BIST Signature Analysis: Analytical Techniques for Computing the Probability of Aliasing

André Ivanov

B. Eng (McGill) 1983, M. Eng. (McGill) 1985

Department of Electrical Engineering

McGill University

A thesis submitted to the Faculty of Graduate Studies and Research

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

September 1988

Abstract

Testing VLSI circuits is a complex task that requires enormous amounts of resources To decrease testing costs, testing issues are considered earlier in the design process. This is known as "design for testability" (DFT). Built-in Self Test (BIST) is one proposed DFT approach. BIST generally consists of incorporating additional circuitry on the chip to generate test patterns and compact the response of the circuit under test (CUT) into a reference signature. Compaction implies an information loss, introducing the possibility that a faulty circuit declares itself as good. Such errors are known as *aliasing* errors. Several BIST schemes have been proposed, and each has a particular performance in regard to aliasing. However, the schemes are often evaluated and compared with ill defined measures for which the underlying assumptions are either not stated or not understood clearly. Here, a novel classification for the measures of aliasing is proposed. By providing clear definitions of different possible measures, the proposed classification augments the understanding of the aliasing problem.

This dissertation focuses on the popular BIST scheme that consists of applying pseudorandom test patterns to a CUT and compacting the latter's response by a signature analysis register which is normally a *linear feedback shift register* (LFSR). Assessing the quality of such a scheme in regard to fault coverage is crucial. Fault coverage can be established by full fault simulation. However, high costs may preclude this approach. Other techniques, probabilistic in nature, have been proposed, but a lack of computationally feasible techniques for analyzing the aliasing problem under a reasonable model has left them elusive. Here, new and computationally feasible techniques are developed More specifically, closed-form expressions for the probability of aliasing are derived for a certain type of LFSRs. Upper bounds are derived for LFSRs characterized by primitive polynomials. An iterative technique is developed for computing the exact probability of aliasing for LFSRs characterized by any feedback polynomial, and for any test sequence length. These new techniques enable better assessments of the quality of BIST schemes that use signature analysis for response compaction. In turn, they are useful for making important design decisions, e.g., determining the number

of test patterns that should be applied to a CUT to achieve a certain test confidence; alternatively, deciding how long the signature analyzer should be, and what type of feedback it should possess to achieve a certain desired test confidence.

The techniques developed for computing the probability of aliasing in BIST are also useful in the context of coding theory. The iterative technique developed for computing the probability of aliasing may be used as an efficient technique for computing the probability of an undetected error for shortened versions of cyclic codes

Résumé

Le test des circuits intégrés à très grande échelle, i.e., circuits VLSI, est un problème complexe qui exige une quantité considérable de ressources informatiques et humaines. Afin de réduire les coûts de test, il est nécessaire de considérer le test aussitôt que possible dans la phase de concéption d'un circuit, i.e., pratiquer du "design for testability" (DFT). Le test intégré, ou BIST (Built-In Self-Test), est un exemple de DFT. Le BIST consiste à intégré avec le circuit original, un circuit generateur de tests, et un circuit qui sert à comprimer la réponse du circuit sous test. Cette compression implique une perte d'information qui introduit la possibilité qu'un circuit défectueux se déclare bon, soit une erreur de "aliasing" ou de "masking". Plusieurs types de BIST ont été proposés. Tous ont une différente performance par rapport au "aliasing". Différentes mesures de "aliasing" sont aussi utilisées. Malheureusement, ces mesures sont souvent ambigues, et prêtent à de mauvaises interprétations. Cette dissertation présente une nouvelle classification de ces mesures. En soi, cette classification augmente la compréhension du problème de "aliasing".

Le type de BIST qui consiste à appliquer des tests pseudo-aléatoires et de comprimer la réponse du circuit avec un "signature anslysis register" ou "linear feedback shift register" (LFSR) constitue une approche très populaire. Cette dernière est l'approche traitée ici. Il est essentiel d'établir la qualité d'un tel type de test, e.g., par rapport à la proportion de pannes détectées. Une technique bien connue qui sert à cette fin est la simulation de panne. Cependant, pour les circuits VLSI, la simulation de panne peut s'avérer très couteuse, et même impossible. D'autres techniques d'analyse probabilistique ont déjà été proposées. Malheureusement, jusqu'a ce jour, dû au manque de techniques d'analyse du problème de "aliasing" en BIST, la qualité de ces techniques probabilistiques est demeurée faible. Ici, de nouvelles techniques d'analyse du problème de "aliasing" sont dévelopées. Plus spécifiquement, des expréssions fermées pour la probabilité de "aliasing" sont obtenues pour un certain type de LFSR, et des bornes supérieures de la probabilité de "aliaising" sont obtenues pour les "signature analysis registers" (LFSR) charactérisés par des polynômes primitifs. Une technique

d'analyse itérative a aussi été dévelopée. Cette dernière permet d'obtenir la probabilite de "aliasing" en fonction de la longueur de la séquence de test, pour tout type de LFSR Ces techniques permettent une meilleur estimation de la qualité du BIST utilisant "signature analysis" comme technique de compréssion Ainsi, ces techniques sont utiles pour déterminer les paramètres tels que la longueur de test, et le type de compresseur requis, afin d'obtenir la qualité de test desirée.

Les techniques d'analyse dévelopées içi sont aussi utiles dans le contexte de la théorie du codage. La technique itérative dévelopée pour le calcul de la probabilite de "aliasing" constitue une technique efficace pour le calcul de la probabilité d'une erreur non detectée lorsque l'information est encodée par des codes cycliques.

Acknowledgements

First, I want to acknowledge my advisor, Vinod K. Agarwal, for his invaluable contributions to my past and future. I am indebted to him for his insight, knowledge, guidance, and enthusiasm. The knowledge and the thinking/reasoning processes that I acquired with him through our numerous (interrupted!) discussions, will undoubtedly positively mark my future work, and hence life. For all of these, I am most grateful.

I also gratefully acknowledge the interest and concern for my progress and person, that Professor Nicolas Rumin illustrated for me during all my years spent as a member of the McGill VLSI Design Lab. To Profs N. Freidman and J. Rajski, I say thank you for for their guidance, time, and help, that they offered me at different times in the course of my work.

On another level, yet not less important, I am enormously grateful to my family and friends. My parents' love and support provided the foundation for the realization of this thesis. I am extremely grateful for everything they did and gave up for me. I am indebted to my wife for her contributions to my life and person during the past few years. Her love made my success possible in the past, and I know her love will continue to make it possible in the future. I must also acknowledge her technical contribution to my work, namely in helping me find and work out the details of the Example in Chapter 2!

I am grateful to my friends from the time of infancy of the McGill VLSI Design Laboratory: Nicos Cotsapas, Michel Dagenais, Karim Khordoc and Marcos Peckel. Although all of them, at one point, left me behind in the lab, they all continued to show interest and concern for my progress. I am especially thankful to Marcos for our innumerable appointment phone calls (threatening of defeat!) followed by athletic activities of different sorts, including real ones! I am thankful to Nicos for consistently overcoming the physical distance separating us, and thus enabling us to continue sharing excellent times.

I am grateful to Trang Nguyen for the frequent and immediate help that she offered me since the first day that she joined the McGill VLSI Design Laboratory. Finally, I am indebted to Prof. D. Avis who (at the last minute!) discovered a flaw in the proof of a lemma contained in the thesis. In turn, I am grateful to J.P. Caisso and R. Aitken for providing me with useful insight for correcting this flaw.

This work was supported by scholarships and grants from the Natural Sciences and Engineering Research Council of Canada and from the Ministere de l'Enseignement Superieur du Quebec through their "actions structurantes" program.

Contents

Abstract ..... *ii*

Résumé ..... *iv*

Acknowledgements ..... *vi*

Table of Contents ..... *viii*

List of Figures ..... *xi*

List of Tables ..... *xiii*

Claim of Originality ..... *xiv*

Chapter 1 Introduction 1

1.1 Perspective and Dissertation Outline 1

1.2 Testing of ICs 3

1.2.1 Classical Approaches 3

1.2.2 Design for Testability 5

Chapter 2 Aliasing and its Measures 12

2.1 Introduction 12

2.1.1 Example 16

2.2 Measures based on DV_f 18

2.2.1 Combinatorial 18

2.2.2 Spectral 19

2.2.3 Probabilistic 19

2.3 Measures based on DV_e 20

2.3.1 Combinatorial 20

2.3.2 Spectral 21

2.3.3 Probabilistic 21

2.4 Summary 25

Chapter 3 Single- and Multi-Stage Probabilistic Analysis of LFSRs 26

3.1 Introduction 26

3.2 Computation of Single-Stage State Probability Sequences 28

3.2.1 Notation and Preliminaries 28

3.2.2 Calculation of $\mu(n)$ 31

3.3 Examples and Heuristic Applications of Single-Stage State Probability Sequences 35

3.4 Multi-Stage State Probability Sequences 38

3.4.1 Formulation as a Signal Probability Problem 38

3.4.2 Polynomials $1 + x^m$ 43

3.4.3 Primitive Polynomials 45

3.5 Probability of Aliasing 47

3.6 Summary 50

Chapter 4 Iterative Technique for Calculating State Probabilities of LFSRs 54

4.1 Introduction 54

4.2 Preliminaries 56

4.3 Computing State Probability Sequences of LFSRs 58

4.3.1 Motivation and Basis for the Technique 58

4.3.2 Calculating Probability Sequences of Disjunctions of States of LFSR Stages 61

4.3.3 Detailed Space and Time Complexity Analysis of the Iterative Algorithm 67

4.4 Experimental Results 69

4.5 Summary 70

Chapter 5 Applications and Extensions 77

5.1 Detection Confidence and Test Length Calculations 77

5.1.1 Detection confidence before compaction 78
5.1.2 Detection confidence and test length after compaction 79
5.1.3 Summary 82
5.2 Extensions to Multi-Output Circuits 83
5.3 Coding Theory Applications 87
5.3.1 Introduction 87
5.3.2 Probability of an Undetected Error 88
5.3.3 P_u for Various Code Generator Polynomials 91
5.3.4 Summary 94

Chapter 6 Conclusion 96

References *102*

Appendix A. Proof of Theorem 3.3 110

Appendix B. Proof of Theorem 4.5 117

Appendix C. Proof of Theorem 4.9 128

List of Figures

2.1 General BIST scheme. 13

2.3 Signature analysis circuit with polynomial $1 + x + x^2$. 17

2.2 Example circuit to illustrate deception volumes. 17

3.1 BIST scheme with a general signature analysis register. 28

3.2 Simple LFSR with characteristic polynomial $f(x) = 1 + x^2 + x^3$ . . 30

3.3 $\mu(n)$ for the polynomial $f(x) = 1 + x^{16}$. 36

3.4 $\mu(n)$ for the polynomial $f(x) = 1 + x + x^7 + x^{10} + x^{16}$. 37

3.5 $\mu(n)$ for the polynomial $f(x) = 1 + x^{10} + x^{30} + x^{31} + x^{32}$. 38

3.6 $\mu(n)$ for the polynomial $f(x) = 1 + x + x^2 + x^{22} + x^{32}$. 39

3.7 $\mu(n)$ for the polynomial $f(x) = 1 + x^2 + x^3 + x^4 + x^{50}$. 40

3.8 $\mu(n)$ for the polynomial $f(x) = 1 + x^7 + x^{10}$. 41

3.9 Example of the formulation of a state probability problem as a signal probability problem. 42

3.10 Gate-level representation for the polynomial $1 + x^2 + x^3$, $n = 7$. . . . 47

3.11 Probability of aliasing for polynomial $1 + x^3$. 49

3.12 Probability of aliasing for polynomial $1 + x^8$. 50

3.13 Bounds on the probability of aliasing for primitive polynomials of degree 8. "——" corresponds to eq. (3.20) and " · " corresponds to eq. (3.21). 51

3.14 Bounds on the probability of aliasing for primitive polynomials of degree 16. "——" corresponds to eq. (3.20) and "···" corresponds to eq. (3.21). 52

