, to avoid computing the rank of $T_{\overline{\omega}}$, the idea of a partner set is introduced

in [25] and it is proven that the cardinality of this set gives the rank of $T_{\overline{\omega}}$.

Before stating the definition of a partner set, we review a maximal matching problem in a bipartite graph [6]. Let $G = (V, E)$ be a bipartite graph with V partitioned as $X \cup Y$ (Each edge of E has the form (x, y) with $x \in X$ and $y \in Y$).

(a) A *matching of* G is a subset of E such that no two edges share a common vertex in X or Y.

(b) A *maximal matching* in G is one that matches as many vertices in X as possible with vertices in Y.

The definition of partner set is stated below.

Definition 4.2. Let ω be a k-cell substate vector of a state vector $s = (s_1, s_2, \ldots, s_n)$ for an n-cell LFSM. $G = (V, E)$ be a bipartite graph with V partitioned as $X \cup Y$, where

$$\begin{aligned} V &= \{s_1, s_2, \ldots, s_n\}, \\ X &= \{s_i \mid s_i \text{ is in } \omega\}, \\ Y &= \{s_j \mid s_j \text{ is in } \overline{\omega}\}, \\ E &= \{(s_i, s_j) \mid s_i \in X, s_j \in Y, \\ &\qquad \text{and } s_j \text{ is an eligible partner of } s_i\}. \end{aligned}$$

If $M, M \subseteq E$, is a maximal matching of G, then $ps(\omega) = \{s_j \mid (s_i, s_j) \in M\}$ is a partner set of ω and $|ps(\omega)| = |M|$.

In example 4.1, a corresponding bipartite graph G is shown in Figure 4.2, where $\bullet$ are the elements chosen in ω, and $\circ$ are the elements in $\overline{\omega}$. Then we have $X = \{s_1, s_5, s_6\}$, $Y = \{s_2, s_3, s_4\}$ and $E = \{(s_1, s_2), (s_1, s_3), (s_5, s_3), (s_6, s_4)\}$.

The possible maximal matching is $\{(s_1, s_2), (s_5, s_3), (s_6, s_4)\}$, and the corresponding partner set is $\{s_2, s_3, s_4\}$. Then $|ps(\omega)| = 3$.

Similarly in example 4.2, the corresponding partner set is $\{s_2, s_3, s_5\}$ and $|ps(\omega)| = 3$ too.

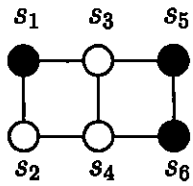


Figure 4.2. *A bipartite graph for example 4.1*

Note that we ignore self-loop edges since they do not affect the evaluation of the rank of $T_{\overline{\omega}}$.

In [4], the relation between partner set and rank given in the theorem below has also been proven.

Theorem 4.3[4]. Let ω be a k-cell substate vector of a $2n$-cell state vector s for a 2-by-m LHCA with a maximum length cycle. If rank $(T_{\overline{\omega}}) = k$, we have

- $|ps(\omega)| = k$;
- No two columns in $T_{\overline{\omega}}$ are the same.

Also mentioned in [4], the two conditions in the above theorem are not sufficient to determine whether rank$(T_{\overline{\omega}}) = k$. In example 4.2, we see that $|ps(\omega)| = 3$, but $T_{\overline{\omega}} = 2$. We discuss more about this in the following section.

4.2 Transitions of 2-by-m LHCA

We have known that it is important to investigate a pair of vectors, namely the two-vector transitions, produced by a generator for VLSI testing applications. In section 4.1, we have introduced the concepts of rank and partner set to help investigate the transition properties. For 2-by-m LHCA, Cattell *et al.* derive a theorem containing formulae calculating different substate vectors which produce a maximum number of distinct transitions [4]. Unfortunately, we find that the formulae are incorrect. In this section, we first give the theorem proposed in the paper and show the inconsistent results come from the theorem and our simulation. Then we analyze the reasons for the discrepancy, which enable us to correct the theorem and prove the new results.

4.2.1 Previous Work

A theoretical analysis of the two-vector transition property is important for testing delay faults. In this section, let's first take a look at what has been shown in paper [4]. It derives a theorem, giving a doubly recursive expression for the number of transitions. Unfortunately, there are some errors in the theorem. We illustrate as follows:

Theorem 4.4[4]. Let $f(m)$ be the maximum number of distinct m-cell substate vectors which produce 2^{2m} transitions for the 2-by-m CA with a maximum length cycle. We have

$$f(m) = 2[g(m-1) + f(m-2)],\ m > 2$$
$$f(0) = 1,\ f(1) = 2,\ f(2) = 4,$$

where $g(m)$ is defined as follows:

$$g(m) = g(m-1) + 3f(m-2) + f(m-3),\ m > 2$$
$$g(1) = 1,\ g(2) = 4.$$

There is a minor mistake in the theorem that for 2-by-m LHCA, if any m-cell substate vector is chosen, the maximum transitions can only be $(2^{2m} - 1)$ rather than 2^{2m}.

To investigate the functions shown in the theorem, we wrote a program using Maple to calculate all of the transitions. The purpose of the program is to calculate the number of distinct m-cell state vectors which produce $(2^{2m} - 1)$ transitions. We generated the maximum length cycle plus the first vector, which is 2^{2m} state vectors and apply all the combinations of m-cell substate vectors in the list. Then we counted the distinct transitions.

Table 4.1 below is a comparison of numbers results from the formula in the theorem and from the direct simulation.

It is easy to find that the number derived from the theorem is different from the result from the simulation for $m = 5,\ 6$ and 7. And the latter is less than the former.

So it appeared that there might be an error in theorem 4.4. We looked in detail at the theorem and its proof, which is case by case analysis, to see if any fail cases are omitted.

The proof defines two structures that need to be avoided. In figure 4.3, using our previous notation, nodes marked with $\bullet$ are included in ω, nodes marked with $\otimes$ may or may

m	From Theorem 4.4	From Simulation
3	12	12
5	74	68
6	190	172
7	906	420

Table 4.1. *Comparison of the numbers get from theorem and simulation*

not be included in ω and those marked $\circ$ are included in $\overline{\omega}$.

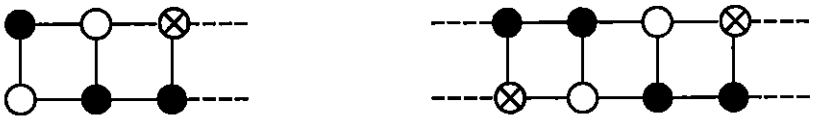


Figure 4.3. *The structures to be avoided in [4].*

For 2-by-5 LHCA, we went through all the possible cases that do not generate maximum transitions ($2^{10} - 1 = 1023$) and found that there are at least two fail cases that are not included in the two structures shown in figure 4.3.

For example, in figure 4.4, we can find that in both cases, $|ps(\omega)| = 5$, while $\text{rank}(T_{\overline{\omega}}) = 4 \neq 5$.

Figure 4.4. *The structures not included.*

At this point, it is clear that there are omissions in the proof of the theorem, so clearly, the statement given is incorrect. In the following section, we give our new derivation.

4.2.2 The New Derivation

We continue our investigation of the transition properties and we derive a rule that allows us to determine whether the rank of the matrix $T_{\varpi} = m$ if a specific structure of 2-by-m LHCA is given.

Before we describe the rule, let's define more notation to help understand the rule.

For any 2-by-m LHCA with regular configuration, consider each column and it is clear that each column has only one of the four possible structures in figure 4.5. We name them (1),(2),(3),(4) respectively. Again, • means the cell is included in ω for substate vector, ○ means it is included in ϖ.

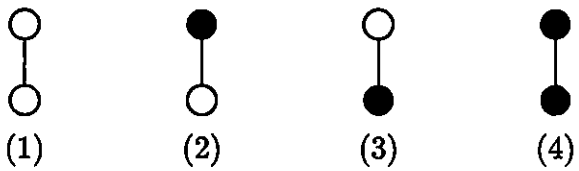


Figure 4.5. *All structures in 2-by-m LHCA.*

Then each 2-by-m LHCA can be denoted using the number of these column structures. For example, the structure in figure 4.6 can be denoted as " 21423 ".

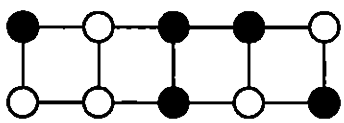
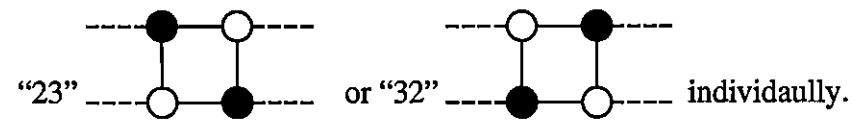


Figure 4.6. *A 2-by-5 example.*

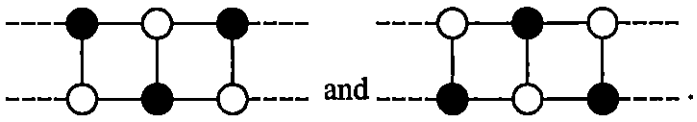
Based on the notation above, we have the following rules:

Rule 1. For a 2-by-m LHCA, if a m-cell substate vector has partner sets that $|ps(w)| = m$, then the rank of the matrix $T_{\varpi} = m$ if the structure doesn't contain "p 23 q" or "q 32 p", and $p \in \{\varnothing, 1, 2, 4\}$, $q \in \{\varnothing, 1, 3, 4\}$.

In another way, we can describe it with diagrams. To make rank$(T_{\varpi}) = m$, the structure of 2-by-m LHCA must not have the structure



But structures "232" and "323" are not counted, like



For example, figure 4.7 lists some of the structures to be avoided.

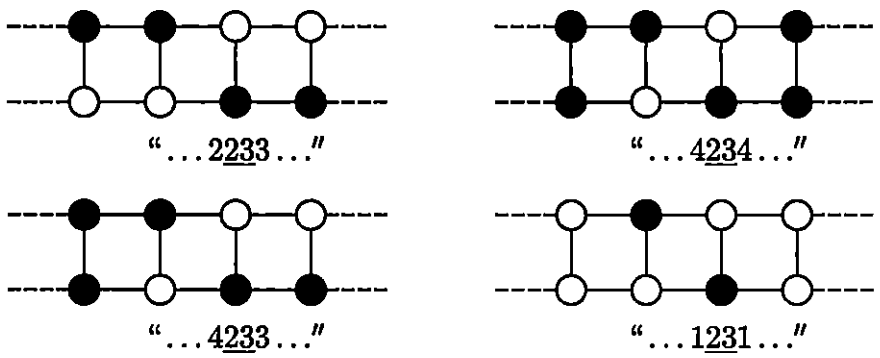


Figure 4.7. *The structures to be avoided.*

Rule 2. For a 2-by-m LHCA, if a m-cell substate vector has partner sets that $|ps(w)| = m$, then the rank of the matrix $T_{\overline{\omega}} = m - r$ if the number of "32" and "23" is r excluding "232" and "323".

To further explain the rule above, we compare the two 2-by-5 CA structures shown in figure 4.8.

In (a), the rank$(T_{\overline{\omega}}) = 5$. But in (b), as it contains **ONE** "23" individually excluding "323", the rank$(T_{\overline{\omega}}) = 5 - 1 = 4$.

Actually we can verify this from another view of the matrix. In section 3.1, we have found that for each row in the transition matrix T, the element '1' is associated with the cells connected. Thus for a 2-by-m LHCA, we have state vector s, substate vector ω of s, and matrix $T_{\overline{\omega}}$, which satisfies columns $\in \omega$, rows $\in \overline{\omega}$.

(a) "22323" (b) "323 23"

Figure 4.8. *Two 2-by-5 LHCAs to be compared.*

In $T_{\overline{\omega}}$, for each row, the columns which are '1's are those cells that can be a partner of those included in ω.

Let's consider the example of figure 4.6. $\omega = (s_1, s_5, s_6, s_7, s_{10})$, and $\overline{\omega} = (s_2, s_3, s_4, s_8, s_9)$. The structure is denoted as "21423". We can see that there is **ONE** "23" in the structure, so the rank = $5 - 1 = 4$. We can verify the rank using the matrix. For each element in ω, table 4.2 shows their partners.

ω	$ps(\omega)$	Notes
s_1	s_2, s_3	s_2 or s_3 can be a partner of s_1
s_5	s_3	s_3 is a partner of s_5
s_6	s_4, s_8	s_4, or s_8 can be a partner of s_6
s_7	s_8, s_9	s_8, or s_9 can be a partner of s_7
s_{10}	s_8, s_9	s_8, or s_9 can be a partner of s_{10}

Table 4.2. *An example of substate vector and its partner set.*

So the corresponding matrix of $T_{\overline{\omega}}$ is

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

We can see that column 4 and 5 are the same. And rank $(T_{\overline{\omega}}) = 4$.

We are now in a position to give the correct theorem and its proof.

Theorem 4.5. For any 2-by-m LHCA, let $f(m)$ be the number of distinct m-cell substate vectors which produce $2^{2m}-1$ transitions for 2-by-m LHCA with a maximum length cycle. We have

$$f(m) = 2f(m-2)+6f(m-3)+2h(m) \quad m>3 \tag{4.4}$$

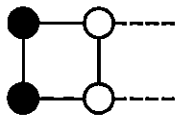
$$h(m) = f(m-2)+f(m-4)-3f(m-5)-h(m-2) \quad m>5 \tag{4.5}$$

where we define the base cases for both $f(m)$ and $h(m)$ by:

$$f(0)=1,\ f(1)=2,\ f(2)=4$$
$$h(0)=h(1)=h(2)=0,\ h(3)=1,\ h(4)=4.$$

Proof: The key to the proof is to determine how many combinations that we can select for ω in 2-by-m LHCA such that $|\ ps(w)| = m$. And then apply the rules we derived above. Before the proof, we define the base case for ($m = 0,\ 1,\ 2$) as indicated above. The following are proved case by case.

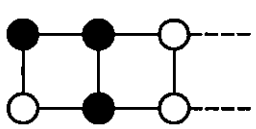
Case 1: Both cells in the first column are selected, then two cells of the second column are their partners. As there is no possibility to have "23" or "32" in these two columns, so the number of combinations in this case is



$$a(m) = f(m-2).$$

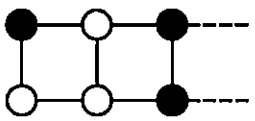
If only one of the cells in the first column is selected (say the top one), then there exists four cases to be considered for the second column:

Case 2: Both cells in the second column are selected, then neither of the cells in the third column can be selected, that is because they must be the partners for the cells in the second column. Below is the structure. There is no possibility to have "23" or "32" in these three columns, so the number in this case is



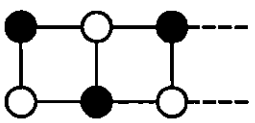
$$b(m) = f(m-3).$$

Case 3: Both cells in the second column are **NOT** selected, then both cells in the third column must be used. Below is the structure. And as same as case 2, the number in this case is



$$c(m) = f(m-3).$$

Case 4: Select the cell which is **NOT** the same row as the cell selected in the first column is selected. In this case, only one situation meets the rule 1, which is the third column is "22". so the structure is "232". Then the number is



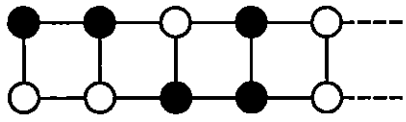
$$d(m) = f(m-3).$$

Case 5: Select the cell which is the same row as the cell selected in the first column is selected. In this case, we have structure beginning with "22", and the number of all combinations is

$$h_1(m) = f(m-2).$$

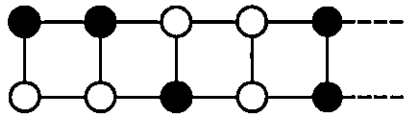
The following structures list all the possibilities that begin with "223" and fit the structure, which should be avoided, described in the rule. The right side shows the structure

notation and the formulas that calculate the numbers for the corresponding structure:



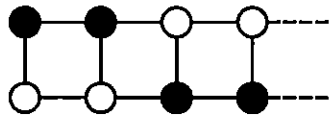
Structure is "22341 $\cdots$".

$$h_2(m) = b(m-2) = f(m-5) \quad (1)$$



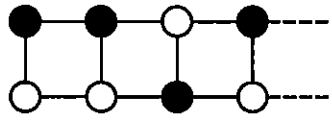
Structure is "22314 $\cdots$".

$$h_3(m) = c(m-2) = f(m-5) \quad (2)$$



Structure is "2233 $\cdots$".

$$h_4(m) = h(m-2) \quad (3)$$



Structure is "2232 $\cdots$".

$$h_5(m) = d(m-2) - f(m-4)$$
$$= f(m-5) - f(m-4) \quad (4)$$

The total number in case 5 is

$$\begin{aligned} h(m) &= h_1(m) - h_2(m) - h_3(m) - h_4(m) - h_5(m) \\ &= f(m-2) - f(m-5) - f(m-5) - h(m-2) - f(m-5) + f(m-4) \\ &= f(m-2) + f(m-4) - 3f(m-5) - h(m-2) \end{aligned}$$

As 2-by-m LHCA is symmetric, the overall number is

$$\begin{aligned} f(m) &= 2(a(m) + b(m) + c(m) + d(m) + h(m)) \\ &= 2f(m-2) + 6f(m-3) + 2h(m) \end{aligned}$$

□

Use the recursive structure of $f(m)$ that we proved in the theorem, it is straightforward to find the number for 2-by-3 to 2-by-7 LHCA.

$$
\begin{aligned}
f(3) &= 2f(1) + 6f(0) + 2h(3) = 4 + 6 + 2 = 12 \\
f(4) &= 2f(2) + 6f(1) + 2h(4) = 8 + 12 + 8 = 28 \\
h(5) &= f(3) + f(1) - 3f(0) - h(3) = 12 + 2 - 3 - 1 = 10 \\
f(5) &= 2f(3) + 6f(2) + 2h(5) = 24 + 24 + 20 = 68 \\
h(6) &= f(4) + f(2) - 3f(1) - h(4) = 28 + 4 - 6 - 4 = 22 \\
f(6) &= 2f(4) + 6f(3) + 2h(6) = 56 + 72 + 44 = 172 \\
h(7) &= f(5) + f(3) - 3f(2) - h(5) = 68 + 12 - 12 - 10 = 58 \\
f(7) &= 2f(5) + 6f(4) + 2h(7) = 136 + 168 + 116 = 420
\end{aligned}
$$

Comparing the results with those shown in table 4.1 and the new theorem's calculation above, we find they agree with each other.

According to theorem 4.5, we can calculate the number of distinct m-cell substate vectors which produce $(2^{2m} - 1)$ transitions for the 2-by-m LHCA with a maximum length cycle. In [25], it proves a theorem to calculate the number for m-cell LHCA. Using both theorems, we have table 4.3 showing the comparison. The comparison is based on the same number of cells for the two kinds of LHCA. For example, we compare the numbers for 1D 6-cell LHCA and 2D 2-by-3 LHCA. They have 8 and 12 substate vectors which produce maximum number of transitions respectively. And the ratio between the two numbers is 1.5. We compare another example, which are 1D 48-cell LHCA and 2D 2-by-24 LHCA. They have 16,777,216 and 1,959,771,916 separately. And their ratio is 116.8. From the table, we see that the ratio is getting large as the cell number increases, which means 2D LHCA have more transitions than the corresponding 1D LHCA. Just a note that as no maximum length cycle exist for 2-by-2 and 2-by-4 LHCA, we don't list the numbers for these two LHCA.

n-cell (2m)	m-cell	1D LHCA (b)	2D LHCA 2-by-m (c)	Ratio (c/b)
6	3	8	12	1.5
10	5	32	68	2.1
12	6	64	172	2.7
14	7	128	420	3.3
16	8	256	1036	4.0
18	9	512	2564	5.0
20	10	1024	6316	6.2
22	11	2048	15588	7.6
24	12	4096	38476	9.4
26	13	8192	94916	11.6
28	14	16384	234220	14.3
30	15	32768	577956	17.6
32	16	65536	1426060	21.8
34	17	131072	3518852	26.8
36	18	262144	8682796	33.1
38	19	524288	21424740	40.9
40	20	1048576	52865740	50.4
42	21	2097152	130446404	62.2
44	22	4194304	321896844	76.7
46	23	8388608	794232612	94.7
48	24	16777216	1959771916	116.8

Table 4.3. *Comparison of two LHCA giving the number of m-cell substate vectors which produce* $2^{2m} - 1$ *transitions*

4.3 Discussion and Conclusion

In this chapter, we have presented another important characteristic of LHCA, which is transition properties. We focus on investigating transitions of substate vectors for their 2D LHCA. A theorem presented in [4] calculates the number of distinct substate vectors which produce maximum transitions for 2-by-m LHCA. We find there are mistakes in the statement. We check the formulae in the theorem and ascertain the nature of the problem. Then we derive and prove a corrected theorem. This theorem is consistent with the simulation results. The derivation of this theorem is to investigate whether 2D LHCA has more transitions than 1D LHCA. Fortunately the correct results still show us that 2D LHCA do have more transitions than 1D LHCA especially as the number of cells increases. This result may be due to the richer connection of 2D LHCA than 1D LHCA.

Transition properties are important when detecting delay faults in VLSI testing. We anticipate that using 2D LHCA as test pattern generator (TPG) will have higher fault coverage than 1D LHCA. In chapter 5, we present relative background for VLSI testing. In chapter 6, we give the experimental results.

Chapter 5

Test Pattern Generators and Delay Faults in BIST

This chapter presents the necessary background for testing applications. The testing process is shown at the beginning followed by a description of the critical part of the scheme - the test pattern generator(TPG). Then concepts of the delay fault model in combinational circuits are introduced. To do fault simulation, fault equivalence rules for delay faults are applied and an efficient way(PPSFP) is presented.

5.1 TPG in Built-In Self-Test

5.1.1 Built-In Self-Test

An integrated circuit such as a chip may behave incorrectly for various reasons: an error in the design(electrical or logical), a defect because the manufacturing process is imperfect, or aging of the components. The purpose of the testing is to verify that there is no error in the circuit.

As the field of VLSI (very large scale integration) has undergone rapid growth during the last decade, it is possible to put a large number of gates on a single chip. The complexity of the circuit, however, puts a heavy burden on testing of manufactured chips, which is very important to detect a fault (or an error) as early as possible, because test effectiveness and high fault coverage at IC level can save a lot of money at the board level and at the system level. Research has attempted to solve this problem by concluding that the best way is to incorporate design for testability features into the chip when the VLSI circuit is designed.

One of the most well-known and widely used techniques is Built-In Self-Test (BIST). This technique can be used to test large and complex systems. The benefit of a BIST implementation is not only characterized by the test length and the low hardware overhead required to achieve complete or sufficiently high fault coverage, but its small price to pay as well.

Figure 5.1 represents an approach to the testing process. First, a test pattern is applied to the circuit under test (CUT) with a defined initial state or exercise some functionality.

Then the test pattern is processed by the circuit, and at last the response, the output of the CUT is checked by a test evaluator.

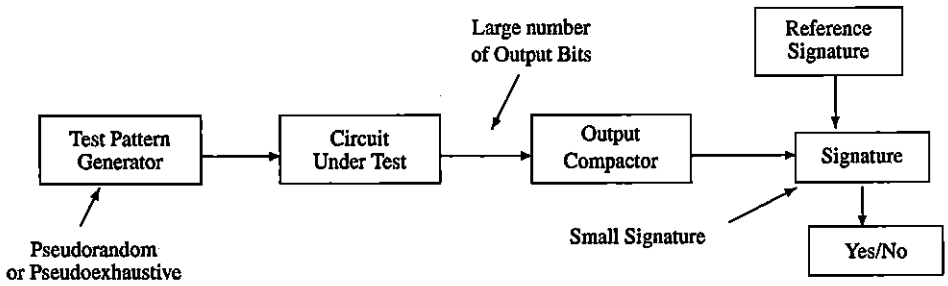


Figure 5.1. *A Testing Scheme.*

The test patterns can be either pseudorandom or pseudoexhaustive. A pseudorandom test set has some important properties of a random sequence, while being totally predictable and repeatable. An exhaustive test is when all possible test patterns are applied, which is impractical for a complete VLSI chip. Using a partitioning method to produce a sum of exhaustive tests to apply progressively to all parts of the circuit is called a pseudoexhaustive test.