3.15 Bounds on the probability of aliasing for primitive polynomials of degree 15, and the exact probability of aliasing for the polynomial $1 + x + x^{15}$. 53

4.1 Probability of aliasing, $f(x) = 1 + x^8 + x^{10}$. 70

4.2 Probability of aliasing, $f(x) = 1 + x^8 + x^{10}$ 71

4.3 Probability of aliasing, $f(x) = 1 + x^7 + x^{10}$ 72

4.4 Probability of aliasing, $f(x) = 1 + x^7 + x^{10}$ 73

4.5 Probability of aliasing, $f(x) = 1 + x + x^{15}$, $\epsilon = .01$. 74

4.6 Probability of aliasing, $f(x) = 1 + x + x^{15}$, $\epsilon = .90$. 75

4.7 Probability of aliasing, $f(x) = 1 + x^7 + x^{10}$, $n \leq 250 : \epsilon = .01$;
$n > 250 : \epsilon = .90$. 76

5.1 Multi-output circuit with a scan chain and an LFSR for SA. 85

5.2 Multi-output circuit with a MISR for SA. 86

5.3 (a) A MISR, and (b) its equivalent single-input LFSR [Sridhar 82]. 87

5.4 Probability of an undetected error; CCITT code,
$g(x) = 1 + x^5 + x^{12} + x^{16}$ 92

5.5 Probability of an undetected error; CCITT code,
$g(x) = 1 + x^5 + x^{12} + x^{16}$ 93

A.1 XOR gate with q uncut lines and r cut lines. 111

List of Tables

2.1 Fault Effects and Signatures for Example Circuit. 17

5.1 Escape Probability and Test Length: Polynomials $1 + x^m$ 81

5.2 Escape Probability and Test Length: HP Polynomial $1 + x^7 + x^9 + x^{12} + x^{16}$ 83

Claim of Originality

The author claims originality for the following contributions of the dissertation.

• In Chapter 2, the proposed classification of aliasing measures is novel. The principal criterion for the classification is the previously defined notion of *deception volume*.

• In Chapter 3, a novel technique is developed for analyzing the probabilistic behaviour of LFSRs under the assumption of a binomial distribution of error sequences. The technique is based on the definition of *single-stage state probability sequences*. The complete theory for obtaining such sequences for LFSRs characterized by any feedback polynomials is given.

• From single-stage state probability sequences, closed-form expressions for the probability of aliasing for LFSRs characterized by a certain class of feedback polynomials $(1 + x^m)$ are derived. Although such polynomials have been studied elsewhere, no closed-form expressions for the probability of aliasing have previously been published. Also, using this analysis, upper bounds on the probability of aliasing are derived for LFSRs characterized by primitive polynomials

• In Chapter 4, a completely different and also novel analysis technique is developed for calculating the probability of aliasing for LFSRs characterized by any feedback polynomial This technique is iterative. For a signature analysis register of length m, the technique involves $2^m - 1$ recursion equations. However, using the proposed notation, these $2^m - 1$ equations are expressed with only two general equations, which renders the implementation of the technique feasible. The technique is very general in that it enables the calculation of the aliasing probability under essentially any distribution of errors. For example, any type of burst errors can be studied readily.

• In Chapter 5, the techniques developed in the earlier chapters to compute the probability of aliasing are used for calculating the predicted test quality assuming that random

patterns are applied to a circuit and the latter's response is compacted by a signature analysis register (LFSR). Such calculations were not previously possible due to the lack of feasible techniques for calculating the probability of aliasing under the binomial error model.

• The iterative technique developed in Chapter 4 for calculating the aliasing probability is also useful in the context of coding theory. It may serve for efficiently calculating the probability of an undetected error for shortened cyclic codes generated by any generator polynomial, under either the standard binary symmetric channel (BSC) model, or more general channel models. The technique may thus be useful for comparing the performance, in regard to the probability of undetected errors, of various shortened cyclic codes. Furthermore, some of the features of the technique make it suitable for helping in adopting certain codes instead of others.

Chapter 1 Introduction

1.1 Perspective and Dissertation Outline

Since the inception of integrated circuits (ICs), over two decades ago, the IC fabrication technologies have not ceased improving. Among other impacts, these technology improvements have been synonymous with continuously increasing scales of integration. Hence, in only two decades, the levels of integration have been qualified from small (SSI) to medium (MSI), large (LSI), very large (VLSI), and ultra large (ULSI). The ever increasing scales of integration have resulted in the possibility of fabricating chips, and from these systems, with an ever increasing number of devices, at lower and lower costs. Of course, today's possibility of fabricating extremely complex circuits at low cost is very attractive for several reasons. Unfortunately, as in any other engineering endeavor, trade-offs arise in the overall enterprise. Let alone the ever increasing difficulty for designers to correctly design the intended circuits, the problem of testing the fabricated circuits against fabrication defects grows much more than linearly with the size (complexity) of the circuits fabricated [Goel 80]. It is now common to hear from manufacturers that the testing cost of a product is more than one third of its total cost. Hence, although many problems associated with testing appeared early in the history of ICs, the recent soaring of the complexity of systems fabricated is forcing experts to address the testing problem with a new perspective, namely that of recognizing that the testing problem plays a key role in the economical success of a product.

This recent recognition of testing's crucial role has spawned a number of research efforts whose fruits have been various, probably the most important being the proposition of new design techniques collectively known as *design for testability* (DFT) [Williams 82]. An increasing number of manufacturers are now adopting such techniques and many research efforts are currently underway to improve DFT techniques and their integration with the classical IC design and fabrication steps.

The broad motivation for this dissertation is the problem of testing digital circuits, with focus on a relatively recently proposed testing technique known as *built in self test* (BIST) [Design & Test 85]. BIST is becoming increasingly popular since it is believed to have the potential of greatly simplifying the test procedure and hence decreasing the overall cost of testing a system. However, as in the case of more traditional testing techniques, it is generally necessary to establish the quality of a BIST scheme in regard to the fault coverage that it achieves [McCluskey 86]. It is generally not straightforward to make such an assessment. Making such an assessment for a particular BIST scheme constitutes the more specific motivation for the work presented in this dissertation. The BIST scheme in question is one that is applicable to general unstructured logic where the application of pseudorandom test sets is combined with the signature analysis (LFSR- or polynomial division- based) compaction technique [Bardell 87]

Compaction implies a loss of information which results in the possibility of erroneous verdicts concerning the proper functioning of circuits. Such errors are known as *aliasing* errors. In general, the nature and measure of such aliasing errors needs to be well understood to establish the fault coverage achieved by any BIST scheme. However, in spite of IC manufacturers' adoption of certain BIST schemes, the general understanding of the associated aliasing problem often remains very skimpy. Due to the lack of sufficient understanding, as well as feasible analytical tools to augment this understanding, unjustified claims are frequently made. The major contribution of this dissertation is the provision of analytical techniques that are useful for making better measurements (predictions) of aliasing, and consequently increase the understanding of the aliasing problem associated with the popular signature analysis compaction technique.

The dissertation is briefly organized as follows. The remainder of this chap

ter presents further general preliminaries for the contents of the subsequent chapters. The classical approaches to testing are briefly discussed. This is followed by a brief introduction to DFT and BIST. In Chapter 2, the problems of aliasing and its measures are formally discussed. A novel classification of aliasing measures is presented. Some relationships between aliasing measures and fault coverage measures are also discussed. In general, both deterministic and probabilistic measures of aliasing may be defined. For large circuits however, it may be infeasible to compute deterministic measures. Hence, probabilistic measures are frequently used. Probabilistic measures are those upon which this dissertation focuses. Unfortunately, probabilistic measures of aliasing are often misunderstood, misused or misinterpreted. The reason for this is that the measures are often derived from an unrealistic assumption on the distribution of errors at a faulty CUT's output. This assumption, and its unrealism, is often forgotten or overlooked. In particular, signature analysis is a widely-used compaction technique which is one for which misleading claims, e.g., in regard to fault coverage, are frequently made based on the assumption of a uniform distribution of error sequences. Chapters 3 and 4 propose novel techniques for calculating probabilities of aliasing for signature analysis, under more realistic error distributions. Chapter 5 presents a framework for using these techniques in establishing the (probabilistic) fault coverage. A generalization of the use of the techniques for multi-output circuits is also presented in Chapter 5. The techniques proposed for calculating the probability of aliasing for signature analysis have important ramifications in coding theory, namely in calculating the probability of undetected errors for shortened cyclic codes. These ramifications are briefly discussed in Chapter 5. Finally, Chapter 6 concludes the dissertation.

1.2 Testing of ICs

1.2.1 Classical Approaches

Testing of digital integrated circuits generally consists of three major steps, the first being the generation of a set of input patterns, the second being the application of these patterns to the circuit under consideration, and the third being the analysis

of the collected output data with an expected response to finally declare the circuit good or bad. The above test procedure may be applied at various fabrication levels of a product, e.g., chip, board, system levels [Mann 80] [Myers 83]. Generally, it is believed that the cost of testing increases by an order of magnitude from one level to the next, and that the overall cost is minimal when the defects are detected as early as possible [Williams 82]. These observations have resulted in a major portion of the recent research in testing to focus on the chip or board levels. The following discussion assumes chip level testing. However, most of the ideas and notions are equally applicable to the board level.

Several different types of physical failures can affect the function and performance of ICs. Thus, several tests to detect defects must be performed at different stages of fabrication. The technology, as well as various factors such as the scale of integration, operating voltage and temperature, etc., together influence the types of failures that may arise in ICs. Since it may be impossible to test circuits for all possible types of physical failures, several technical and economic factors govern the testing of circuits (e.g., time and budget available for testing, critical failures required to be tested, etc.) Because of the large number and the complex nature of all the possible physical failures, testing usually consists of merely detecting the presence or absence of defects, buts does not generally require knowledge of the exact failure itself, i.e., diagnosis.

One practical approach used to avoid dealing with the physical failure per se consists of developing a fault model which describes the physical failures at a higher level of abstraction (e.g., logic gate, register). A higher level model reduces the number of entities that must be considered when deriving tests for the circuit. Once a fault model that accurately describes or captures the possible physical failures has been developed, it suffices to generate tests for detecting all the faults in the fault model instead of deriving tests for all the possible physical failures. Despite recent questioning of its appropriateness, especially for CMOS circuits [Wadsack 78] [Galiay 80], the most widely-used fault model is the single *stuck-at* fault model, in which the lines of a circuit, if faulty, are assumed to be permanently stuck at the logic value 0 or 1.

Given a circuit under test (CUT), a fault in the CUT is said to be detected

when a particular input pattern applied to the CUT produces an incorrect logic response on one or more of its outputs. Such incorrect responses are called *errors*, and such input pattern constitutes a test for that fault. A set of input patterns that detect a set of faults is called a *test set*, and a set of test patterns that detects all the testable modeled faults is said to constitute a complete test set. If a test set is not complete, then the most commonly used measure of the quality of the test set is the *fault coverage* measure which is simply the ratio (usually expressed as a percentage) of the detected faults to the total number of detectable faults in the CUT. Hence, in the case of a complete test set, the fault coverage of detectable faults is 100%.

Several test pattern generation algorithms have been developed in the past to generate test sets, e.g., [Roth 67] [Goel 81] [Fujiwara 83] [Rajski 87] [Schulz 88]. Although they are generally highly desirable, complete test sets are also generally prohibitively expensive to generate, even with the best available test pattern generation algorithms. One process often used in combination with test pattern generation algorithms is called *fault simulation* [Chang 70] [Armstrong 72] [Levendel 81] [Bose 82] [Maamari 88] which consists of determining all the faults that a particular set of patterns may detect. Fault simulation is usually combined with test pattern generation since its computational complexity is less (both on the average and in the worst case) than that of test pattern generation. Nevertheless, in spite of constant advances in the algorithms for test pattern generation and fault simulation, in the case of large circuits these processes may simply be too expensive to be feasible. When the situation is such, alternative methods to classical test pattern generation and fault simulation must be used. Several of these have been proposed recently.