To verify whether the CUT is functioning properly the large number of output bits must be compared with known correct ones. Using a direct comparison of the complete response data with the reference values results in a huge hardware overhead, if applied in on-chip testing. The common method to solve this is to use output data compaction, which is implemented by a division process. The remainder resulting from this process is called the "signature". The output compactor enables a reduction of the original response data stream to a small signature. And this signature is compared with a reference signature.

In a circuit with BIST, both test patterns generation and output response analysis are built on chip. In this thesis, we focus on the TPG in BIST.

5.1.2 Test Pattern Generator (TPG)

Test pattern generation is the most crucial problem in VLSI testing especially when specifically considering logic faults. A deterministic test pattern generator (TPG), either pseudorandom or pseudoexhaustive, aims at testing all faults in a prescribed set; it may be automated (Automated Test Pattern Generators), but even for the single stuck-at fault model (a simple model which does not cover all the possible defects), the automated test generation is very long and costly for large circuits. Because of this, test logic may be added in VLSI design to enhance the ability to achieve a high quality metric, to ease the ability to generate vectors, to reduce the time involved with vector generation, or to reduce the cost involved with the application [7]. This is generally referred to as Design for Test (DFT).

Test patterns can be generated using several approaches. Pseudorandom test vectors are commonly used in testing logic circuits [1]. This is one way that only requires a low area overhead while still providing acceptable fault coverage. It can be done by an autonomous LFSR with no input. This is implemented by feeding back selected bits from a shift register through an exclusive-OR function. Another alternative way is using linear hybrid cellular automata (LHCA). As both techniques can generate maximum test cycle lengths, the outputs are connected to multiplexors, which allow the inputs to the circuit to be switched between normal operation and test mode. Fault coverage is determined by fault simulation. We give the definition of fault coverage below. Pseudorandom testing can be applied to both combinational and sequential circuits. In this thesis, we only investigate combinational circuits.

5.2 The Delay Fault Model in Combinational Circuits

5.2.1 Fault Model

In digital circuits, there may exist one or more faults caused by the process of manufacturing. The faults may lead to erroneous behaviors and produce different output response from

the fault-free circuits. Fault modelling is a basic idea in digital systems testing design. It is extremely useful and practical that we are able to predict the effect of faults in integrated circuits.

To make a further simplification in fault modelling, we focus only on the main character of faults and use the single-fault assumption. This assumption states that there is only one fault present in a system when it is tested. This assumption is based on a so-called frequency of testing strategy. This assumes that the interval between two consecutive tests is small enough so that the probability of multiple independent faults occurring is sufficiently small. In a real system, this appears to be a reasonable assumptions. Even though the single-fault assumption has some limitations, it is still widely applied to the area of the study of logic faults. And it sufficiently reduces the complexity of fault research and analysis.

There are several classes of faults: stuck-at faults, delay faults, stuck-open faults. In this thesis, we only consider delay fault, specifically gate delay faults, which are also called delay faults.

5.2.2 The Delay Fault Model

A delay fault, also named a gate delay fault, is considered as a logical model for a defect that delays either a rising or a falling transition. There are two types of delay faults for the two transitions. One is slow-to-rise, the other is slow-to fall.

Definition 5.1 Let $x_1, x_2, \cdots, x_n$ be the correct expected bit sequence for a line of a circuit over some time period. The delay faults yield the sequence $y_i, 0 < i \leq n$, defined as below:

1) slow-to-rise

$$y_i = \begin{cases} 1 & \text{if } x_i = x_{i-1} = 1, 1 < i \leq n; \\ 0 & \text{otherwise.} \end{cases}$$

2) slow-to-fall

$$y_i = \begin{cases} 0 & \text{if } x_i = x_{i-1} = 0, 1 < i \leq n; \\ 1 & \text{otherwise.} \end{cases}$$

where $x_0 = x_1$.

The delay faults reside on the inputs and outputs of logic gates. The slow-to-rise delay fault behaves as a stuck-at-zero fault temporarily. Likewise, the slow-to-fall delay fault corresponds to stuck-at-one fault. Figure 5.2 shows an AND gate and the effects of the delay faults. Within a time period measured, the delay fault can be found.

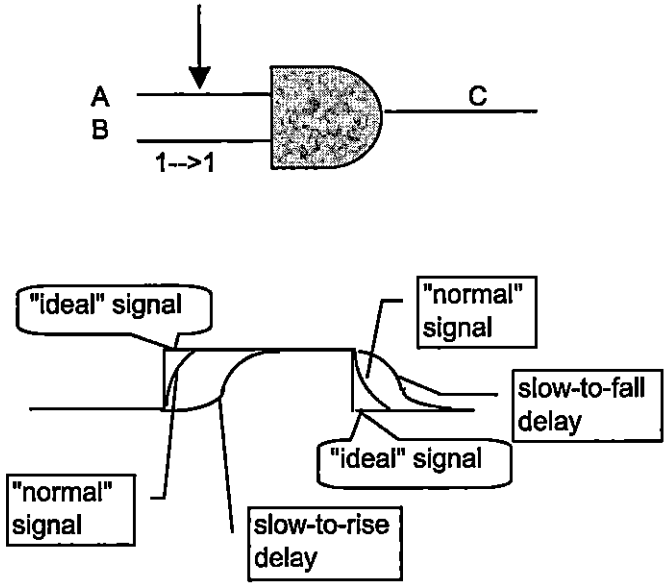


Figure 5.2. *Delay Fault*

As each line of the circuit may have either slow-to-rise or slow-to-fall delay faults, a circuit with p lines has $2p$ possible single delay faults.

Given the definition of a delay fault, a test must create appropriate transitions at the line of the fault and propagate the effect of the transition to at least one circuit output. In general, two patterns(vectors) are required, initialization and transition propagation. The initialization vector places the initial transition value at the line having the fault. The transition propagation vector places the final value at the line and propagates the effect to a

primary output. When a pair of vectors is applied, four possible sequences are 00, 01, 10, 11. The sequence 01 is used to detect slow-to-rise since it changes 01 to 00 within a certain period. Likewise, sequence 10 is used to detect slow-to-fall since it changes 10 to 11 temporarily.

5.3 Fault Simulation

5.3.1 Fault Equivalence Rules for Delay Faults

Fault simulation is to simulate a circuit in the presence of the relevant faults. A list of delay faults needs to be generated to be simulated. For faults grouped in a fault equivalence class, a test for any fault implies a test for all faults in the class. Delay fault equivalence rules have been discussed in [19] and [24].

Delay faults are considered equivalent only if one of the following two conditions holds: [19]

- If a gate has one input, then the input delay faults are equivalent to the output delay faults. In figure 5.3 input slow-to-rise is equivalent to output slow-to-fall.

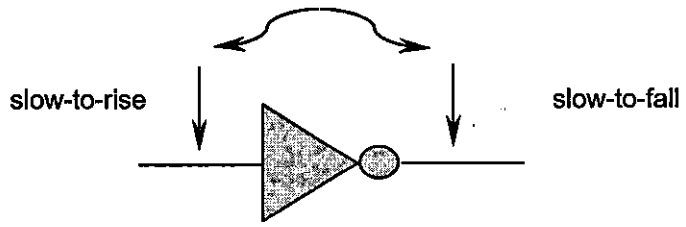


Figure 5.3. *Fault equivalence rules (1)*

- If a gate has only one fanout, then the output delay faults are equivalent to the input delay faults on the fanout gate. See figure 5.4

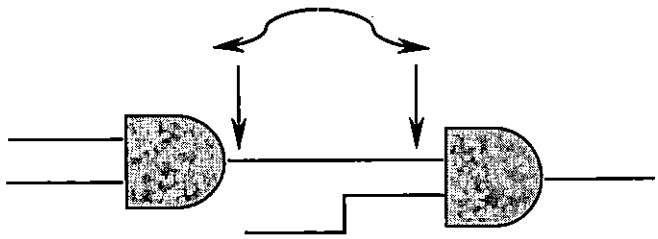


Figure 5.4. *Fault equivalent rules (2)*

In [24], the following faults are removed since they have been proven to be equivalent to gate input delay faults:

- slow-to-rise and slow-to-fall faults for the NOT gate output
- slow-to-rise faults for NOR and AND gate outputs
- slow-to-fall faults for NAND and OR gate outputs

5.3.2 Fault Simulation Strategy

A delay fault is considered detectable by a pattern if a transition occurs at the point of the fault and the corresponding stuck fault is detected in a timed measure. The delay fault simulator is just a stuck at fault simulator to identify logic gates that experience a transition in a pattern relative to the preceding pattern.

PPSFP(Parallel Pattern Single Fault Propagation) fault simulation is an efficient way to simulate delay faults [18]. It combines the concepts of parallel pattern evaluation, which is used to evaluate the binary signal values in the full length of a machine word, and single fault propagation, which is used to minimize the reevaluation for faulty circuit simulation, to perform a simulation. First it does a parallel fault-free simulation of a group of W vectors. Then the remaining undetected faults are serially injected and fault values are computed in parallel for the same set of vectors. Comparisons between good and faulty values involve W bits. The propagation of fault effects continues as long as faulty values are different from the good values in at least one vector. Detected faults are dropped and

the above steps are repeated until all vectors are simulated or all faults are detected. The simulation involves two values, zero delay, and 256 patterns per pass. Figure 5.5 describes the PPSFP delay fault simulation algorithm.

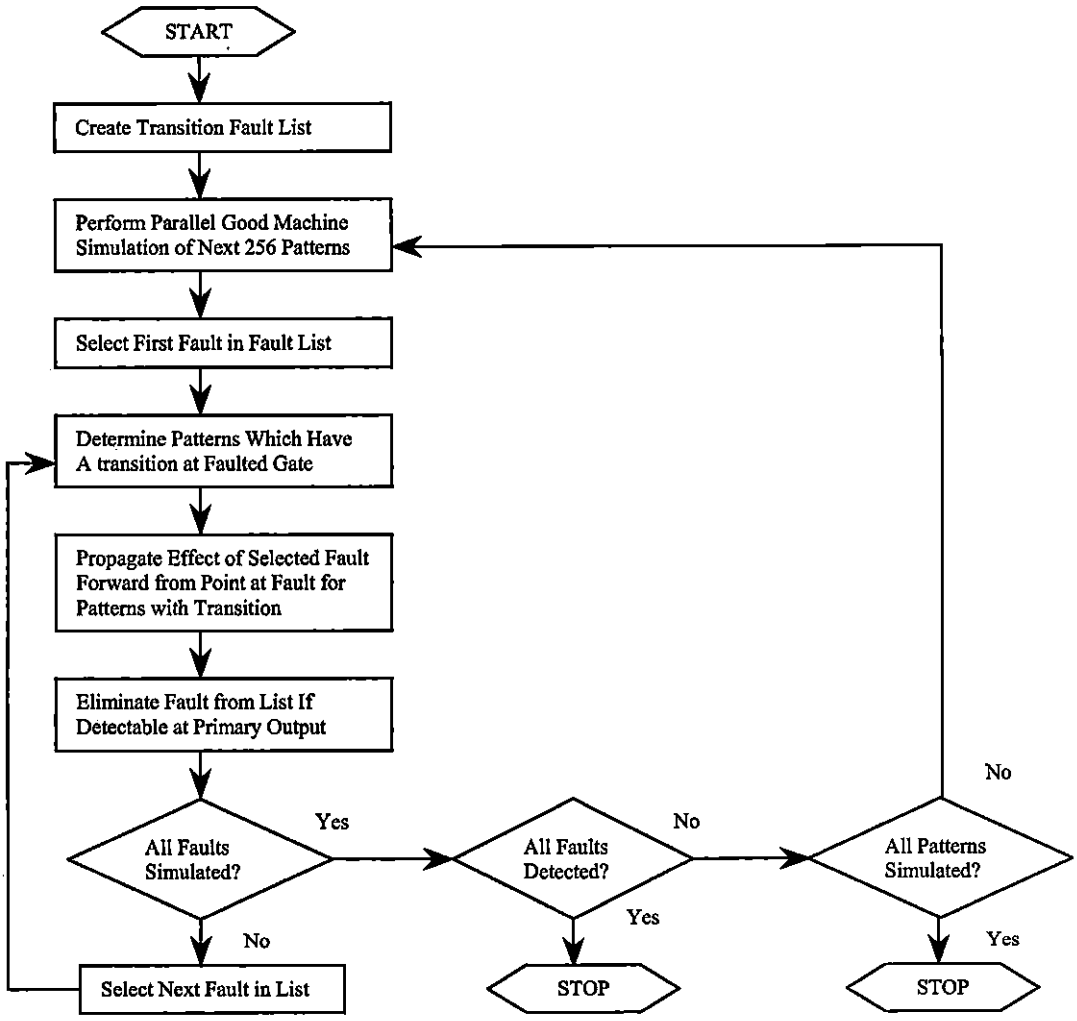


Figure 5.5. *Algorithm for PPSFP Delay Fault Simulation.*

5.3.3 Fault Coverage

Fault coverage is the ratio between the number of faults detected by the test sequence and the total number of faults in the assumed fault set F. Fault coverage is defined formally in the sequel. Given a circuit C and a set F of faults under consideration, fault coverage is a measurement of the effectiveness of a test sequence S.

$$\text{Fault Coverage} = \frac{(N-U)}{N} \times 100$$

where N is the number of faults in F, U is the number of undetected faults.

5.4 Summary

This chapter provides necessary background for the testing experiments, which are contained in the next chapter. Based on the theory of 2-D LHCA described in the previous several chapters with their transition properties, we investigate as to how well the 2D LHCA works as a pseudorandom test pattern generator for built-in self-test, in the next chapter.

Chapter 6

Fault Simulation for 2D LHCA as Test Pattern Generators

The theoretical analysis in chapter 4 gives us an overview about the transition properties for the 2-by-m LHCA. In this chapter, we use simulation experiments on the ISCAS85 benchmark circuits to observe the effectiveness of test vectors generated by 2-by-m LHCA.

6.1 Previous Work

Some previous simulation studies have been undertaken by the VLSI group at the University of Victoria. A fault is detected if the output data stream in the presence of the fault differs from the output data stream in the fault-free case.

In [25], Zhang provides simulation studies of the ISCAS85 benchmark circuits to observe the effectiveness of test vectors generated by LHCA and LFSR. The experiments are based on delay fault (a single slow-ro-rise and slow-to-fall gate delay fault) simulation applying comprehensive delay fault equivalence rules. The results are collected by performing the fault simulation using test sequences of length 102,000 with 100 different random connections between the circuit inputs and the test vector generator outputs for each circuit. The results in [25] shows that an LHCA has more potential to achieve a higher fault coverage than LFSR for stimulating faults requiring a pair of vectors.

Cattell in [4] does the same simulation experiments, but using 2-by-n LHCA instead of one dimensional LHCA as test pattern generators. The results provides evidence to support the assertion that 2-by-n LHCA perform slightly better than LHCA, which may be due to the richer interconnection pattern of 2-by-n LHCA compared to 1D LHCA. Our new experiments are also based on the simulation program, but we undertake a much wider range of experiments, with a more extensive set of initial states, and a comprehensive evaluation of the fault coverage based on the specific lengths of the input stream. We also present the statistical analysis for the fault coverage results such as maximum, minimum and range.

Circuit Name	Input	Output	Total Delay Faults
c432nr.isc	36	7	658
c3540nr.isc	50	22	4290
c880.isc	60	26	1305
c6288nr.isc	32	32	9987

Table 6.1. *Circuits Information*

6.2 New Experimental Results

Since the number of transitions is an important issue regarding detecting faults with sequential behavior, we use the simulation program called **sim3**, which was written by Zhang in 1993. Its implementation is based on the Parallel Pattern Single Fault Propagation(PPSFP) technique discussed in chapter (5.3.2) and the comprehensive fault equivalent rules for fault collapsing are applied.

Besides the existing test pattern generators in **sim3**, we add one more generator, which is a minimal-cost maximum length 2-by-m LHCA, do the fault simulation and compare the results with one dimensional LHCA. The studies are also based on the ISCAS85 benchmark combinational circuits.

Two distinct experiments are undertaken.

Experiment 1: Given 30 different initial states, using 1D LHCA or 2D LHCA as test pattern generators separately to simulate delay faults with increasing state length. We test four circuits here. The circuits information is listed in table 6.1. Part of the results are shown in table 6.2 (All of the detail tables are listed in appendix C.) We select two circuits. One is presented in table and the other is used graphical analysis. In table 6.2, 'Mean' is the average fault coverage among 30 tests for the circuit. 'Max' is the maximal fault coverage within 30 tests and 'Min' is the minimal. 'Range' is the difference between 'Max' and 'Min'.

Some observations can be derived from the table and the graphs:

Length	Mean		Max(A)		Min(B)		Range(A-B)	
	1D	2D	1D	2D	1D	2D	1D	2D
256	94.23	93.90	96.81	96.66	92.71	91.64	4.1	5.02
512	97.35	97.44	99.09	98.78	95.9	94.83	3.19	3.95
768	98.33	98.31	99.39	99.09	97.11	96.96	2.28	2.13
1024	98.65	98.68	99.54	99.24	97.42	97.72	2.12	1.52
2048	99.08	99.33	99.85	99.7	98.02	98.33	1.83	1.37
3072	99.35	99.53	99.85	100	98.63	98.63	1.22	1.37
4096	99.48	99.71	99.85	100	99.09	99.39	0.76	0.61
5120	99.60	99.77	99.85	100	99.24	99.39	0.61	0.61
6144	99.65	99.86	99.85	100	99.24	99.7	0.61	0.3
7168	99.70	99.89	99.85	100	99.54	99.7	0.31	0.3
8192	99.72	99.91	99.85	100	99.54	99.7	0.31	0.3
9216	99.73	99.93	99.85	100	99.54	99.7	0.31	0.3
10240	99.77	99.94	99.85	100	99.54	99.7	0.31	0.3
16384	99.81	100.00	99.85	100	99.7	100	0.15	0

Table 6.2. *A summary of fault coverage vs. sequence length (%)(c432nr)*

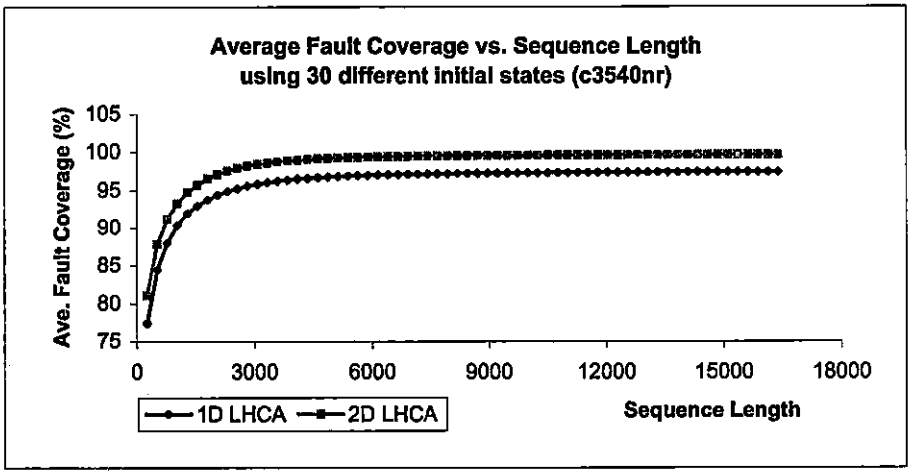


Figure 6.1. *Delay Fault Simulation Results(c3540nr-1)*

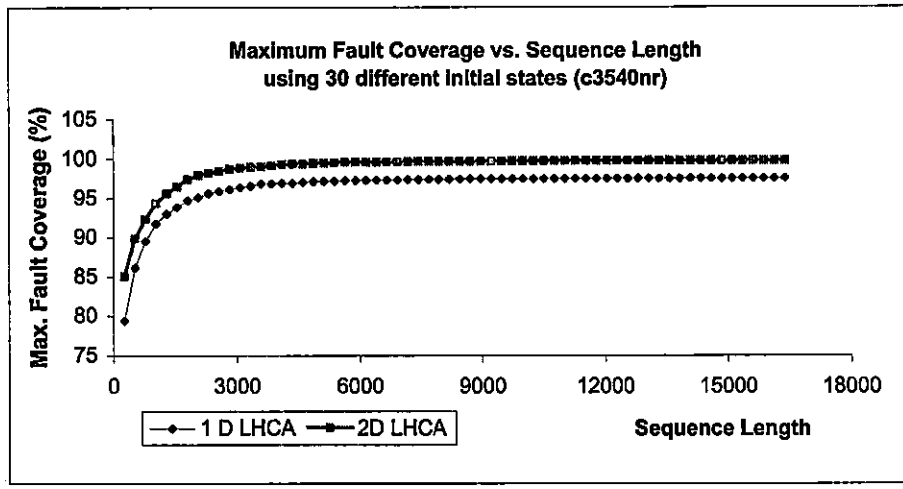


Figure 6.2. *Delay Fault Simulation Results(c3540nr-2)*

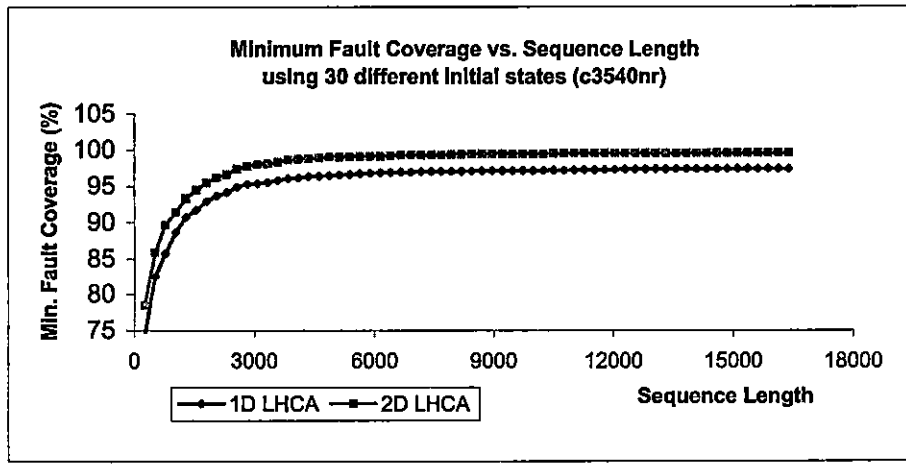


Figure 6.3. *Delay Fault Simulation Results(c3540nr-3)*

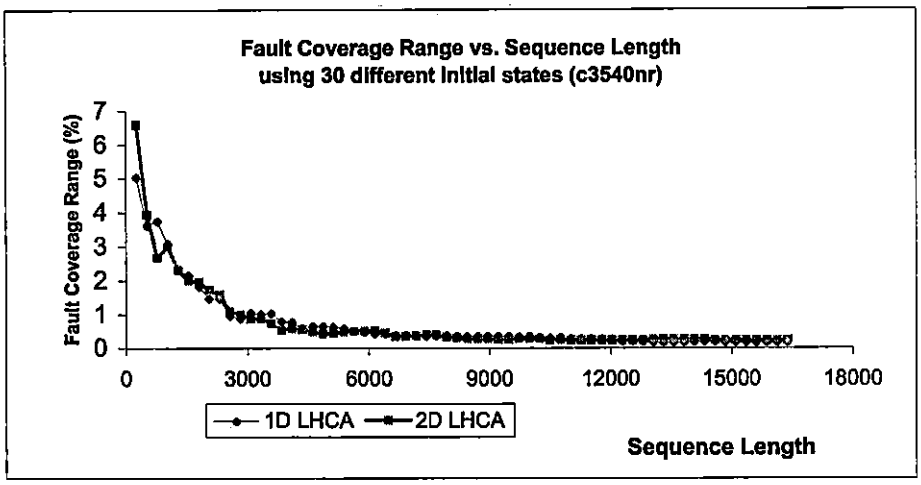


Figure 6.4. *Delay Fault Simulation Results(c3540nr-4)*

- For either 1D or 2D LHCA, when the length increases, the fault coverage increases. Because the more test patterns are running, the more faults in the fault list can be detected. We can see that 2D LHCA starts off faster and levels off at a slightly higher fault coverage than 1D LHCA.
- When the test vectors increase beyond a certain value, the fault coverage of both LHCA increases slowly, and is close to being stable. And for some circuits, the 2D LHCA reach 100 % fault coverage while 1D LHCA do not. This illustrates that all delay faults can be detected by using 2D LHCA as TPG, while there are still a few that can't be detected by the 1D LHCA generator.
- In the circuits we test, it can be seen that with same vector length, and the same initial state, most of results shows that 2D LHCA leads to slightly higher fault coverage as compared with the 1D LHCA.
- To get the same fault coverage, 2D LHCA need less vectors than 1D LHCA. For example, in c432nr.isc, 2D LHCA need 5120 vectors to reach 99.77 % fault coverage, while 1D LHCA need 10240 vectors.
- For both 1D LHCA and 2D LHCA, different initial states given to the circuits may

lead to different fault coverage. The range between maximal fault coverage and minimum fault coverage is large for small sequence lengths but becomes very small for large sequence lengths. We can't see any obvious difference in this aspect between 1D LHCA and 2D LHCA.