1.2.2 Design for Testability

1.2.2.1 Scan Path Testing

The limitations of the classical approaches to testing have fostered the proposal by IC manufacturing and design communities of new approaches for testing complex VLSI circuits. Such approaches are commonly known as *design for testability*

(DFT) techniques [Williams 82]. Probably the most well-known example of such techniques is an idea that arose from the particular difficulty in testing sequential circuits. This idea is the notion of *scan* design [Eichelberger 78] [Stewart 77] [Funatsu 75] which essentially reduces the problem of testing a sequential circuit to that of testing a combinational circuit. This is accomplished by introducing two modes of operation: normal and test. The reduction in the complexity of testing a sequential circuit is thus obtained at the expense of extra hardware required to reconfigure the circuit at test time. Another possible drawback of introducing a test mode is that certain types faults may be undetected since the circuit is being tested outside its normal mode of operation. Alternatively, while a circuit could perform its normal function correctly, making a declaration about the correctness of the circuit by observing it in test mode introduces the possibility that the same circuit be declared faulty because of a fault in the extra circuitry exercised only in test mode.

The concept underlying scan design remains the same in spite of the differences that appear in the details of the various schemes proposed [Eichelberger 78] [Stewart 77] [Funatsu 75]. In scan design, the circuit's latches are extended to have two modes of operation. In *normal* mode, the latches perform as ordinary latches. In *test* mode, the normal feedback paths of the sequential circuit are broken and the latches form a shift register. With the shift register configuration, a test vector can be *scanned* in the register from one primary input of the circuit. Then, this test vector may be applied to the circuit by returning the circuit to its normal mode for one clock cycle. Once the response of the test vector is latched, the circuit may be returned to test mode to scan out this response through a primary output of the circuit. This response may then be compared to that expected. Exploiting the scanning in and out capabilities of such a scheme enables the combinational logic of the circuit to be fully tested, i.e., tested for the faults that the applied test set was constructed to detect. Since they form the path through which test patterns are applied, the latches are also tested in this procedure.

In exchange for the reduced complexity in testing, scan design implies hardware overhead. In the case of IBM's Level Sensitive Scan Design (LSSD) [Eichelberger

78], this overhead varies from 15% to 30% [Williams 82]. Also, scan design generally requires a more complex clocking scheme than would normally be required. For example, in IBM's LSSD, the control of the circuit requires two clocks (master and slave) in both the test mode and in the normal mode of operation. The sharing of one is possible, hence a total of three clocks is required. Nevertheless, in LSSD, the normal operating speed of the circuit is only slightly degraded by the additional functionality of the latches

Thus, the most obvious advantage of scan design is the reduced complexity of generating test sets for a sequential circuit. This is accomplished by treating the latter as a combinational circuit when it is being tested This reduction in complexity is generally obtained against little or no performance degradation for the normal function of the circuit. The principal disadvantages are the hardware overhead, arising from extensions in the latches and the extra control circuitry, extra primary inputs and outputs (up to four in some cases), and the time required for scanning tests in and out of a circuit.

1.2.2.2 Built-In Self-Test

Another design for testability approach scheme which can well be coupled with scan design is the idea of *built-in self test* (BIST) [Design & Test 85] [Bardell 87]. BIST is usually understood as a scheme where all circuitry associated with testing a chip is part of the chip itself. Thus, an input pin is required to select between normal and self-test modes. In self-test mode, test patterns are generated on the chip, and the responses are observed and analyzed on the chip. Finally a simple form of go/no go or pass/fail signal is delivered at one of the chip's output pins For sequential circuits, scan design is usually assumed Hence, in self-test mode, the CUT is combinational, and the latches form part of the test circuitry.

BIST requires an on-chip source of test patterns. The most popular scheme for generating patterns is to configure certain latches into a *linear feedback shift register (LFSR)* [Bardell 87]. For combinational blocks with less than say 20 inputs, the

LFSR can be used to apply all possible input combinations to the CUT [McCluskey 84]. For blocks with a larger number of inputs, the LFSR is usually used to apply a pseudorandom source of input patterns [Bardell 87].

BIST usually implies a relatively large number of test patterns, and hence a considerable response data, especially if the circuit has several outputs. For example, applying $O(2^{20})$ combinations to a 10-output circuit yields $O(10^7)$ output bits. Thus, in BIST, some form of data compaction is used to reduce this output data into a *signature* of only a few bits long, e.g., 16 or 32. At the end of the test sequence the signature is compared to a stored reference signature, which was determined at design time by simulation or analytic means. If the signature of the CUT is different from the reference signature, the CUT declares itself faulty through an output pin. Obviously, several variations of the aforementioned BIST ideas exist. For example, either or both the test pattern source and the response compactor may reside outside the actual CUT. The analyses presented in subsequent chapters are in fact applicable to any testing scheme where the output response is compacted by a particular type of circuit, which may be residing either on or off chip.

There are several obvious advantages of BIST. Since pseudorandom patterns are applied, no potentially costly test generation is required. Also, a BIST can be applied at close to the normal operating speed of the CUT. Thus more types of faults, e.g., delay faults, may be detected which would not necessarily be detected using other techniques. The extra test circuitry required is small. Only two extra pins are required, one for the mode of operation and one for the pass/fail signal. Finally, BIST introduces the possibility of easy and inexpensive field testing. This may be particularly attractive at the system level, where the BIST capabilities of individual chips may also greatly facilitate diagnosis.

Of course BIST also has its disadvantages and limitations. The most obvious disadvantage is the area overhead, which is highly design dependent. Also, as for scan path testing, a fault in the extra test circuitry can result in a circuit being declared faulty while the latter could perform its function normally. Various possible limitations

also arise depending on the circuit and the particular BIST scheme adopted. For example, if exhaustive testing is performed, circuits with a large number of inputs may require an exceedingly large test set and associated test time [Bozorgui-Nesbat 80]. On the other hand, if pseudorandom testing is performed, certain faults, known as *random pattern resistant* faults, may exhibit extreme reluctance to detection by the pseudorandom patterns [Eichelberger 83]. Such undetected faults may result in an unsatisfiable fault coverage.

One of the most serious difficulties with BIST is the information loss incurred by the compaction stage. This loss introduces the possibility that a fault be *masked*, or *aliased* [McCluskey 85] That is, the compaction introduces the possibility that a faulty circuit produce the same signature as that of the fault-free circuit. This problem of aliasing makes the assessment of the fault coverage in BIST a very difficult problem [Bhavsar 84] [Cox 88].

1.2.2.3 Compaction Schemes for BIST

Several compaction schemes have been proposed for BIST, e.g., [Carter 82] [Frohwerk 77] [Hassan 84] [Hayes 76] [Hurst 85] [Miller 84] [Muzio 83] [Savir 80] [Saxena 86] [Susskind 81] [Zorian 86]. The reader is referred to these references for details about these schemes. In the following, only a few brief comments will be made concerning the better-known schemes, from which several variations have been derived.

In *syndrome* testing [Savir 80], the signature for a single-output circuit is the number of ones that appear in the response data when all the possible input patterns are applied to the circuit. To count the number of ones in the response when not all input patterns are applied is of course also possible, just as it is possible to constitute the signature from other counts, such as the number of transitions [Hayes 76]. *Spectral coefficient* testing [Hurst 85] [Miller 84] [Muzio 83] [Susskind 81] is a generalization of the syndrome concept. In general, the signature in spectral coefficient testing consists of several counts Each of the counts correspond to the number of ones in the exclusive-OR of the circuit output with a particular subset of inputs. Thus, in general, spectral

testing extends the fault coverage of syndrome testing. However, this is at the expense of a greater area overhead required for the exclusive-ORs, the additional counters, and reference counts. This significant additional hardware in return for not necessarily much better fault coverage is the strongest deterrent against the practical use of spectral testing.

The fault coverage achieved by either syndrome or spectral testing can be assessed using different methods. Fault simulation is one, while for the common single stuck-at fault model, the coverage can in some cases be determined using analytical methods. Such methods however, are generally quite impractical to handle.

For a single-output circuit, an LFSR can be used to compact the output response of the CUT. This is the approach proposed by Hewlett Packard [Frohwerk 77] for testing microprocessor boards, known as *signature analysis*. The mathematical interpretation of compaction by an LFSR is rather straightforward. The process can be interpreted as polynomial division over a Galois field [Golomb 82] [Peterson 72]. The LFSR compactor can also easily be extended to the multiple-output circuit, using a *multiple-input shift register* (MISR).

Compaction by an LFSR is very attractive in that it is very simple and does not require much area overhead compared to other BIST schemes. In some cases, the compaction circuit can be combined with the test pattern source [Koenemann 79]. Unfortunately, there is no easy way of precisely characterizing the effect of a particular fault on the output bits of a CUT when pseudorandom test patterns are applied to the latter. Therefore, establishing the fault coverage when an LFSR is used to compact the response of a circuit is not straightforward. This will be explained further in later sections of this dissertation since this problem constitutes the underlying motivation for the work presented herein.

A final comment is in order. This dissertation is neither directly concerned with the justification of previously proposed BIST schemes nor the proposal of a new BIST scheme. The interested reader is referred to the literature for a thorough discussion on BIST, e.g., see [Zorian 87] for an excellent bibliography on BIST, or the special issue

[Design & Test 85]. This dissertation focuses on the problem of aliasing for BIST schemes in which the signature analysis technique is used to compact the CUT's output.

Chapter 2 Aliasing and its Measures

2.1 Introduction

Recently, various built-in self test (BIST) schemes have received considerable attention. There are several reasons for the interest in BIST but one of the principal reasons is due to BIST's effectiveness for overcoming the problems now encountered by conventional methods when dealing with large circuits. In general, as illustrated in Fig. 2.1, BIST circuits consist of the original circuit to which three major blocks are added: one for the test pattern generation, a second for the response compaction, and a third for comparing the compacted output (signature) to a reference. Concerning the comparison circuitry, several schemes exist in which the latter is a trivial circuit, or is external to the CUT. Here the focus is on the compaction aspect of BIST. Response compaction implies an information loss that translates itself in the possibility of wrong diagnosis whereby a faulty circuit declares itself as good. Such incorrect diagnosis is caused by what is known as an *aliasing* or *masking* error [McCluskey 85]. Various compaction techniques (functions) have been proposed, e.g., [Carter 82] [Frohwerk 77] [Hassan 84] [Hayes 76] [Miller 84] [Muzio 83] [Savir 80] [Saxena 86] [Susskind 81] [Zorian 86]. Each of the different compaction schemes leads to particular information losses. Hence, whenever adopted, each compaction function must be analyzed to establish the impact that its associated loss of information will have on the quality of the BIST scheme.

Ideally, in the same way that fault coverage measures are generally used to

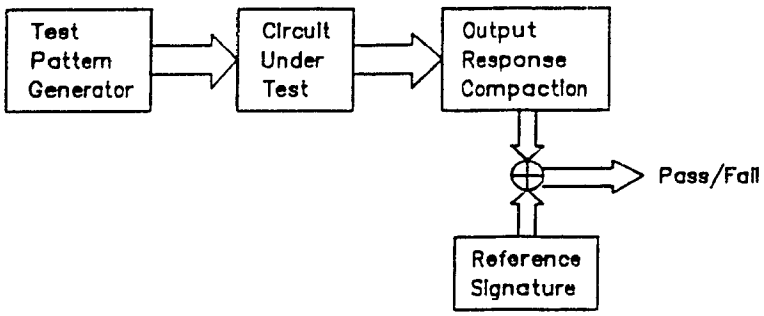

Figure 2.1 General BIST scheme

assess the quality of a test set in conventional (external testing) situations, the use of the same measures would also be desirable in the context of BIST. Unfortunately, the compaction stage in BIST complicates the matter. One reason for this complication is the following. Without compaction, the fault coverage is a monotonically increasing function of the number of test patterns. In the case where exact fault coverage cannot be established, monotonicity enables lower bounds to be claimed. However, in general, strict monotonicity disappears when compaction is performed.

For the sake of convenience, until a later section of this dissertation, the circuit under test (CUT) is assumed to be single-output. Assuming that an input test sequence of length n is applied to the CUT, the output response sequence is also of length n. The sequence produced by a faulty CUT which contains erroneous bits is referred to as an *error sequence*. In the context of testing without output response compaction, a faulty CUT is correctly diagnosed as being faulty if the output sequence contains at least one erroneous bit. However, in the case of testing with output response compaction, a faulty CUT is correctly diagnosed as being faulty only if its output sequence contains a least one erroneous bit *and* if the error sequence it produced is not mapped onto the signature of the fault-free circuit.

The role of the compaction stage is to reduce a CUT's output sequence to a signature of only a few bits long, say m, where $m \ll n$. Let E be the set of all the $2^n - 1$ possible binary error sequences of length n. Hence, $|E| = 2^n - 1$. Let S be the set of all the possible signatures that a compaction function can yield. The cardinality of

the set S depends on the specific compaction function. Without regard to a CUT and its potential faults and hence potential error sequences, a compaction function maps each of the sequences contained in the set E onto one particular signature of the set S. Since only the final signature matters after a particular sequence is compacted, the set of error sequences that map onto the same signature forms an equivalence class with respect to the specific compaction function. Let E_0 be the subset of E that contains all the elements of E that map onto the signature S_0 of the fault-free circuit, and define the *error domain deception volume* DV_e to be the cardinality of E_0, i.e., $DV_e = |E_0|$.

For most compaction functions, the mapping of the elements of E onto the set of signatures S is well understood in that it is theoretically well characterized. Hence, it is generally easy to characterize E_0 and thus find DV_e for most compaction functions. In the context of BIST, DV_e is a valuable measure since it gives the number of error sequences that would escape detection because they are mapped onto the signature S_0 of the fault-free circuit and consequently result in aliasing errors. Although not always explicitly, DV_e is traditionally used for establishing the quality of compaction functions. Examples are discussed later.

Assessing the quality of compaction functions is one facet of the problem. However, in reality, one is usually interested in determining the overall quality of a BIST scheme, i.e., fault coverage obtained from a specific set of test patterns applied to a specific CUT in a specific order. Thus, the problem of establishing the quality of compaction functions is complicated when a specific CUT and its faults are considered. Assuming that the test set is fixed (types and order of patterns) if there are N possible faults (single or multiple) in the circuit, then there are at most N different possible error sequences that can be produced by the specific CUT to which the specific test set is applied. Let F be the set of these possible error sequences. Typically, $|F| \ll |E|$. In turn, let F_0 be the subset of F which contains all the elements of F that map onto the signature S_0 of the fault-free circuit. Finally, let the *fault domain deception volume* DV_f be defined as the cardinality of F_0, i.e., $DV_f = |F_0|$.

Similarly to DV_e, DV_f is a measure of the loss of information and hence measure of aliasing caused by the compaction stage. However, unlike DV_e, DV_f is

specific to a CUT and its faults, as well as specific to the test set and the order in which it is applied to the CUT. Hence, because the relation between the potential faults of the CUT and the elements of F is an invertible function (one-to-one and onto), DV_f can directly be combined with fault coverage measures [Cox 88] to yield a fault coverage measure that takes into account the effects of compaction. However, it is not so when F is not known and only E is available. The relation between the elements of E and the potential faults of a CUT is not necessarily one-to-one and onto. This prevents DV_e from being directly used with the classical fault coverage measures.

Therefore, making claims on the fault coverage when compaction is performed requires a precise knowledge of the error sequences that a CUT may produce under the influence of its different faults, i.e., requires the knowledge of F. Unfortunately, finding all the elements of the set F may be computationally prohibitive for large circuits. Not knowing F, a minimal requirement is to have some characterizations, e.g., statistics, of the set F. Unfortunately, such characterizations have generally been missing until now. These two difficulties associated with F result in DV_f generally being very difficult to find. On the other hand, finding DV_e analytically remains generally easy.

Nevertheless, in the past, because fault coverage is generally the quality measure considered of prime interest, (as opposed to the mere assessment of the quality of a compaction function independently from a specific CUT and its test set), in the cases where compaction was performed but where F could not be obtained, various assumptions on F and hence DV_f have been made. Some of these assumptions have led to claims that become somewhat paradoxical upon brief reflection. An example of such claim is derived assuming that $F = E$, and that $DV_f = DV_e$. Combined with the unrealistic assumption that all the elements of F are equally likely to occur, i.e., uniform distribution of error sequences, this leads to the 2^{-m} probability of aliasing associated with signature analysis. Although the result is correct under the stated assumptions, these assumptions are often forgotten or overlooked. The result states that for signature analysis, the probability of aliasing is 2^{-m}, irrespectively of the CUT, the effects its faults may produce at its output, the number and order of test patterns applied to it,

etc. That the aliasing probability can be independent of all these factors and be made arbitrarily low simply by increasing the value of m is paradoxical. If this were true, signature analysis would constitute a sort of panacea, and hence no further research efforts should be aimed at studying this compaction function further, nor would efforts be expended at finding and studying other alternative compaction functions.

Thus, before the field is impregnated by too many misleading or misunderstood results, a critical regard on how the quality of compaction functions is measured and reported is believed to be in order. As mentioned above, the deception volume constitutes a fundamental measure of the information loss in the context of testing. Deception volume has thus been used in different ways, either implicitly or explicitly, to assess the quality of compaction functions. However, as in the case of fault coverage measures, the different ways of computing the deception volume may yield misleading results [Cox 88]. In the following, different methods for computing and interpreting measures based on deception volume are reviewed, and parallels between these methods and those discussed in [Cox 88] for reporting fault coverage are established. Although the discussion is applicable to different compaction functions, the focus is on signature analysis since it is the most popular scheme, often supported by unjustified claims. Moreover, signature analysis is the compaction technique on which the remaining chapters of this dissertation focus. The underlying assumptions of the measures are discussed, as well as their advantages and disadvantages. Two categories emerge: measures based on DV_f and measures based on DV_e. Prior to discussing these measures, an example that illustrates some of the concepts discussed above is given.

2.1.1 Example

Consider the circuit in Fig. 2.2. Assume that single stuck-at faults are considered, and that an exhaustive test sequence is applied to the CUT in the given order, i.e., 000 to 111. In Table 2.1, the response of the fault-free (FF) circuit, as well as that of the circuit under the effects of ten different possible stuck-at faults is reported. Therefore, $|F| = 10$ and $|E| = 2^8 - 1 = 255$. Thus, for this example, the

cardinalities of F and E differ significantly. In Table 2.1, A/0 denotes the fault A stuck-at 0, A 1 denotes the fault A stuck-at 1, etc. The signatures that result from three different compaction schemes is also given in Table 2.1. These compaction schemes are: signature analysis [Frohwerk 77], syndrome [Savir 80], and transitions count [Hayes 76]. In the case of signature analysis, the polynomial characterizing the register is $1+x+x^2$. Such a register is shown in Fig 2.3.

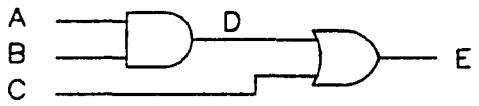

Figure 2.2 Example circuit to illustrate deception volumes

Fault Effects & Signatures											
Input Vector	Faults										
ABC	FF	A 0	A 1	B/0	B/1	C/0	C/1	D/0	D/1	E/0	E/1
000	0	0	0	0	0	0	1	0	1	1	0
001	1	1	1	1	1	0	1	1	0	1	0
010	0	0	1	0	0	0	1	0	1	1	0
011	1	1	0	1	1	0	1	1	0	1	0
100	0	0	0	0	1	0	1	0	1	1	0
101	1	1	1	1	0	0	1	1	0	1	0
110	1	0	1	0	1	1	0	0	1	1	0
111	0	1	0	1	0	1	0	1	0	1	0
Compression Scheme	Signature										
signature analysis	11	10	01	10	00	01	00	10	11	01	00
syndrome	4	4	4	4	4	2	6	4	4	8	0
transitions	6	7	4	7	6	1	1	7	7	0	0

Table 2.1 Fault Effects and Signatures for Example Circuit

In the case of signature analysis, only the fault D/1 yields the same signature as that of the fault-free circuit. Hence, for this specific circuit and this specific test sequence, the fault domain deception volume is one, i.e., $DV_f = 1$. However, the error domain deception volume is [Smith 81]: $DV_e = 2^{8-2} - 1 = 63$. For syndrome, from

Table 2.1, the fault domain deception volume is: $DV_f = 6$. On the other hand, the error domain deception volume is: $DV_e = \binom{8}{4} - 1 = 69$. Finally, for transitions count, from Table 2.1 the fault domain deception volume is: $DV_f = 1$. The error domain deception volume for transitions count is [Bardell 87]: $DV_e = 2\binom{8-1}{6} - 1 = 13$

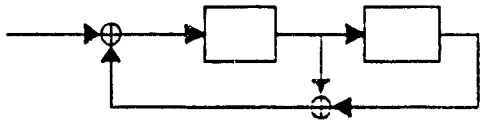

Figure 2.3 Signature analysis circuit with polynomial $1 + x + x^2$

Only single faults are considered here, however, multiple faults could easily have been considered.

2.2 Measures based on DV_f

Measures of DV_f require that F be known. Obtaining F requires that the CUT be simulated under the effect of each of its potential faults to obtain all the possible error sequences The principal disadvantage of this approach is the prohibitive cost of the fault simulation in the case of large circuits. For a circuit with N lines and hence $O(N)$ faults, the fault simulation of n patterns implies a computational effort of $O(nN^2)$. However, the clear advantage of this approach is that DV_f can be directly combined with fault coverage measures.

2.2.1 Combinatorial

Since F is known, one can define $DV_{f,comb}$ to be the analog of combinatorial fault coverage [Cox 88] where $DV_{f,comb}$ is simply the following ratio:

$$DV_{f,comb} = DV_f/|F|. \tag{2.1}$$

This measure is very straightforward to obtain given F and DV_f. $DV_{f,comb}$ is a true measure of the proportion of a CUT's possible error sequences that would alias.

For the Example in Section 2.1.1, the following values for $DV_{f,comb}$ result:

signature analysis: $DV_{f,comb} = 1\ 10$.

syndrome: $DV_{f,comb} = 6/10$.

transitions count: $DV_{f,comb} = 1\ 10$.

2.2.2 Spectral

An analog of spectral fault coverage [Cox 88] in the case of DV_f can also be defined. Let $\mathbf{DV_f}$ be a vector such that $\mathbf{DV_f} = (DV_{f,1}, DV_{f,2}, \ldots, DV_{f,q})$ where $DV_{f,i}$ corresponds to the number of error sequences of weight i that are mapped onto the signature of the fault-free circuit. In turn, the following ratios can be defined:

$$DV_{f,i}/|F_i|, \quad i = 1, 2, \ldots, q. \tag{2.2}$$

where $|F_i|$ is the number of elements of F that have weight i. It would be feasible to compute such ratios in the given context because the extra work that would be required to obtain the components $DV_{f,i}$ would be negligible compared to the efforts required for obtaining F and DV_f in the first place.

2.2.3 Probabilistic

An analog to the probabilistic measure of fault coverage [Cox 88] can also be defined for DV_f. Let the set F_v be composed of the elements $(f_1, f_2, \ldots, f_q)$. By definition, $q = DV_f$. Let each f_i have the probability p_i of occurring over the entire set F. Then the following probabilistic measure of DV_f can be defined:

$$DV_{f,prob} = \sum_{i=1}^{q} p_i. \tag{2.3}$$

As in the case of the probabilistic measure of fault coverage discussed in [Cox 88], the clear difficulty with this measure lies in finding the probabilities p_i.