Experiment 2: We investigate the ISCAS'85 benchmark combinational with both non-redundant and redundant circuits using 1D LHCA and 2D LHCA as test pattern generators separately. Given 30 different initial states, we simulate delay faults with three vector lengths (4096, 8192, 16384) and compare the average fault coverage.

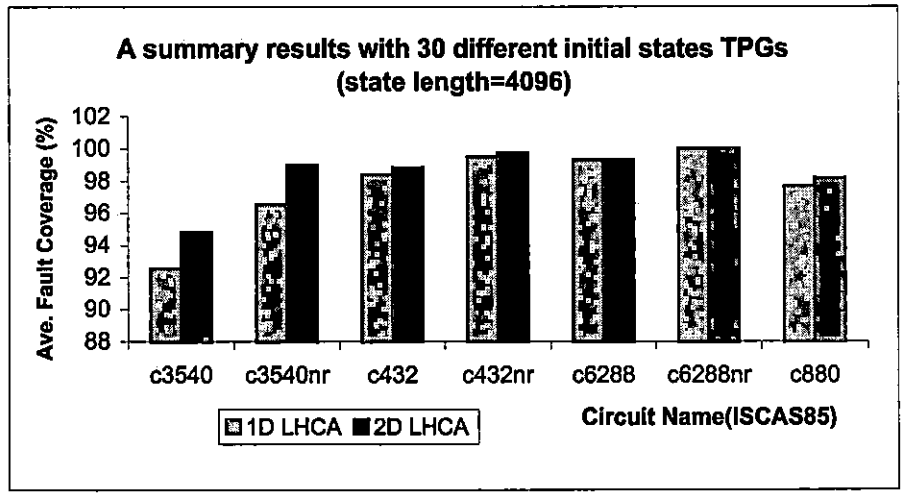


Figure 6.5. *Delay Fault Simulation Results (ex2-1)*

Observations:

From the three graphs with three different lengths, it seems that 2-by-n LHCA give slightly higher average fault coverage than 1D LHCA in delay fault simulation experiments. We hypothesize this may be due to the richer interconnection pattern of 2-by-m LHCA compared to 1D LHCA.

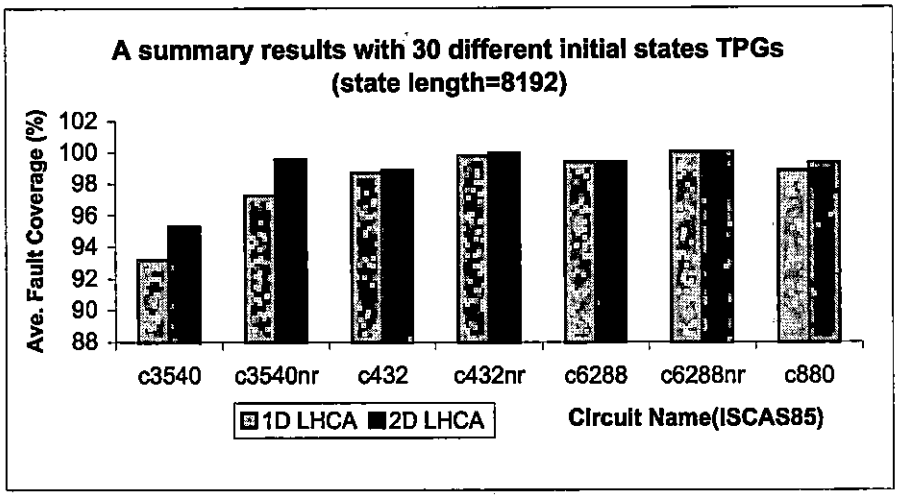


Figure 6.6. *Delay Fault Simulation Results (ex2-2)*

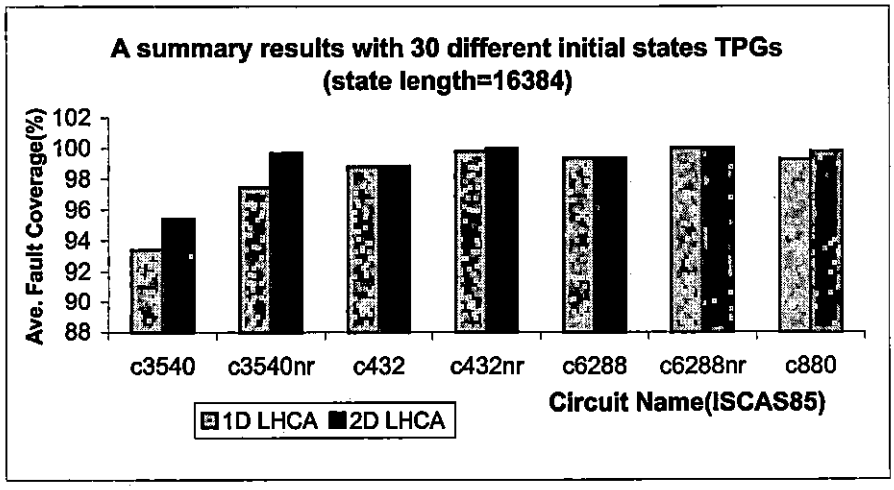


Figure 6.7. *Delay Fault Simulation Results (ex2-3)*

6.3 Discussion and Conclusion

A theoretical analysis of the two vector transition properties of 2-by-m LHCA in chapter 4 shows that they are potentially superior on average to 1D LHCA. We test this observation using simulation experiments on the ISCAS85 benchmark circuits. We look at the detection of fault coverage regarding to delay faults and compare the result with 1D LHCA. This is tested using a set of standard benchmark circuits. As we anticipated in chapter 4, better transition properties might lead to better performance for testing faults with sequential behaviour. The experimental results show that, on average, 2D LHCA provides slightly higher fault coverage which performing better than 1D LHCA in the application. The improved performance is probably a direct result of the slightly richer interconnection structure of a 2-by-n LHCA as compared to a 1D LHCA.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis, we investigate maximum length 2D LHCA with regular configurations. As it is a linear finite state machine, we can use the transition matrix in its analysis and to generate the next state vector. The transition matrix structure of n-by-m LHCA is derived. Since the characteristic polynomial of the transition matrix can determine some properties of the CA, for example, the characteristic polynomial(CP) is primitive if and only if the CA has a maximal length cycle, we focus on the calculation of the CP. As the dimension of the matrix is very large for n-by-m LHCA, two ways to compute the CP are presented in [4] for 2-by-n CA. One is deriving the recurrence relation and the other is using matrix formulation. We follow a similar approach to derive formulas for 3-by-n CA. Though the approaches are still feasible, it turns out to be unreasonably complex. So we can't continue this approach for n-by-m ($n \geq 4$) LHCA.

We are also interested in the transition properties of 2-by-m LHCA. Since the number of transitions is an important issue regarding detecting faults in VLSI testing applications with sequential behavior. For any n-cell LHCA, it is not enough to just consider the transitions of n-cell vectors, but also the transitions of k-cell substate vectors for $k \leq n$ are necessary to be looked at. As it has been proved in [25] that for a k-cell substate vector (k=n/2) of n-cell vector generator with a maximum length cycle, the upper bound of distinct transitions is $2^n - 1$. Thus we calculate the number of different k-cell substate vectors, which have 2^{2k} transition capability for both 1D LHCA and 2D LHCA. The result showed in Table 4.3 illustrates that 2D LHCA have more transitions than 1D LHCA. The difference increases as the number of cells increases.

As the theoretical results that the 2D LHCA indicate much more transitions than 1D LHCA, it was hoped that 2D LHCA might perform much better as test pattern generators in VLSI testing. We did some experiments based on a single slow-to-rise and slow-to-fall gate delay fault model. The experiments apply comprehensive delay fault equivalence rules and PPSFP (Parallel Pattern Single Fault Propagation) technique. The circuits we used are

standard ISCAS'85 benchmarks. And the results indicate that for gate delay faults, 2D LHCA have slightly higher fault coverage than 1D LHCA and that for some circuits, 2D LHCA can detect some faults which 1D LHCA can't.

7.2 Future Work

In this thesis, we only analyze the transition properties of 2-by-m 2D LHCA. The results show that using 2-by-m LHCA as test pattern generators give slightly higher fault coverage than 1D LHCA. This conclusion is based on our theoretical analysis and some limited experiments. Similar work is suggested for n-by-m LHCA ($n \geq 3$). And it is better to have more analysis from other aspects to provide more evidence supporting our conclusions. As we are interested in the minimal-cost LHCA with maximum-length cycle, table 3.2 shows the LHCA with small values of n and m. More work can be done to get higher degree LHCA with large n and m.

The LHCA we investigate in the thesis are all fully connected to their neighborhood. When we connect the cells which are not edge ones to the circuit under test (CUT), the interconnect lines will cross so that it is hard to get a planar representation of the circuit layout. One way to solve this is to choose the edge cells to be inputs for the test vectors. It is recommended to do some research on the properties of subvectors. At the same time, we could expand the analysis of 2D LHCA to a more flexible structure, which means the next state functions can be from some subset of the neighbors.

Future work is suggested below:

- Generate the minimal cost n-by-m 2D LHCA with maximum length cycle with larger values of n and m and observe the results.
- Investigate the characteristics of randomness of 2D LHCA and compare the results with 1D LHCA.
- Develop the transition properties of n-by-m LHCA ($n, m \geq 3$).

- Derive the properties of the substate vectors based on cells that are edge ones and compare the results with 1D LHCA.
- Instead of connecting to all the four neighbor cells, connect to some subset of neighbors and investigate their properties.

Bibliography

[1] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*. IEEE Press, 1990.

[2] P. H. Bardell, W. H. McAnney, and J. Savir, *Built-In Test For VLSI Pseudorandom Techniques*. John Wiley and Sons, Inc., 1987.

[3] K. Cattell and J. C. Muzio, "Analysis of one-dimensional linear hybrid cellular automata over GF(q)," *IEEE Transactions on Computers*, vol. 45, no. 7, pp. 782–792, July 1996.

[4] K. Cattell, S. Zhang, M. Serra, and J. C. Muzio, "2-by-n hybrid cellular automata with regular configuration: Theory and application," *IEEE Transactions on Computers*, vol. 48, no. 3, pp. 285–295, March 1999.

[5] K. Cattell, S. Zhang, X. Sun, M. Serra, J. C. Muzio, and D. M. Miller, "One-dimensional linear hybrid cellular automata: Their synthesis, properties and applications in VLSI testing," Available at: http://www.csr.uvic.ca/ mserra/CA.html.

[6] T. Cormen, C.E.Leiserson, and R.L.Rivest, *Introduction to Algorithms*. The MIT Press, 1990.

[7] A. Crouch, *Design-for-Test for Digital IC's and Embedded Core Systems*. Prentice Hall PTR, 1999.

[8] K. Furuya and E. J. McCluskey, "Two-pattern test capabilities of autonomous TPG circuits," *Proceeding IEEE International Test Conferrence*, pp. 704–711, 1991.

[9] P. D. Hortensius, "Parallel computation of non-deterministic algorithms in VLSI," Ph.D. dissertation, University of Manitoba, Manitoba, BC, Canada, 1997.

[10] P. D. Hortensius, R. D. McLeod, and H. C. Card, "Cellular automata-based signature analysis for built-in self-test," *IEEE Tansaction on Computers*, vol. 39, pp. 1273–1283, 1990.

[11] P. D. Hortensius, R. D. McLeod, W. Pries, D. M. Miller, and H. C. Card, "Cellular aautomata-based pseudorandom number generators for built-in self-test," *IEEE Tansaction on Computer-Aided Design*, vol. 8, pp. 842–859, 1989.