2.3 Measures based on DV_e

In most cases, because of the prohibitive cost of fault simulation, F is not available. For such cases, the only reasonable alternative for determining the quality of a compaction function is to consider E and hence DV_e. The major disadvantage in not knowing F is that the knowledge of DV_e cannot be directly combined to classical fault coverage measures. On the other hand, several measures of DV_e can easily be obtained analytically. This is illustrated next.

2.3.1 Combinatorial

Similarly to $DV_{f,comb}$, a combinatorial measure $DV_{e,comb}$ can be defined to be the following ratio:

$$DV_{e,comb} = DV_e/|E|. \tag{2.1}$$

The advantage of this measure is that it is very easily obtained because $|E|$ is by definition trivial to obtain, and DV_e can generally be easily obtained analytically. For example, in the case of signature analysis, because of the well-established theory underlying the properties of linear feedback shift registers (LFSR), the precise number of error sequences that are mapped onto the fault-free signature, i.e., DV_e is easily determined to be $2^{n-m} - 1$ for a signature analyzer of size m [Smith 80] [Bardell 87]. In the case of count-based compaction functions (*ones count* or *transitions count*) it is possible to compute DV_e using simple combinatorics [Bardell 87].

For the Example in Section 2.2.1, the following values for $DV_{e,comb}$ result

signature analysis: $DV_{e,comb} = 63/255 \approx 1/4$,

syndrome: $DV_{e,comb} = 69/255 \approx 1/4.5$,

transitions count: $DV_{e,comb} = 13/255 \approx 1/20$.

For this particular example, for the three different types of compaction, the difference between $DV_{f,comb}$ and $DV_{e,comb}$ is quite significant. Thus, assuming these ratios to be similar is not reasonable in this case.

2.3.2 Spectral

Similarly to the definition of a spectral measure of DV_f, a spectral measure of DV_e can also be defined. For this, let $\mathbf{DV_e} = (\mathrm{DV_{e,1}}, \mathrm{DV_{e,2}}, \ldots, \mathrm{DV_{e,q}})$ where $DV_{e,i}$ is defined as the number of error sequences of weight i that map onto the signature of the fault-free circuit. Then, from the vector $\mathbf{DV_e}$, the following ratios can be defined:

$$DV_{e,i}/|E_i|, \quad i = 1, 2, \ldots, q. \tag{2.5}$$

where $|E_i|$ is the total number of error sequences of weight i.

In the situation where F is known, the additional effort of finding the components of $\mathbf{DV_f}$ would be justifiable. However, in this case where F is not known, due to the possibly enormous size of E, finding all the components of $\mathbf{DV_e}$ may be infeasible. On the other hand, for signature analysis, the well-established mathematical theory (coding theory) that underlies the compaction function enables $DV_{e,i}$ to be found analytically for several specific values of i. For example, the theory yields that for all feedback configurations, $DV_{e,1} = 0$ [Smith 80]. Certain feedback configurations guarantee that $DV_{e,i} = 0$ for all odd i [Bardell 87]. In fact, $DV_{e,i}$ for all i can be determined analytically in the cases where the weight distribution of the underlying code (determined by the feedback configuration) is known [Lin 83]. For the cases where the weight distribution is not known, other techniques can be used to efficiently compute $DV_{e,i}$, e.g., [Fujiwara 85].

2.3.3 Probabilistic

Probabilistic measures derived from DV_e are probably the most widely used, and are generally better-known as *probabilities of aliasing*. Unfortunately, as mentioned in the introductory discussion of this section, these are also the measures that can easily be misinterpreted, especially when they are based on unreasonable assumptions.

Probabilistic measures of DV_e can be defined similarly to the probabilistic measures of DV_f. Let $E_v = (e_1, e_2, \ldots, e_q)$, where $q = DV_e$. Let each e_i have a

probability of occurrence p_i. Then the following probabilistic measure of DV_e can be defined:

$$DV_{e,prob} = \sum_{i=1}^{q} p_i. \tag{2.6}$$

Although it may ideally be a good measure, the major problems with this measure is that the enumeration of all the error sequences (e_i's) that are mapped onto the fault-free signature may typically require an enormous amount of computational effort, and it may be impossible to determine a reasonable p_i to assign to each of these sequences.

Thus, other types of probabilistic measures that do not require an enumeration of the set E_e, nor the assignment of a unique probability to each of the sequences, are usually used. These assume a certain probability distribution for the set of error sequences E.

Uniform Distribution of Error Sequences

The probability model most commonly used is to assume that all $2^n - 1$ error sequences of length n have an equal likelihood of occurrence, i.e., a uniform distribution. Unfortunately, there is no indication that this assumption is realistic for circuits in general. A simple example to illustrate that this assumption is not reasonable is to consider a CUT with a stuck-at fault on its one output. Assuming random test patterns are applied to the CUT, if the latter was fault-free, "1"'s would be produced with a certain probability p. If the output was stuck-at-0, then the probability that the output would be erroneous would be $1 - p$. In general, $p \neq 1/2$, therefore $1 - p \neq 1/2$. Therefore, to claim that such output fault is likely to generate any one of the $2^n - 1$ possible error sequences with the same probability is clearly not a good assumption. A more formal argument for the inadequacy of the equally likelihood assumption can be found in [Carter 82].

A well-known example of a result that follows from such an assumption arises in the case of signature analysis. Equal likelihood implies $p_i = 1/(2^n - 1)$ for all i. For signature analysis, $q = DV_e = 2^{n-m} - 1$ [Smith 80]. This yields:

$$DV_{e,prob} = \sum_{i=1}^{q} p_i = \frac{2^{n-m} - 1}{2^n - 1} \approx 2^{-m}. \tag{2.7}$$

The above result states that for signature analysis, under the assumption of equally likely error sequences, the probability of aliasing is 2^{-m}, independently of the effect of the fault on the output response of the CUT, the length and order of the test sequence, and the feedback configuration of the signature register. If faith is placed in the assumption, this result can be used to argue that the probability that a fault will be masked can be made arbitrarily small simply by increasing m, and hence the expected fault coverage of the BIST scheme made arbitrarily high. Hence, because assuming a uniform distribution of error sequences is not a reasonable assumption, the use of the results based on this assumption must be interpreted extremely carefully.

Binomial Distribution of Error Sequences

Another possible assumption on the distribution of error sequences is to assume that the effect of a fault in a CUT is the production of an error at the CUT's output with a certain probability ϵ. This model of fault-effects, (for single faults) has been successfully used in several schemes proposed for establishing the quality of a random test set at the cost of a computational effort only linear in the size of the circuit, e.g., [Brglez 84] [Seth 85]. Note that in practice, truly random test patterns are not applied to a CUT. Pseudorandom test patterns generated from maximal-length LFSRs are usually applied. In that case, because of the "sampling without replacement" type of process, i.e., non-repeatability of patterns, ϵ does not actually remain constant. However, for test sequence lengths that are relatively small compared to the total number of possible patterns that may be generated by the LFSR, i.e., the LFSR's cycle length, treating ϵ for a particular fault as a constant does not incur a substantial error in the analysis [Wagner 87] [McCluskey 87]. Once the effect of a fault d_i is assumed to be characterized by a certain probability ϵ_i that any given bit of the output response is in error, this implies that the distribution of error sequences for that particular fault is binomial with probability of success (or failure) ϵ_i. That is, any error sequence with j erroneous bits have a probability of occurrence $\epsilon_i^j(1-\epsilon_i)^{n-j}$. Since there are $\binom{n}{j}$ such sequences, the probability that an error sequence has j erroneous bits is $\binom{n}{j}\epsilon_i^j(1-\epsilon_i)^{n-j}$. Assuming that the distribution of error sequences resulting from a particular fault is binomial, the problem is then to determine what is the probability of an aliasing error

for a particular fault d_i characterized by a certain probability of error ϵ_i.

In [David 78], some results on aliasing assuming a binomial distribution of error sequences were derived for a very specific type of signature analysis registers. More recently, more general results on aliasing for signature analysis under this model of fault effects were obtained by Williams et al. [Williams 86], who derived the important result that as the test sequence length tends to infinity, the probability of aliasing approaches 2^{-m} for all feedback configurations and all probabilities of error ϵ. They also obtained the qualitative results that signature registers characterized by primitive polynomials approach this 2^{-m} asymptote more rapidly (shorter test lengths) than do registers characterized by non-primitive polynomials. However, for obtaining non-asymptotic results, the method of analysis used is not computationally feasible for registers of length of interest in practice, say $m > 4$. Williams et al. themselves resorted to simulation methods to analyze longer registers.

In Chapter 3, an alternative analysis technique is proposed which enables some non-asymptotic results to be obtained under the binomial distribution of error sequences. In Chapter 4, a completely general, hence more powerful, technique for calculating the exact probability of aliasing, given any feedback configuration and any probability of error ϵ, is given. The calculation technique described in Chapter 4 also enables an extended model of fault-effect to be analyzed easily. That is, it is possible to analyze aliasing for signature analysis assuming distributions other than binomial distributions of error sequences. More specifically, the technique permits the probability of aliasing to be calculated in the situation where the probability of an error may differ for every bit. That is, the technique may yield the probability of aliasing for a sequence characterized by the probabilities of error $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$. Each probability being different may be an unlikely extreme situation, but the situation where the probability of error changes over the test sequence does arise in the case of biased random testing [Lisanke 86] [Wunderlich 87] [Waicukauski 88], a scheme shown to be very efficient for reducing the required test length for achieving extremely good fault coverage results.

Here, the calculation of $DV_{\epsilon,prob}$ under the assumption of a binomial distribution of error sequences is discussed only for signature analysis. In [Aitken 88],

this measure is discussed for ones count compaction. It is important to analyze compaction functions under this error model because the binomial distribution constitutes a more reasonable characterization of E under which it is more acceptable to assume that $E = F$ since the binomial distribution weighs the elements of E more reasonably compared to the uniform distribution which assigns equal weight to all the elements of E. The binomial distribution assumptions enables measures based on DV_e to be used in the same way that DV_f would be used with classical fault coverage measures to establish the quality of a test set.

2.4 Summary

The principal measures of information loss entailed by compaction functions for BIST have been categorized into two broad categories: those based on DV_f and those based on DV_e. When F is available, the measures based on DV_f are the most precise, and hence most desirable However, in most cases, it is infeasible to find F. In that case, measures based on DV_e must be used. The problem with measures based on DV_e is that they may not generally be directly combined with classical fault coverage measures to claim a fault coverage after compaction. However, assuming a binomial distribution of error sequences constitutes a reasonable assumption on E that render the resulting measures based on DV_e more justifiably usable directly with the classical fault coverage measures. Unfortunately, few BIST compaction techniques have been analyzed under the binomial distribution of error sequences. Most have been only analyzed for the uniform distribution of error sequences, the major reason for this being the lack of good analysis tools to handle the binomial distribution. The next two chapters present analysis tools that enable the signature analysis compaction scheme to be analyzed under the binomial distribution of error sequences.

Chapter 3

Single- and Multi-Stage Probabilistic Analysis of LFSRs

3.1 Introduction

In the preceding chapter, it was briefly mentioned that both [David 78] and [Williams 86] reported the performance (in regard to aliasing) of signature analysis registers or *linear feedback shift registers* (LFSRs) under the binomial distribution of error sequences. David only studied a particular class of signature analysis registers, namely the configurations where only the last stage feeds back to the input stage. Williams et al. generalized the study to all types of feedback configurations. Unfortunately, despite the originality of their analysis, the latter is severely hindered by the exponential size of the problem. In their analysis, Williams et al. model the LFSR by a Markov chain [Trivedi 82]. By demonstrating some of the stochastic properties of the system, Williams et al. derived the important asymptotic result that the probability of aliasing tends to 2^{-m} for a register of size m when the test sequence length tends to infinity, irrespectively of the feedback configuration and the probability of error. For the non-asymptotic or dynamic behaviour of the system, Williams et al. proposed the use of the z-transform technique. The major problem with this proposal is that obtaining the inverse z-transform involves a computational time complexity of $O(2^{3m})$, where m is the number of stages of the LFSR. This complexity arises from the fact that the Markov chain contains 2^m states, and that the z-transform inversion requires the inversion of a matrix of size $2^m \times 2^m$. Assuming a standard matrix inversion algorithm implies

$O(2^{3m})$ complexity. Such complexity renders the calculation of the analytic solution for LFSRs of sizes of interest in practice generally infeasible. Due to the overwhelming complexity of the solution technique, the plots for the dynamic behaviour of different signature analysis registers that appear in [Williams 86] were not obtained using the z-transform technique. Instead, a form of simulation was used [Williams 86], for which no details were given. Due to the limitations of their techniques for obtaining the dynamic solution of interesting LFSRs, Williams et al. subsequently proposed upper bounds for the probability of aliasing [Williams 87], which they claimed to be valid for all ranges of test sequence lengths.

In this chapter, an alternative analysis technique for obtaining both the asymptotic and dynamic behaviour of a signature analysis register (LFSR), under the binomial error model, is described. Other than providing new insight into the dynamic behaviour of LFSRs, the principal advantage of this new analysis technique is that it is computationally inexpensive. However, in general, the problem at hand possesses an inherent exponential complexity (see Chapter 4) (though whether a lower bound on the complexity of the problem has formally been proven remains unknown to the author). Therefore, not surprisingly, the technique presented in this chapter does not yield an inexpensive solution in all cases, i.e., for all types of feedback configurations. Nevertheless, for the cases where the computational complexity of an exact solution becomes intolerable, the technique presented here still yields useful heuristic (approximate) information.

The essence of the technique presented in this chapter is to decompose an original problem of size 2^m into m smaller interrelated problems. The "work-horse" of the technique presented in detail in subsequent sections of this chapter is the *single-stage state probability sequence* of an LFSR. Such sequences were studied in [David 78]. In the latter however, the sequences were only studied for a particular type of feedback configuration, whereas here the completely general theory, i.e., one applicable to any feedback configuration is presented. The single-stage state probability sequence for a particular LFSR may be obtained very easily, and may be used as a monitoring heuristic to detect "reasonable" proximity to the asymptotic behaviour of the LFSR. The

theory developed for obtaining the single-stage state probabilities is later generalized to obtain exact or approximate *m-stage state probability sequences*. Exact m-stage state probability equations for a certain type of feedback configuration are derived, while bounds are derived for other commonly used configurations. In turn, this generalization to m stages yields exact values for the probability of aliasing for the former case, while it yields useful bounds for the latter case.

3.2 Computation of Single-Stage State Probability Sequences

3.2.1 Notation and Preliminaries

The feedback connections of a LFSR can be represented by polynomials in x with coefficients in the Galois field $GF(2)$ [Golomb 82] [Peterson 72]. For example, for the general LFSR depicted in Fig. 3.1, the *characteristic* polynomial $f(x)$ of the structure is:

$$f(x) = x^m \oplus h_{m-1}x^{m-1} \oplus h_{m-2}x^{m-2} \oplus \cdots \oplus 1, \tag{3.1}$$

where $h_i = 1$ if there is a feedback connection from stage x^i, and $h_i = 0$ otherwise.

Note that in this dissertation, only the *external* type of LFSR is strictly considered. The *internal* type of LFSR, i.e., the one where the output of the last stage is fed back to several of the earlier stages, is not formally considered. The reason is that because of the isomorphic relation that exists between the two types of structures [Hlawiczka 86], the analyses presented here can be shown to be applicable to either types.

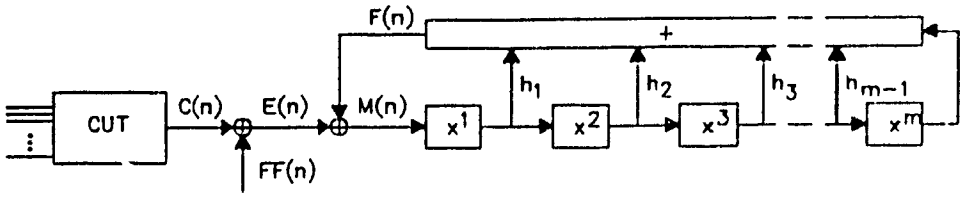

Figure 3.1 BIST scheme with a general signature analysis register

For analysis purposes, it has been shown in [David 78] that it is possible to consider the binary error sequence $E(n)$ only, where $E(n) = C(n) \oplus FF(n)$; $C(n)$ being the binary sequence produced by the CUT and $FF(n)$ the corresponding expected binary sequence of a fault-free circuit. Notice that in general, $E(n)$ is not explicitly produced in any BIST implementation. Also, in the error domain analysis, the signature register is always assumed to be initially in the "all-zero" state.

As shown in Fig. 3.1, let $E(n) = \{E_1, E_2, \ldots, E_n\}$ denote the incoming error sequence, $F(n) = \{F_1, F_2, \ldots, F_n\}$ denote the feedback sequence, and $M(n) = \{M_1, M_2, \ldots, M_n\}$ denote the modulo-2 sum sequence of $E(n)$ and $F(n)$, i.e., $M(n) = E(n) \oplus F(n)$. Assuming the "all-zero" state to be the initial condition of the LFSR, with E_i, F_i, and M_i denoting the i^{th} element of the respective sequences, M_i can be expressed as:

$$\begin{aligned} M_i &= E_i \oplus F_i \\ &= E_i \oplus \sum_{j=1}^{m} h_j M_{i-j}: \quad h_j = 0,1; \quad h_m = 1: \quad M_0 \ldots M_{-m+1} = 0; \quad i > 0; \end{aligned} \tag{3.2}$$

where the summation is modulo-2 addition. From here on in this dissertation, the expression "at time i" will denote the situation after the bit E_i has been shifted in the LFSR.

In the present context, the assumption is that the exact (deterministic) knowledge of the binary sequences $E(n)$, $M(n)$, and $F(n)$, is not available because such sequences can only be obtained through exact fault simulation. However, the bits E_i of the error sequence are assumed to have a constant probability of being equal to one called the probability of error ϵ, i.e., $pr(E_i = 1) = \epsilon$. In the same way that ϵ is a statistical descriptor for the sequence $E(n)$, ϵ can also become a descriptor for $M(n)$ since $E(n)$ and $M(n)$ are related through a feedback equation. The key for achieving the probabilistic description of $M(n)$ is to consider the E_i's of $E(n)$ as boolean variables and then use the concept of the probability of a boolean expression being true from [Parker 75], where proofs for the relationship between boolean operations and algebraic operations upon probabilities are provided. Based on this concept, a sequence of probabilities $\mu(n) = \{\mu_1, \mu_2, \ldots, \mu_n\}$ such that $\mu_i = pr(M_i = 1)$, is defined, and called the

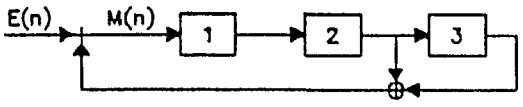

Figure 3.2 Simple LFSR with characteristic polynomial $f(x) = 1 + x^2 + x^3$

single-stage state probability sequence of the LFSR.

For example, consider the LFSR depicted in Fig. 3.2. For that register, assuming an "all-zero" state initial condition, the equations for the first six terms of the binary sequence $M(n)$ would be (see Section 3.2.2 for more details):

$$M_1 = E_1,$$

$$M_2 = E_2,$$

$$M_3 = E_3 \oplus M_1 = E_3 \oplus E_1,$$

$$M_4 = E_4 \oplus M_2 \oplus M_1 = E_4 \oplus E_2 \oplus E_1,$$

$$M_5 = E_5 \oplus M_3 \oplus M_2 = E_5 \oplus (E_3 \oplus E_1) \oplus E_2,$$

$$M_6 = E_6 \oplus M_4 \oplus M_3 = E_6 \oplus (E_4 \oplus E_2 \oplus E_1) \oplus (E_3 \oplus E_1) = E_6 \oplus E_4 \oplus E_3 \oplus E_2,$$

$$\vdots$$

Assuming that the binary sequence $M(n)$ is not known, the requirement is to generate the corresponding state probability sequence $\mu(n)$ for which the equations for the first six terms for the same example would be:

$$\mu_1 = pr(M_1 = 1) = \epsilon,$$

$$\mu_2 = pr(M_2 = 1) = \epsilon,$$

$$\mu_3 = pr(M_3 = 1) = 2\epsilon - 2\epsilon^2,$$

$$\mu_4 = pr(M_4 = 1) = 3\epsilon - 6\epsilon^2 + 4\epsilon^3,$$

$$\mu_5 = pr(M_5 = 1) = 4\epsilon - 12\epsilon^2 + 16\epsilon^3 - 8\epsilon^4,$$

$$\mu_6 = pr(M_6 = 1) = 4\epsilon - 12\epsilon^2 + 16\epsilon^3 - 8\epsilon^4,$$

$$\vdots$$

The "delay" property of binary sequences whereby the sequence that appears at the input of a particular stage appears at the input of the succeeding stage one time unit later also holds for the probability sequences. Therefore, in the way it was defined, $\mu(n)$ denotes the sequence that appears at the input of the stage corresponding to x^1 in the polynomial representation of the LFSR in Fig. 3.1. Because of the shifting property of LFSRs, clearly, the sequence of probabilities at the input of any succeeding stage x^j is simply the same sequence $\mu(n)$ shifted by j units in time, i.e., the sequence at the input of stage x^j is $\mu(n-j)$.

Readers familiar with the work presented in [David 78] might hurriedly conclude that the same ideas are being repeated here. However, this is not the case. Those familiar with the ideas presented in [David 78] will recall that the latter discussed the notion of a single-stage state probability sequence only for the polynomials $1 + x^m$. Here, the theory for single-stage state probability sequences for *any* linear polynomials, i.e., LFSRs characterized by any feedback configuration, is defined and established.

3.2.2 Calculation of μ(n)

Since an LFSR is constituted of memory elements and modulo-2 adders (XOR gates), the calculation of the single-stage state probability sequence rests principally on the calculation of the signal probability [Parker 75] of the output of an XOR gate function given the signal probabilities of the inputs. Assuming signal independence at the inputs, if the input signal probabilities of a two-input XOR gate are a and b, then the signal probability of the output c is [Parker 75]:

$$c = a + b - 2ab. \tag{3.3}$$

From the associative property of modulo-2 sums, for functions of more than two input variables, the above equation can be used iteratively.

To find $\mu(n)$ for a given LFSR, recall that the assumption is that each error sequence bit E_i is an independent event with possible outcome space 0 or 1, and probability ϵ that $E_i = 1$. By expressing each element M_i of the sequence $M(n)$ as

a modulo-2 sum of the independent error bits E_1 . . E_i, then each element μ_i of the probability sequence $\mu(n)$ can be generated using the equation for the two-input function (eq. (3.3)). Note that the expression for M_i must be minimal (irredundant), i.e., not contain more than one appearance of any error bit E_j. Subsequently, the minimal modulo-2 sum expression of M_i in terms of the independent bits E_1 . E_i is referred to as a *canonical* expression for M_i.

Recursively generating the canonical expressions for the elements of $M(n)$ is simple. Given the LFSR's feedback equation and the canonical expressions for $M_i \ \ldots \ M_{i-m+1}$, the canonical expression for the next sequence element, M_{i+1}, can be obtained by performing XOR operations, according to the feedback equation, between the m expressions corresponding to the previous sequence bits. Thus, to obtain the single-stage state probability sequence of length n recursively, it is required to generate linear expressions of up to n variables and to keep at least m of these expressions stored at all times to enable the generation of the subsequent expression. This implies a complexity of $O(mn^2)$ logical XOR operations and $O(mn)$ bits of storage. Finally, given the canonical expressions for the elements of $M(n)$, the recursive generation of the n elements of $\mu(n)$ requires a total of $O(n)$ floating point operations.