[12] H. Janoowalla, "Analysis of maximum length linear two-dimensional cellular au-

tomata," Master's thesis, Department of Computer Science. The University of Victoria, Victoria, BC, Canada, May 1992.

[13] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*. Cambridge University Press, 1986.

[14] W. Pries, A. Thanailakis, and H. Card, "Group properties of cellular automata and VLSI applications," *IEEE Transactions on Computers*, vol. C, no. 35, pp. 1013–1024, 1986.

[15] H. S. Stone, *Discrete Mathematical Structures And Their Applications*. Science Research Associates, Inc, 1973.

[16] P. Tsalides, "Cellular automata-based built-in self-test structures for VLSI systems," *IEEE Electronics Letters*, no. 26, pp. 1350–1352, 1990.

[17] P. Tsalides, T. A. York, and A. Thanailakis, "Pseudorandom number generators for systems based on linear cellular automata," *IEEE Proceedings of Part E: Computers and Digital Techniques*, no. 138, pp. 241–249, 1991.

[18] J. A. Waicukauski, E. B. Eichelbergér, D. O. Forlenza, E. Lindbloom, T. McCarthy, and IBM, "A statistical calculation of fault detection probabilities by fast fault simulation," *IEEE 1985 International Test conference*, no. 21, pp. 779–783, 1985.

[19] J. A. Waicukauski, E. Lindbloom, B. K. Rosen, V. S. Iyengar, and IBM, "Transition fault simulation," *IEEE Desing and Test*, pp. 32–38, 1987.

[20] Wikipedia, "Finite field," Available at: http://www.wikipedia.com/wiki/Finite_field, June 2002.

[21] S. Wolfram, "Statistical mechanics of cellular automata," *Rev. of Modern Physics*, pp. 55: 601–644, 1983.

[22] S. Wolfram, "Universality and complexity in cellular automata," *Physica*, pp. 10D:1–35, 1984.

[23] S. Wolfram, "Random sequence generation by cellular automata," *Advances in Applied Mathematics*, pp. 7:123–169, 1986.

[24] S. Zhang, "BIST generators for faults with sequential behaviors," Master's thesis, Department of Computer Science. The University of Victoria, Victoria, BC, Canada, 1993.

[25] S. Zhang, R. Byrne, J. C. Muzio, and D. M. Miller, "Quantitative analysis for linear hybrid cellular automata and LFSR as built-in self-test generators for sequential faults," *Journal of Electronic Testing: Theory and Applications (JETTA)*, pp. 209–221, March 1995.

[26] S. Zhang, R.Byrne, and D. M. Miller, "BIST generators for sequential faults," *Proceedings of IEEE International Conference on Computer Design*, pp. 260–263, 1992.

Appendix A

Program - Calculate Characteristic Polynomial for 3-by-m LHCA

This appendix contains the source code of the Maple program written by Cathy that is used to generate the Characteristic Polynomial of the particular 3-by-m LHCA. The program is based on Lemma 3.2, 3.3, and Theorem 3.4 in this thesis.

```
# Load the necessary libraries.
with(networks):with(linalg): with(combinat,choose):
# Main program starts.
recur:=proc(n)
    local x, y, i, j, k, M;
    # Initialization with base cases given.
    for i from -1 to 0 do
        for j from -1 to 0 do
            for k from -1 to 0 do
                if (k=-1) or (i=-1) or (j=-1) then
                    M[i, j, k]:=0;
                elif (i=0) and (j=0) and (k=0) then
                    M[i, j, k]:=1;
                fi;
```

```
        od;
    od;
od;
```

#Define the cell rule of a 3-by-m LHCA.

#The numbers indicate the rule one cells. The sequence is from up to down and then left to right. To change the cell rule, just change the numbers in "Cone".

```
Cone:=[1,4,5];
for i from 1 to 3*m do
    if member(i, Cone) then x[i]:=1;
        else x[i]:=0; fi;
od;
t:=seq (x [ (i-1)*3+1 ] , i=1..m);
b:=seq (x [ (i-1)*3+2 ] , i=1..m);
d:=seq (x [ (i-1)*3+3 ] , i=1..m);
```

#Start recursive calculation to get characteristic polynomial using loop structures.

```
for k from 1 to m do
    r:= x + t [ k ];  p:= x + b [ k ];  q:= x + d [ k ];
    M[k-1,k-1,k]:=sort(expand(q*M[k-1,k-1,k-1]+M[k-1,k-1,k-2]) mod 2);
    M[k-1,k,k-1]:=sort(expand(p*M[k-1,k-1,k-1]+M[k-1,k-2,k-1]) mod 2);
    M[k,k-1,k-1]:=sort(expand(r*M[k-1,k-1,k-1]+M[k-2,k-1,k-1]) mod 2);
    M[k-1,k,k]:=(p*q+1)*M[k-1,k-1,k-1]+p*M[k-1,k-1,k-2]+q*M[k-1,k-2,k-1]+M[k-1,k-2,k-2];
    M[k-1,k,k]:=sort(expand(M[k-1,k,k]) mod 2);
    M[k,k-1,k]:=(r*q)*M[k-1,k-1,k-1]+r*M[k-1,k-1,k-2]+q*M[k-2,k-1,k-1]+M[k-2,k-1,k-2];
    M[k,k-1,k]:=sort(expand(M[k,k-1,k]) mod 2);
    M[k,k,k-1]:=(r*p+1)*M[k-1,k-1,k-1]+r*M[k-1,k-2,k-1]+p*M[k-2,k-1,k-1]+M[k-2,k-2,k-1];
    M[k,k,k-1]:=sort(expand(M[k,k,k-1])mod 2);
```

```
        M[k,k,k]:=(r*p*q+d[k]+t[k])*M[k-1,k-1,k-1]+(r*p+1)*M[k-1,k-1,k-2]+(p*q+1)*M[k-2,k-1,k-1];

        M[k,k,k]:=M[k,k,k]+r*q*M[k-1,k-2,k-1]+r*M[k-1,k-2,k-2]+p*M[k-2,k-1,k-2];

        M[k,k,k]:=M[k,k,k]+q*M[k-2,k-2,k-1]+M[k-2,k-2,k-2];

        CP:=sort(expand(M[k,k,k]) mod 2);

    od;

    print('The characteristic polynomial of 3-by-m the LHCA is', CP);

end:
```

Appendix B

Program - Generate Minimal-cost n-by-m LHCA with Maximum-length Cycle

This appendix contains the source code of the Maple program written by Cathy that is used to find the minimal-cost maximum-length n-by-m LHCA. The algorithm is presented in section 3.2.3. It computes the characteristic polynomial of the CA and check if it is primitive, if so, the CA has the maximum length cycle and program halts.

```
# Load the necessary libraries.
with(networks): with(linalg): with(combinat,choose):
# Generate transition matrix.
tm1:=proc(x,y) local j, A, B;
    A:=diag(seq(1,i=1..x)):
    for j from 1 to x-1 do
        A:=addcol(A, j+1, j);
    od;
    for j from 1 to x-1 do
        A:=addcol(A, x-j, x-j+1, 1);
    od;
```

```
    B:= diag(1,seq(2,i=1..x-1));
    A:=matadd(A, B, 1, -1);
    B:=diag(seq(A,i=1..y)); B;
end:
tm2:=proc(x,y) local j, A, B;
    A:=diag(seq(1,i=1..x)):
    for j from 1 to x-y do
        A:=addcol(A, j+y, j);
    od;
    for j from 1 to x-y do
        A:=addcol(A, x-j-y+1, x-j+1);
    od;
    B:= diag(seq(1,i=1..y),seq(2,i=1..x-y));
    A:=matadd(A, B, 1, -1); A;
end:
# x-by-y matrix with all rule-zero cells
tm:=proc(x,y) local A, B, M;
    M:=matadd(tm2(x*y, x), tm1(x, y)); M;
end:
# Generate the transition matrix for the n-by-m LHCA and calculate its characteristic polynomial. If the CP is primitive, then breaks.
jc:=proc(n,m,j)
global p, F, Cone, q;
    local k, T;
    k:=n*m; p:=0; T:=tm(n,m);
    # Assign cell-rule to the transition matrix.
    Cone:=convert(choose(k,j),list);
    for q from 1 to nops(Cone) do
        for i from 1 to k do
```

```
            if member(i,Cone[q]) then x[i]:=1;
            else x[i]:=0; fi:
        od;
        B:=diag(seq(x[i],i=1..k));
    # Get the transition matrix with cell-rule and its characteristic polynomial.
        TMatrix:=matadd(T, B);
        E:=matadd(TMatrix, x*diag(seq(1,i=1..k))):
        CP:=det(E) mod 2;
    # If characteristic polynomial is primitive, set flag and stop the searching.
        if Primitive(CP) mod 2 then p:=1; break; fi;
    od:
    if p=1 then do
        print('The first Maximal length', n, 'by', m, 'CA with', j, 'rule-one cells is', Cone[q]);
        print ('The Primitive CP is',CP); end;
    else print('There are no Maximal length', n, 'by', m, 'CA with', j, 'rule-one cell.');
    fi;
end:
# Main program starts. Find the first Maximal Length CA starting from 1 rule-one cell.
nbymTM:=proc(n,m) local i;
    for i from 1 to n*m do
        jc(n,m,i);
        if p=1 then
            break; fi;
    od;
end:
# Here is an example to call the program and pass the parameters. Find a Primitive CP for minimal cost and maximum length 3-by-4
CA.
nbymTM(3,4);
```

The output of the program as below:

There are no Maximal length, 3, by, 4, CA with, 1, rule-one cell.

There are no Maximal length, 3, by, 4, CA with, 2, rule-one cell.

The first Maximal length, 3, by, 4, CA with, 3, rule-one cells is, [1,4,5]

The Primitive CP is , $1 + x + x^4 + x^6 + x^7 + x^8 + x^9 + x^{11} + x^{12}$

Appendix C

Experimental Results

The tables below show the detailed results of the Experiment 1 discussed in chapter 6. The experiment is that given 30 different initial states, using 1D LHCA or 2D LHCA as test pattern generators separately to simulate delay faults with increasing state length. 'Mean' is the average fault coverage among 30 tests for the circuit. 'Max' is the maximal fault coverage within 30 tests and 'Min' is the minimal. 'Range' is the difference between 'Max' and 'Min'. The experiment is based on the PPSFP algorithm and comprehensive fault equivalence rules for fault collapsing.

Length	Mean		Maximum (A)		Minimum (B)		Range (A-B)	
	1D	2D	1D	2D	1D	2D	1D	2D
256	94.23	93.90	96.81	96.66	92.71	91.64	4.1	5.02
512	97.35	97.44	99.09	98.78	95.9	94.83	3.19	3.95
768	98.33	98.31	99.39	99.09	97.11	96.96	2.28	2.13
1024	98.65	98.68	99.54	99.24	97.42	97.72	2.12	1.52
1280	98.80	99.04	99.7	99.7	97.57	97.72	2.13	1.98
1536	98.94	99.17	99.7	99.7	97.87	98.33	1.83	1.37
1792	99.02	99.24	99.7	99.7	98.02	98.33	1.68	1.37
2048	99.08	99.33	99.85	99.7	98.02	98.33	1.83	1.37
2304	99.17	99.42	99.85	99.85	98.02	98.63	1.83	1.22
2560	99.20	99.46	99.85	100	98.18	98.63	1.67	1.37
2816	99.32	99.50	99.85	100	98.33	98.63	1.52	1.37
3072	99.35	99.53	99.85	100	98.63	98.63	1.22	1.37
3328	99.41	99.58	99.85	100	98.94	98.63	0.91	1.37
3584	99.42	99.66	99.85	100	99.09	99.39	0.76	0.61
3840	99.46	99.69	99.85	100	99.09	99.39	0.76	0.61
4096	99.48	99.71	99.85	100	99.09	99.39	0.76	0.61
4352	99.52	99.73	99.85	100	99.09	99.39	0.76	0.61
4608	99.56	99.74	99.85	100	99.09	99.39	0.76	0.61
4864	99.58	99.76	99.85	100	99.24	99.39	0.61	0.61
5120	99.60	99.77	99.85	100	99.24	99.39	0.61	0.61
5376	99.63	99.80	99.85	100	99.24	99.39	0.61	0.61
5632	99.63	99.82	99.85	100	99.24	99.54	0.61	0.46
5888	99.64	99.84	99.85	100	99.24	99.54	0.61	0.46
6144	99.65	99.86	99.85	100	99.24	99.7	0.61	0.3
6400	99.66	99.87	99.85	100	99.24	99.7	0.61	0.3
6656	99.69	99.87	99.85	100	99.39	99.7	0.46	0.3
6912	99.69	99.89	99.85	100	99.39	99.7	0.46	0.3
7168	99.70	99.89	99.85	100	99.54	99.7	0.31	0.3
7424	99.71	99.89	99.85	100	99.54	99.7	0.31	0.3
7680	99.71	99.90	99.85	100	99.54	99.7	0.31	0.3
7936	99.72	99.90	99.85	100	99.54	99.7	0.31	0.3
8192	99.72	99.91	99.85	100	99.54	99.7	0.31	0.3
8448	99.72	99.92	99.85	100	99.54	99.7	0.31	0.3