Despite the above, the space and time complexity for generating the canonical expressions for the elements of $M(n)$ can be proven to be less than what is claimed above. This is shown next.

The requirement is to generate canonical expressions of the type

$$M_i = \sum_{j=1}^{n} c_{ij} E_j; \ i = 1, \ \ldots, \ n; \ c_{ij} = 0, 1. \tag{3.4}$$

Letting the elements of the binary sequences $M(n)$ and $E(n)$ form n-element vectors, i.e., $\mathbf{M} = [M_1, M_2, \ \ldots, \ M_n]$ and $\mathbf{E} = [E_1, E_2, \ \ldots, \ E_n]$, and defining an $n \times n$ matrix of binary coefficients $\mathbf{C}$, then the two vectors can be related through the matrix equation

$$\mathbf{M} = \mathbf{CE}. \tag{3.5}$$

which in expanded form yields:

$$\begin{pmatrix} M_1 \\ M_2 \\ \vdots \\ M_n \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{pmatrix} \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{pmatrix}. \tag{3.6}$$

The matrix of binary coefficients C is hence what is required to generate all the canonical expressions for the elements of the sequence $M(n)$. Given this matrix representation of the canonical expressions for the elements of $M(n)$, the following lemma can be proven.

Lemma 3.1: For the matrix C corresponding to any LFSR, if the first column of coefficients $c_{11}, c_{21}, \ldots, c_{n1}$, is interpreted as a binary sequence, the latter corresponds to the sequence $M(n)$ obtained by initializing the LFSR to the "all-zero" state and to shift the error sequence $E(n) = \{1,0,0, \ldots, 0\}$ in the LFSR. Also, for any matrix C, $c_{ij} = 1$ for $i = j$, $c_{ij} = 0$ for $j < i$, and $c_{i+1j+1} = c_{ij}$ for $j > i$.

Proof: Since the first column of coefficients multiplies only the error bit E_1, the coefficient c_{i1}, determines whether bit E_1 is present or not in the canonical expression for M_i. The bit E_1 is shifted in the LFSR at time $i = 1$. Subsequently, i.e., for $i > 1$, the presence or absence of E_i in the canonical expression for M_i depends on whether E_1 "cancels" itself out, i.e., has odd or even parity. To determine this parity, consider the two possible cases:

1) $E_1 = 0$: The coefficients have no importance since $c_{ij}E_1 \equiv 0$. Therefore any sequence, for $c_{11}, \ldots, c_{n1}$ is valid;

2) $E_1 = 1$: It is clear that the parity of E_i, as a function of time i, corresponds exactly to $M(n)$ that results when the error sequence $E(n) = \{1,0,0, \ldots, 0\}$ is shifted in the LFSR.

From eq. (3.2), $M_i = E_i \oplus F_i$, where F_i is a function of the previous bits $E_1 \ldots E_{i-1}$ only, i.e., F_i is not a function of bits E_j, $j \geq i$. Therefore, clearly $c_{ij} = 0$ for $j < i$ and $c_{ij} = 1$ for $i = j$. By superposition, the scenario for bit E_1 also holds independently for the other error bits $E_2, \ldots, E_n$. Therefore, for $j > i$, the columns 2 ... n of

C follow from the first. That is, the sequence corresponding to the second column of coefficients is that of the first column shifted down one position (delayed by one time unit), with a 0 shifted in the vacant position (c_{12}) and truncated by one element (c_{n1}) at the bottom. The sequence for the third column is a shifted and truncated version of the second column, and similarly for the other columns. Hence, in general, $c_{i+1,j+1} = c_{ij}$

■

Shifting the error sequence $E(n) = \{E_1, E_2, \ldots, 0\} = \{1, 0, 0, \ldots, 0\}$ in the LFSR is equivalent to initializing the LFSR to the "100...000" state, i.e., initializing the stages associated with the different powers of x to $x^1 = 1, x^2 = x^3 = \ldots = x^m = 0$, followed by cycling the LFSR for $n-1$ cycles in the autonomous mode (with no external inputs). LFSRs cycled in the autonomous mode yield periodic sequences [Golomb 82]. Letting p be the period of the autonomous-mode sequence that results when the LFSR is initialized to the "100...000" state, the following theorem can be proven.

Theorem 3.2: The total time and space complexity required for generating the canonical expressions for all the elements of the sequence $M(n)$ is $O(min(n, p))$.

Proof: From Lemma 3.1, the matrix of coefficients C is characteristic of the LFSR only. All the information about C is contained in the first column, and by symmetry, in the last row as well. Moreover, as a result of the periodicity of the row and column sequences, it follows that all that is required for generating C and hence the canonical expressions for the elements of $M(n)$, is the simulation of the LFSR for $min(n, p)$ cycles where p is the period of the autonomous-mode sequence that results when the LFSR is initialized to the state "100...000". It follows that the space requirements to store the entire information contained in the required matrix C is $O(min(n, p))$ as well.

■

As an example, consider the following. If the particular LFSR depicted in Fig. 3.2 is initialized to the state "100", then in autonomous mode, the following sequence of states results ($p = 7$):

$$100, 010, 101, 110, 111, 011, 001, 100, \ldots . \tag{3.7}$$

Taking the leftmost bits of the sequence of states yields the sequence 10111001... . From this sequence, the following matrix of coefficients results for $n = 7$:

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix} \tag{3.8}$$

Hence, in this example, the generation of the coefficient matrix requires only the simulation of one period of the sequence of states that results when the LFSR is initialized to the "100" state. Furthermore, in this case, since the sequence 1011100 is a maximum-length sequence, for $n > 7$ one can augment the matrix C by simply concatenating this maximum-length sequence in the columns and rows of the matrix. Simple concatenation would also apply to non maximum-length sequences. However, it should be stressed that although the matrix C must implicitly be augmented to the dimensions $n \times n$ for sequences of length n, the matrix C of such dimensions never has to be explicitly generated due to its forementioned properties.

3.3 Examples and Heuristic Applications of Single-Stage State Probability Sequences

From the previous section, the single-stage state probability sequence of an LFSR is easy to generate. Figs. 3.3 - 3.8 contain plots of the sequences $\mu(n)$ obtained for different polynomials and different values of ϵ. The polynomial associated with Fig. 3.3 is non-primitive while those associated with Figs. 3.4 -3.8 are primitive. The polynomials associated with Figs. 3.5 and 3.6 are reciprocals of each other, where the reciprocal $g(x)$ of $f(x)$ is defined as [Peterson 72]:

$$g(x) = x^m f(x^{-1}). \tag{3.9}$$

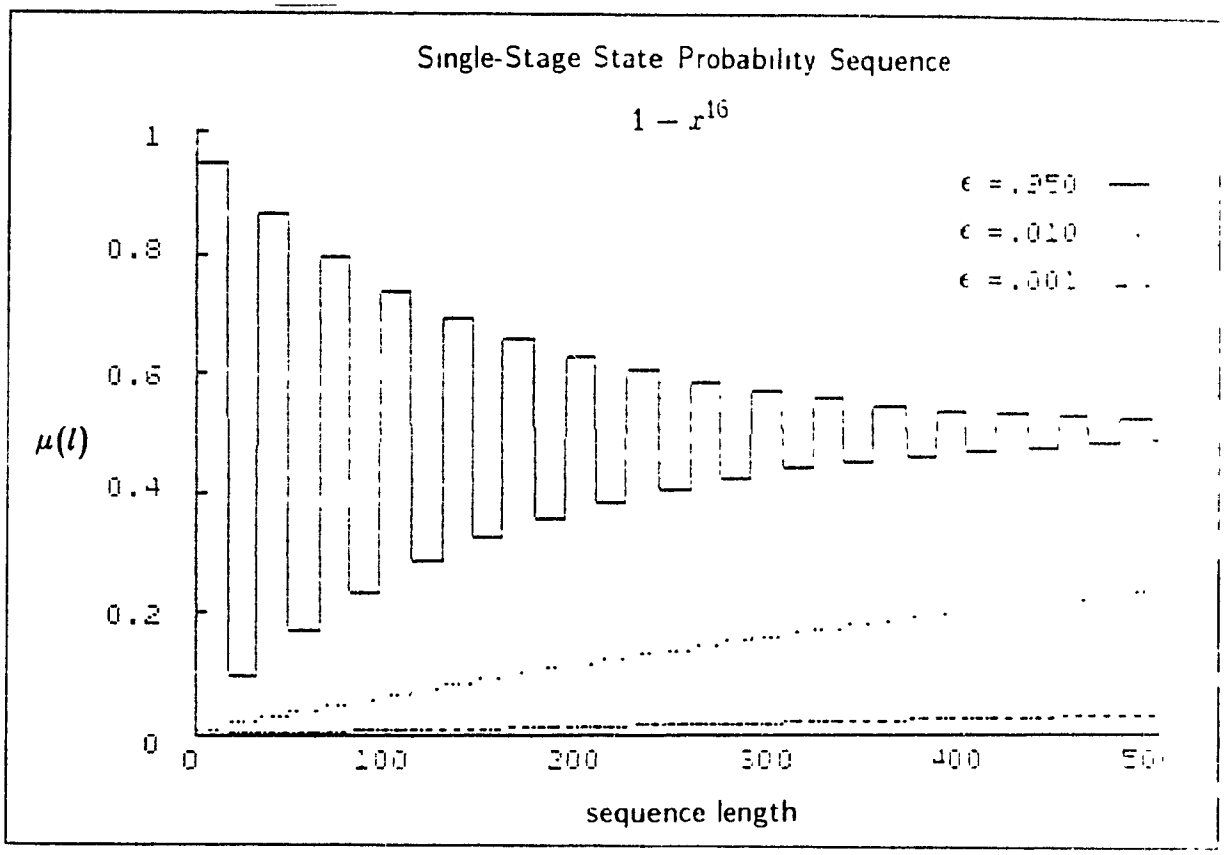

Figure 3.3 $\mu(n)$ for the polynomial $f(x) = 1 + x^{16}$

There are several possible heuristic applications of single-stage state probability sequences. Given their easy generation, they may be particularly helpful for making some important design decisions The dynamic behaviour of the single-stage state probability sequence can be interpreted as a first degree approximation to that of the m stages of the LFSR. As a result of its close approximation of the behaviour of the m stages of an LFSR, the single-stage state probability sequence can be used as a heuristic to determine a lower bound on the test sequence length necessary to "safely" assume the probability of aliasing to be close to 2^{-m}. For example, compare the single-stage state probability sequence plotted in Fig 3.8 with the exact probability of aliasing obtained for the polynomial $1 + x^7 + x^{10}$ plotted in Fig 4.3 Also, the single-stage state probability sequences can be used to compare the convergence rate of different polynomials. Hence, single-stage state probability sequences may be useful for choosing between different possible feedback configurations. More specifically, for the single-stage state probability sequences reported in Figs 3.3 -3.7, clearly, in all cases,