Table C.1. *Fault Coverage vs. Sequence Length (c432nr-1)*

Length	Mean		Maximum (A)		Minimum (B)		Range (A-B)	
	1D	2D	1D	2D	1D	2D	1D	2D
8704	99.72	99.93	99.85	100	99.54	99.7	0.31	0.3
8960	99.73	99.93	99.85	100	99.54	99.7	0.31	0.3
9216	99.73	99.93	99.85	100	99.54	99.7	0.31	0.3
9472	99.74	99.93	99.85	100	99.54	99.7	0.31	0.3
9728	99.75	99.94	99.85	100	99.54	99.7	0.31	0.3
9984	99.76	99.94	99.85	100	99.54	99.7	0.31	0.3
10240	99.77	99.94	99.85	100	99.54	99.7	0.31	0.3
10496	99.77	99.94	99.85	100	99.54	99.7	0.31	0.3
10752	99.77	99.94	99.85	100	99.54	99.7	0.31	0.3
11008	99.77	99.94	99.85	100	99.54	99.7	0.31	0.3
11264	99.77	99.95	99.85	100	99.54	99.7	0.31	0.3
11520	99.77	99.96	99.85	100	99.7	99.7	0.15	0.3
11776	99.78	99.96	99.85	100	99.7	99.7	0.15	0.3
12032	99.78	99.96	99.85	100	99.7	99.7	0.15	0.3
12288	99.78	99.96	99.85	100	99.7	99.7	0.15	0.3
12544	99.78	99.96	99.85	100	99.7	99.7	0.15	0.3
12800	99.78	99.96	99.85	100	99.7	99.7	0.15	0.3
13056	99.79	99.96	99.85	100	99.7	99.7	0.15	0.3
13312	99.79	99.97	99.85	100	99.7	99.85	0.15	0.15
13568	99.79	99.98	99.85	100	99.7	99.85	0.15	0.15
13824	99.80	99.98	99.85	100	99.7	99.85	0.15	0.15
14080	99.80	99.98	99.85	100	99.7	99.85	0.15	0.15
14336	99.80	99.99	99.85	100	99.7	99.85	0.15	0.15
14592	99.80	100.00	99.85	100	99.7	99.85	0.15	0.15
14848	99.80	100.00	99.85	100	99.7	99.85	0.15	0.15
15104	99.80	100.00	99.85	100	99.7	99.85	0.15	0.15
15360	99.80	100.00	99.85	100	99.7	99.85	0.15	0.15
15616	99.81	100.00	99.85	100	99.7	100	0.15	0
15872	99.81	100.00	99.85	100	99.7	100	0.15	0
16128	99.81	100.00	99.85	100	99.7	100	0.15	0
16384	99.81	100.00	99.85	100	99.7	100	0.15	0

Table C.2. *Fault Coverage vs. Sequence Length (c432nr-2)*

Length	Mean		Maximum (A)		Minimum (B)		Range (A-B)	
	1D	2D	1D	2D	1D	2D	1D	2D
256	77.44	81.07	79.37	85.08	74.34	78.48	5.03	6.6
512	84.46	87.87	86.13	89.88	82.52	85.94	3.61	3.94
768	88.06	91.23	89.49	92.35	85.76	89.67	3.73	2.68
1024	90.39	93.37	91.75	94.38	88.67	91.38	3.08	3
1280	92.00	94.83	93.03	95.64	90.72	93.33	2.31	2.31
1536	93.04	95.79	93.87	96.48	91.72	94.48	2.15	2
1792	93.82	96.58	94.73	97.41	92.91	95.45	1.82	1.96
2048	94.49	97.11	95.13	97.93	93.66	96.2	1.47	1.73
2304	94.93	97.57	95.62	98.21	94.15	96.62	1.47	1.59
2560	95.33	97.94	95.87	98.48	94.9	97.39	0.97	1.09
2816	95.65	98.21	96.15	98.74	95.27	97.74	0.88	1
3072	95.89	98.46	96.39	98.88	95.34	98	1.05	0.88
3328	96.08	98.61	96.55	98.97	95.55	98.09	1	0.88
3584	96.28	98.76	96.83	99.04	95.8	98.3	1.03	0.74
3840	96.43	98.89	96.85	99.16	96.06	98.62	0.79	0.54
4096	96.54	99.00	96.92	99.3	96.15	98.72	0.77	0.58
4352	96.64	99.08	96.92	99.37	96.34	98.81	0.58	0.56
4608	96.74	99.15	97.04	99.37	96.39	98.88	0.65	0.49
4864	96.81	99.20	97.11	99.46	96.46	99.02	0.65	0.44
5120	96.88	99.23	97.16	99.49	96.53	99.04	0.63	0.45
5376	96.93	99.28	97.18	99.53	96.6	99.04	0.58	0.49
5632	96.98	99.32	97.25	99.58	96.74	99.07	0.51	0.51
5888	97.01	99.35	97.25	99.6	96.78	99.09	0.47	0.51
6144	97.05	99.38	97.27	99.63	96.85	99.11	0.42	0.52
6400	97.09	99.42	97.3	99.63	96.88	99.18	0.42	0.45
6656	97.12	99.44	97.3	99.65	96.92	99.3	0.38	0.35
6912	97.14	99.46	97.3	99.65	96.92	99.3	0.38	0.35
7168	97.17	99.47	97.37	99.67	96.99	99.3	0.38	0.37
7424	97.18	99.49	97.37	99.7	97.02	99.3	0.35	0.4
7680	97.20	99.51	97.39	99.7	97.02	99.3	0.37	0.4
7936	97.22	99.52	97.39	99.7	97.02	99.37	0.37	0.33
8192	97.23	99.53	97.39	99.7	97.04	99.39	0.35	0.31
8448	97.25	99.54	97.41	99.7	97.06	99.42	0.35	0.28

Table C.3. *Fault Coverage vs. Sequence Length (c3540nr-1)*

Length	Mean		Maximum (A)		Minimum (B)		Range (A-B)	
	1D	2D	1D	2D	1D	2D	1D	2D
8704	97.26	99.55	97.46	99.7	97.11	99.42	0.35	0.28
8960	97.28	99.56	97.46	99.7	97.11	99.42	0.35	0.28
9216	97.29	99.57	97.46	99.72	97.11	99.46	0.35	0.26
9472	97.30	99.58	97.46	99.72	97.11	99.46	0.35	0.26
9728	97.31	99.59	97.46	99.74	97.13	99.46	0.33	0.28
9984	97.32	99.59	97.48	99.74	97.13	99.46	0.35	0.28
10240	97.33	99.60	97.48	99.74	97.16	99.46	0.32	0.28
10496	97.34	99.61	97.48	99.74	97.18	99.51	0.3	0.23
10752	97.35	99.61	97.48	99.77	97.18	99.51	0.3	0.26
11008	97.35	99.62	97.48	99.77	97.2	99.53	0.28	0.24
11264	97.36	99.62	97.51	99.77	97.23	99.53	0.28	0.24
11520	97.37	99.62	97.51	99.77	97.23	99.53	0.28	0.24
11776	97.38	99.62	97.51	99.77	97.25	99.53	0.26	0.24
12032	97.39	99.62	97.51	99.77	97.25	99.53	0.26	0.24
12288	97.39	99.63	97.51	99.77	97.3	99.53	0.21	0.24
12544	97.40	99.63	97.51	99.77	97.3	99.53	0.21	0.24
12800	97.41	99.63	97.51	99.77	97.32	99.53	0.19	0.24
13056	97.41	99.64	97.51	99.79	97.34	99.53	0.17	0.26
13312	97.42	99.64	97.51	99.79	97.34	99.53	0.17	0.26
13568	97.43	99.64	97.51	99.79	97.34	99.53	0.17	0.26
13824	97.43	99.64	97.51	99.79	97.34	99.53	0.17	0.26
14080	97.44	99.65	97.53	99.79	97.34	99.53	0.19	0.26
14336	97.44	99.65	97.53	99.79	97.34	99.53	0.19	0.26
14592	97.44	99.66	97.53	99.79	97.34	99.56	0.19	0.23
14848	97.45	99.66	97.53	99.79	97.37	99.56	0.16	0.23
15104	97.45	99.66	97.53	99.79	97.37	99.56	0.16	0.23
15360	97.45	99.66	97.53	99.79	97.37	99.56	0.16	0.23
15616	97.45	99.66	97.53	99.79	97.37	99.56	0.16	0.23
15872	97.45	99.66	97.53	99.79	97.37	99.56	0.16	0.23
16128	97.45	99.67	97.53	99.79	97.37	99.56	0.16	0.23
16384	97.46	99.67	97.53	99.79	97.37	99.56	0.16	0.23

Table C.4. *Fault Coverage vs. Sequence Length (c3540nr-2)*

	Mean		Maximum (A)		Minimum (B)		Range (A-B)	
Length	1D	2D	1D	2D	1D	2D	1D	2D
256	88.95	89.14	91.19	92.64	87.36	86.51	3.83	6.13
512	92.20	92.72	93.64	94.87	89.96	90.73	3.68	4.14
768	93.67	94.42	94.94	96.09	92.57	92.34	2.37	3.75
1024	94.64	95.20	96.32	97.09	93.56	93.33	2.76	3.76
1280	95.28	95.74	96.7	97.24	93.72	93.87	2.98	3.37
1536	95.75	96.11	97.09	97.24	94.02	94.33	3.07	2.91
1792	96.05	96.46	97.62	98.24	94.41	94.41	3.21	3.83
2048	96.40	96.80	97.93	98.62	94.56	94.87	3.37	3.75
2304	96.58	97.01	98.01	98.77	94.94	95.17	3.07	3.6
2560	96.70	97.23	98.01	98.77	94.94	95.17	3.07	3.6
2816	96.99	97.43	98.01	99	95.25	95.56	2.76	3.44
3072	97.17	97.61	98.16	99.08	95.63	95.56	2.53	3.52
3328	97.39	97.71	98.47	99.08	95.79	95.56	2.68	3.52
3584	97.54	97.92	98.54	99.08	95.79	96.25	2.75	2.83
3840	97.60	98.02	98.54	99.08	95.79	96.32	2.75	2.76
4096	97.67	98.17	98.54	99.16	95.86	96.32	2.68	2.84
4352	97.74	98.36	98.62	99.31	95.94	96.63	2.68	2.68
4608	97.85	98.44	98.77	99.31	96.02	96.63	2.75	2.68
4864	98.02	98.51	98.93	99.39	97.09	96.63	1.84	2.76
5120	98.13	98.62	99.08	99.46	97.09	96.63	1.99	2.83
5376	98.15	98.65	99.08	99.46	97.09	96.78	1.99	2.68
5632	98.21	98.71	99.08	99.54	97.09	96.78	1.99	2.76
5888	98.33	98.77	99.16	99.69	97.32	96.86	1.84	2.83
6144	98.40	98.88	99.16	99.69	97.32	97.09	1.84	2.6
6400	98.50	98.95	99.23	99.69	97.62	97.16	1.61	2.53
6656	98.53	98.99	99.23	99.77	97.62	97.16	1.61	2.61
6912	98.57	99.04	99.23	99.77	97.7	97.16	1.53	2.61
7168	98.60	99.06	99.23	99.77	97.85	97.16	1.38	2.61
7424	98.66	99.15	99.31	99.77	97.85	97.24	1.46	2.53
7680	98.70	99.20	99.31	99.77	98.01	97.39	1.3	2.38
7936	98.75	99.22	99.31	99.77	98.01	97.39	1.3	2.38
8192	98.79	99.29	99.31	99.77	98.01	97.39	1.3	2.38
8448	98.82	99.32	99.39	99.77	98.01	97.39	1.38	2.38

Table C.5. *Fault Coverage vs. Sequence Length (c880-1)*

Length	Mean		Maximum (A)		Minimum (B)		Range (A-B)	
	1D	2D	1D	2D	1D	2D	1D	2D
8704	98.83	99.34	99.39	99.85	98.16	97.39	1.23	2.46
8960	98.85	99.39	99.39	99.85	98.16	98.31	1.23	1.54
9216	98.88	99.41	99.39	99.85	98.24	98.31	1.15	1.54
9472	98.90	99.44	99.39	99.92	98.31	98.31	1.08	1.61
9728	98.93	99.46	99.39	99.92	98.31	98.39	1.08	1.53
9984	98.95	99.48	99.39	99.92	98.31	98.39	1.08	1.53
10240	98.99	99.48	99.39	99.92	98.47	98.39	0.92	1.53
10496	99.03	99.49	99.39	99.92	98.54	98.39	0.85	1.53
10752	99.05	99.52	99.46	100	98.54	98.39	0.92	1.61
11008	99.05	99.54	99.46	100	98.54	98.39	0.92	1.61
11264	99.06	99.56	99.46	100	98.62	98.77	0.84	1.23
11520	99.09	99.57	99.46	100	98.62	98.77	0.84	1.23
11776	99.10	99.58	99.46	100	98.62	98.77	0.84	1.23
12032	99.12	99.60	99.46	100	98.62	98.77	0.84	1.23
12288	99.13	99.61	99.46	100	98.62	98.77	0.84	1.23
12544	99.13	99.61	99.46	100	98.62	98.77	0.84	1.23
12800	99.15	99.62	99.46	100	98.77	98.77	0.69	1.23
13056	99.16	99.63	99.46	100	98.77	98.77	0.69	1.23
13312	99.16	99.64	99.46	100	98.77	98.77	0.69	1.23
13568	99.18	99.65	99.46	100	98.77	98.77	0.69	1.23
13824	99.19	99.66	99.46	100	98.77	98.77	0.69	1.23
14080	99.20	99.68	99.46	100	98.77	98.77	0.69	1.23
14336	99.21	99.69	99.46	100	98.77	98.77	0.69	1.23
14592	99.22	99.69	99.46	100	98.77	98.77	0.69	1.23
14848	99.22	99.71	99.46	100	98.77	99	0.69	1
15104	99.23	99.73	99.46	100	98.77	99	0.69	1
15360	99.24	99.75	99.46	100	98.77	99.23	0.69	0.77
15616	99.24	99.75	99.46	100	98.77	99.23	0.69	0.77
15872	99.24	99.75	99.46	100	98.77	99.23	0.69	0.77
16128	99.25	99.76	99.46	100	98.77	99.23	0.69	0.77
16384	99.25	99.77	99.46	100	98.77	99.23	0.69	0.77

Table C.6. *Fault Coverage vs. Sequence Length (c880-2)*

Length	Mean		Maximum (A)		Minimum (B)		Range (A-B)	
	1D	2D	1D	2D	1D	2D	1D	2D
256	99.91	99.90	99.97	99.97	99.8	99.82	0.17	0.15
512	99.98	99.97	99.99	99.98	99.96	99.95	0.03	0.03
768	99.99	99.98	99.99	99.98	99.98	99.97	0.01	0.01
1024	99.99	99.98	99.99	99.98	99.99	99.97	0	0.01
1280	99.99	99.98	99.99	99.98	99.99	99.97	0	0.01
1536	99.99	99.98	99.99	99.98	99.99	99.97	0	0.01
1792	99.99	99.98	99.99	99.98	99.99	99.97	0	0.01
2048	99.99	99.98	99.99	99.98	99.99	99.97	0	0.01
2304	99.99	99.98	99.99	99.98	99.99	99.97	0	0.01
2560	99.99	99.98	99.99	99.98	99.99	99.98	0	0
2816	99.99	99.98	99.99	99.98	99.99	99.98	0	0
3072	99.99	99.98	99.99	99.98	99.99	99.98	0	0
3328	99.99	99.98	99.99	99.98	99.99	99.98	0	0
3584	99.99	99.98	99.99	99.98	99.99	99.98	0	0
3840	99.99	99.98	99.99	99.98	99.99	99.98	0	0
4096	99.99	99.98	99.99	99.98	99.99	99.98	0	0
4352	99.99	99.98	99.99	99.98	99.99	99.98	0	0
4608	99.99	99.98	99.99	99.98	99.99	99.98	0	0
4864	99.99	99.98	99.99	99.98	99.99	99.98	0	0
5120	99.99	99.98	99.99	99.98	99.99	99.98	0	0
5376	99.99	99.98	99.99	99.98	99.99	99.98	0	0
5632	99.99	99.98	99.99	99.98	99.99	99.98	0	0
5888	99.99	99.98	99.99	99.98	99.99	99.98	0	0
6144	99.99	99.98	99.99	99.98	99.99	99.98	0	0
6400	99.99	99.98	99.99	99.98	99.99	99.98	0	0
6656	99.99	99.98	99.99	99.98	99.99	99.98	0	0
6912	99.99	99.98	99.99	99.98	99.99	99.98	0	0
7168	99.99	99.98	99.99	99.98	99.99	99.98	0	0
7424	99.99	99.98	99.99	99.98	99.99	99.98	0	0
7680	99.99	99.98	99.99	99.98	99.99	99.98	0	0
7936	99.99	99.98	99.99	99.98	99.99	99.98	0	0
8192	99.99	99.98	99.99	99.98	99.99	99.98	0	0
8448	99.99	99.98	99.99	99.98	99.99	99.98	0	0

Table C.7. *Fault Coverage vs. Sequence Length (c6288nr-1)*

Length	Mean		Maximum (A)		Minimum (B)		Range (A-B)	
	1D	2D	1D	2D	1D	2D	1D	2D
8704	99.99	99.98	99.99	99.98	99.99	99.98	0	0
8960	99.99	99.98	99.99	99.98	99.99	99.98	0	0
9216	99.99	99.98	99.99	99.98	99.99	99.98	0	0
9472	99.99	99.98	99.99	99.98	99.99	99.98	0	0
9728	99.99	99.98	99.99	99.98	99.99	99.98	0	0
9984	99.99	99.98	99.99	99.98	99.99	99.98	0	0
10240	99.99	99.98	99.99	99.98	99.99	99.98	0	0
10496	99.99	99.98	99.99	99.98	99.99	99.98	0	0
10752	99.99	99.98	99.99	99.98	99.99	99.98	0	0
11008	99.99	99.98	99.99	99.98	99.99	99.98	0	0
11264	99.99	99.98	99.99	99.98	99.99	99.98	0	0
11520	99.99	99.98	99.99	99.98	99.99	99.98	0	0
11776	99.99	99.98	99.99	99.98	99.99	99.98	0	0
12032	99.99	99.98	99.99	99.98	99.99	99.98	0	0
12288	99.99	99.98	99.99	99.98	99.99	99.98	0	0
12544	99.99	99.98	99.99	99.98	99.99	99.98	0	0
12800	99.99	99.98	99.99	99.98	99.99	99.98	0	0
13056	99.99	99.98	99.99	99.98	99.99	99.98	0	0
13312	99.99	99.98	99.99	99.98	99.99	99.98	0	0
13568	99.99	99.98	99.99	99.98	99.99	99.98	0	0
13824	99.99	99.98	99.99	99.98	99.99	99.98	0	0
14080	99.99	99.98	99.99	99.98	99.99	99.98	0	0
14336	99.99	99.98	99.99	99.98	99.99	99.98	0	0
14592	99.99	99.98	99.99	99.98	99.99	99.98	0	0
14848	99.99	99.98	99.99	99.98	99.99	99.98	0	0
15104	99.99	99.98	99.99	99.98	99.99	99.98	0	0
15360	99.99	99.98	99.99	99.98	99.99	99.98	0	0
15616	99.99	99.98	99.99	99.98	99.99	99.98	0	0
15872	99.99	99.98	99.99	99.98	99.99	99.98	0	0
16128	99.99	99.98	99.99	99.98	99.99	99.98	0	0
16384	99.99	99.98	99.99	99.98	99.99	99.98	0	0

Table C.8. *Fault Coverage vs. Sequence Length (c6288nr-2)*

Appendix D

User Manual for "sim3"

NAME

sim3 - a combinational logic fault simulator

SYNOPSIS

sim3 [options] **circuitfile**

DESCRIPTION

Read in a description of a combinational circuit from circu- itfile with the ISCAS'85 netlist format produced by OASIS (an Open Architecture Silicon Implementation Software). Simulate fault-free, single stuck-at faults, and/or single delay faults and/or single transistor stuck-open faults for a given pseudorandom test pattern generator implemented by either a LFSR, LHCA or Binary-Counter. The test patterns can also be from a file or the standard input. sim3 evalu- ates exact fault coverage and the number of undetected faults. The fault coverage is given by $((N-U)/N) * 100$ where N is the number of total faults of the type being considered and U is the number of undetected faults.

The implementation of sim3 is based on the PPSFP (parallel Pattern Single Fault Propagation) technique, and the comprehensive fault equivalence rules for fault collapsing are applied. Before simulating stuck-open faults, sim3 replaces XOR, AND and OR gates with NAND, NOR and NOT gates in the circuit under test. The LHCA considered is one- dimensional linear cellular automata. Two kinds of LHCA are used, namely LHCA with minimum cost and CA corresponding to the LFSR with minimum cost. The types of linear feedback shift register (LFSR) used are the LFSR with minimum cost, LFSR with half taps

evenly distributed and LFSR similar to the LHCA with minimum cost.

sim3 can also produce the circuit netlist with the local format and a truth table with the espresso format.

OPTIONS

-V Produce circuit file with local format.

-F Produce a truth table with the espresso format.

-R Replace XOR, AND and OR with NOT, NAND and NOR gates in the circuit under test.

-S Simulate Stuck-at faults.

-D Simulate Delay faults.

-O Simulate Stuck-Open faults (imply option -R).

-Gn Choose test pattern generator, where n is a number from the range 0 to 5, corresponding to the following types of the test pattern generator:

0 - 1D LHCA with minimum cost(default).

1 - LHCA corresponding to the LFSR with minimum cost.

2 - LFSR with minimum cost.

3 - LFSR with half taps evenly distributed.

4 - LFSR corresponding to the LHCA with minimum cost.

5 - Binary Counter.

By default of -Gn, the test patterns are produced by the LHCA with minimum cost. If choosing more than one generator, the last one is used.

-I Use Type I (XORs are between the cells) if choosing the LFSR. By default, use Type II (XORs are on the feedback path).

-X Use the crossover technique for the generator.

-Q Include all zeros' pattern for the LHCA and LFSR generator.

-i *seed* Set initial random seed for the generator. the seed is a positive number for producing a random value.

-l *length* Set test sequence length. By default, the length of sequence is 2**m, where

m is the number of inputs in the circuit. If m $\geq$ 30, this option must be specified.

-v *vectorfile* Test patterns are from the vectorfile. If using option -v +, the test pattern are read from the standard input stdin. If the -v option specified, the option -Gn will be ignored.

-T Produce fault coverage for each 256 test patterns.

-o *outfile* Send the output produced to the file outfile. By default, the output is written to the standard output stdout.

EXAMPLES

sim3 -SD -l 4096 -o c432.out c432.isc

Read circuit description from c432.isc and evaluate fault coverages with single stuck-at faults and delay faults using 4096 test patterns produced by the LHCA with minimum cost. The output is sent to the file c432.out.

VITA

Surname: Zhu ***Given Names:*** Jie Xia (Cathy)

Place of Birth: China ***Date of Birth:*** Oct. 28, 1972

Educational Institutions Attended

University of Victoria 2001 to 2003

Degrees Awarded

B.EE. Jinan University 1995

UNIVERSITY OF VICTORIA PARTIAL COPYRIGHT LICENSE

Title of Thesis:

Two-Dimensional Linear Hybrid Cellular Automata
as Test Pattern Generators in VLSI Testing

Author: [redacted]
JIE XIA ZHU (CATHY)
April 24, 2003