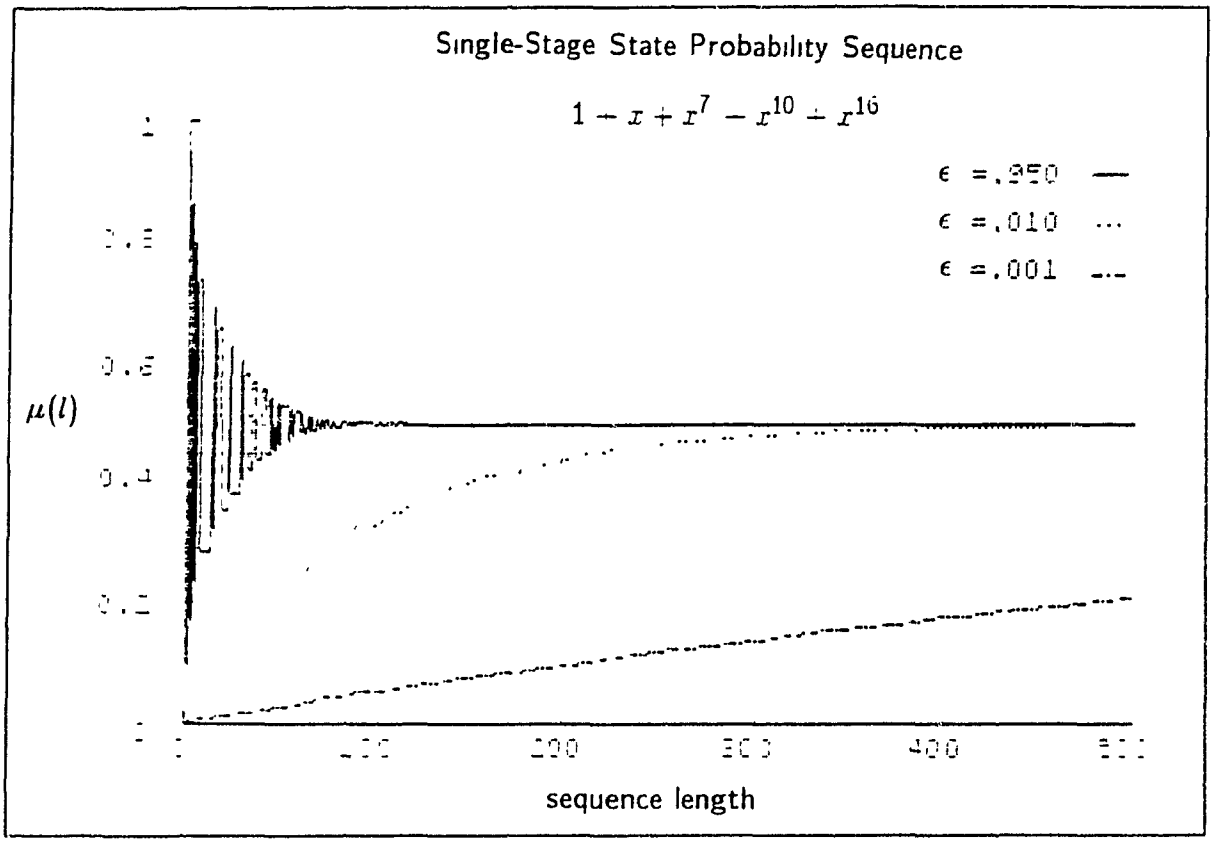

Figure 3.4 $\mu(n)$ for the polynomial $f(x) = 1 + x + x^7 + x^{10} + x^{16}$

a test length of 500 is insufficient to reach close "proximity" to the asymptotic value of 1/2 when $\epsilon = .001$. However, for $\epsilon = .950$ or $\epsilon = .010$, 500 patterns is sufficient to approach the asymptote quite closely in all cases except possibly for the polynomial $1 + x^6$. This is in agreement with Williams et al.'s claim [Williams 86] that primitive polynomials have a faster convergence rate to steady-state than do non-primitive polynomials. Another interesting behaviour to notice from the plots is that the two primitive polynomials that are reciprocals of each other do not converge to steady-state at the same rate. The polynomial in Fig. 3.6 converges faster than the one in Fig. 3.5. Furthermore, the plots reveal that the degree of the polynomials is not necessarily a predominant factor in the rate of convergence to steady-state; e.g., the polynomial of degree 50 converges at approximately the same rate as does the polynomials of degree 32 or 16. The type of feedback polynomial, e.g., primitive vs. non-primitive polynomial, is apparently more significant than the degree of the polynomial.

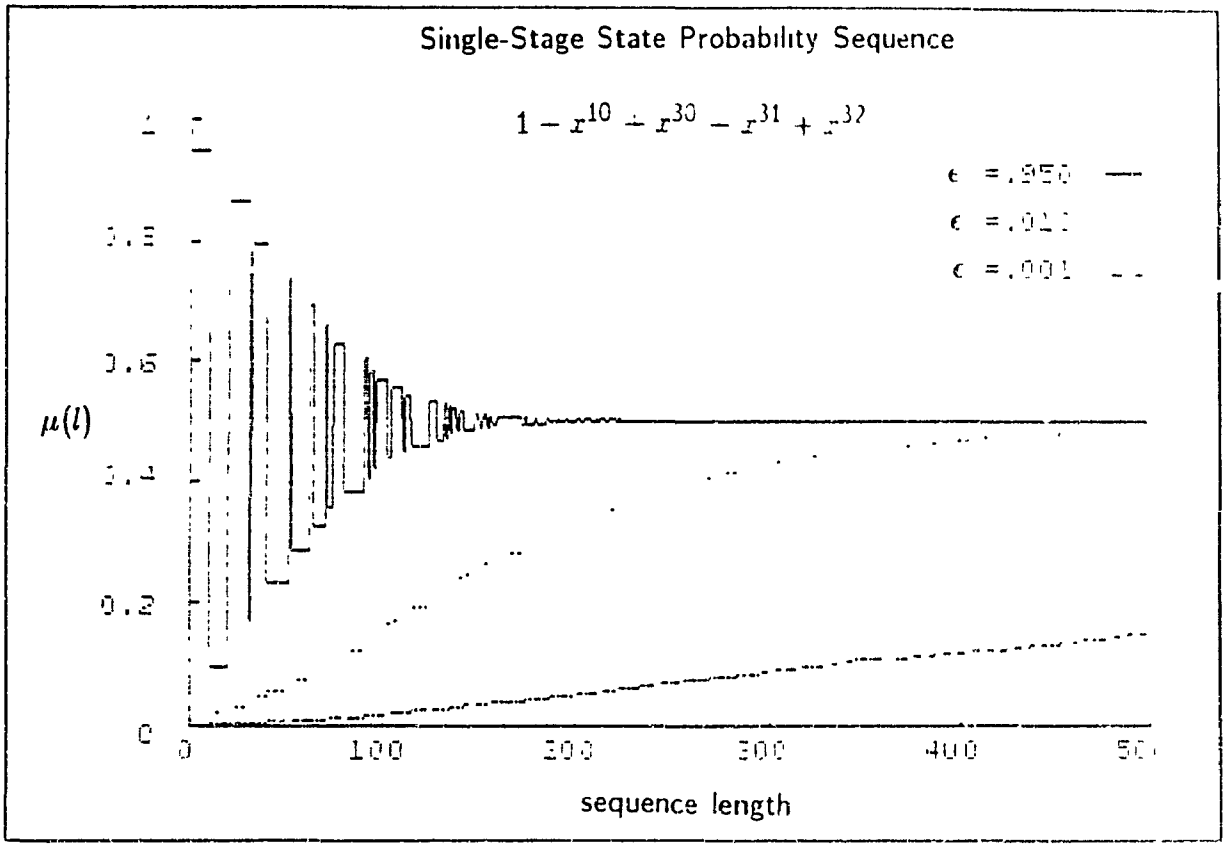

Figure 3.5 $\mu(n)$ for the polynomial $f(x) = 1 + x^{10} + x^{30} + x^{31} + x^{32}$

3.4 Multi-Stage State Probability Sequences

As shown in the previous section, single-stage state probability sequences have useful heuristic value. However, to claim stronger results, e.g., to derive the exact probability of aliasing or non-trivial bounds on the latter, not only one but all the stages of the register have to be considered *jointly*. Thus, in the framework developed here, to derive results that are quantitatively stronger than those provided by single-stage state probability sequences, *multi-stage state probability sequences* need to be studied. In particular, assuming registers with m stages, m-stage state probability sequences need to be studied.

3.4.1 Formulation as a Signal Probability Problem

Since a register of m stages has 2^m different possible states, a probability

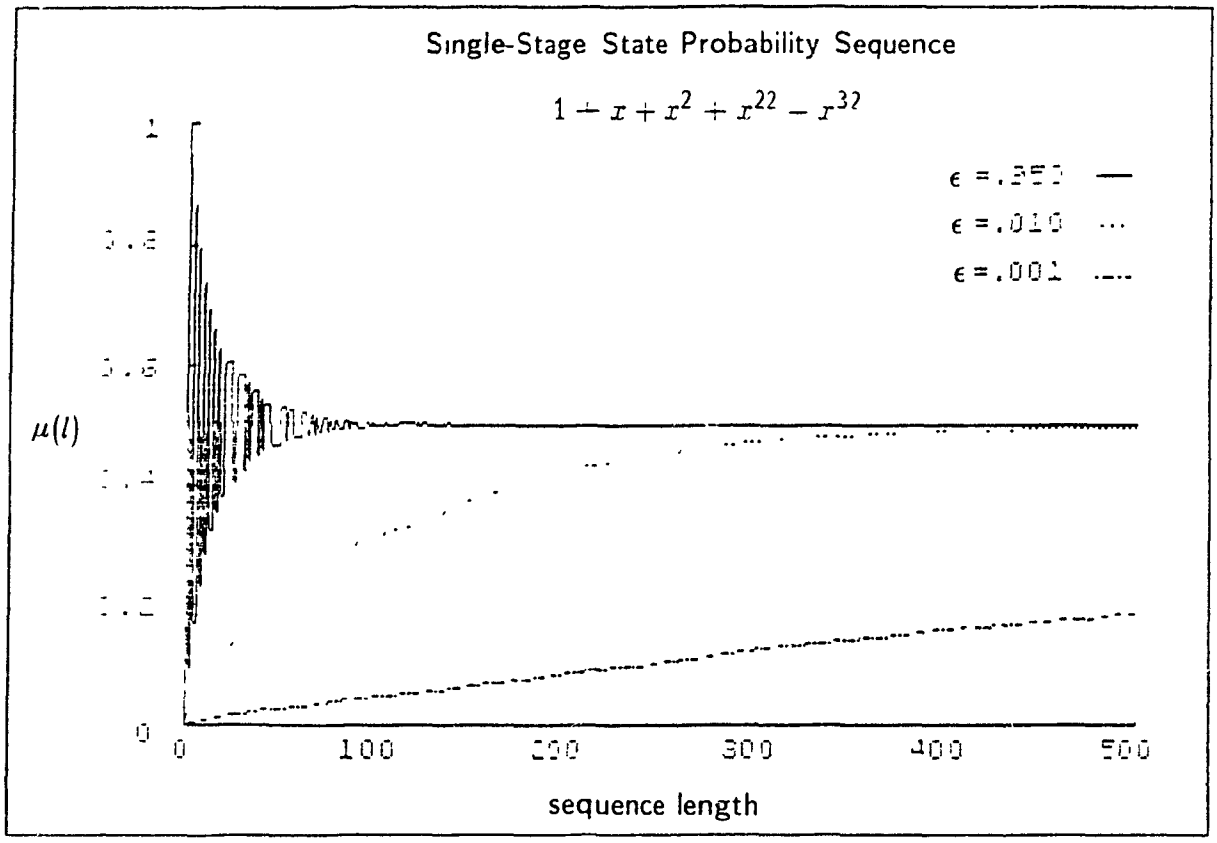

Figure 3.6 $\mu(n)$ for the polynomial $f(x) = 1 + x + x^2 + x^{22} + x^{32}$

sequence for each of these states can be defined. In the remainder of this section, the state of particular interest is the "all-zero" state. The reason for this emphasis is due to the importance (explained further in the next subsection) of this state when calculating the probability of aliasing. Let $Z(n)$ be the m-stage "all-zero" state probability sequence. Therefore, Z_n is the probability that the LFSR is in the "all-zero" state at time n, i.e., after n error bits have been shifted in the LFSR

The canonical equations for the elements of the binary sequence $M(n)$ constitute the basis from which the single-stage state probability sequence $\mu(n)$ is derived. These equations for $M(n)$ can also form the basis for jointly analyzing more than one stage Because of the shifting property of the LFSR, jointly analyzing several stages is equivalent to jointly analyzing several consequent values of a single stage. That is, examining the m canonical equations for the m stages at time n is equivalent to examining the equations for $M(n)$ at times $n - m + 1$ to n, i.e., $M_{n-m+1}, M_{n-m+2} \cdots, M_n$.

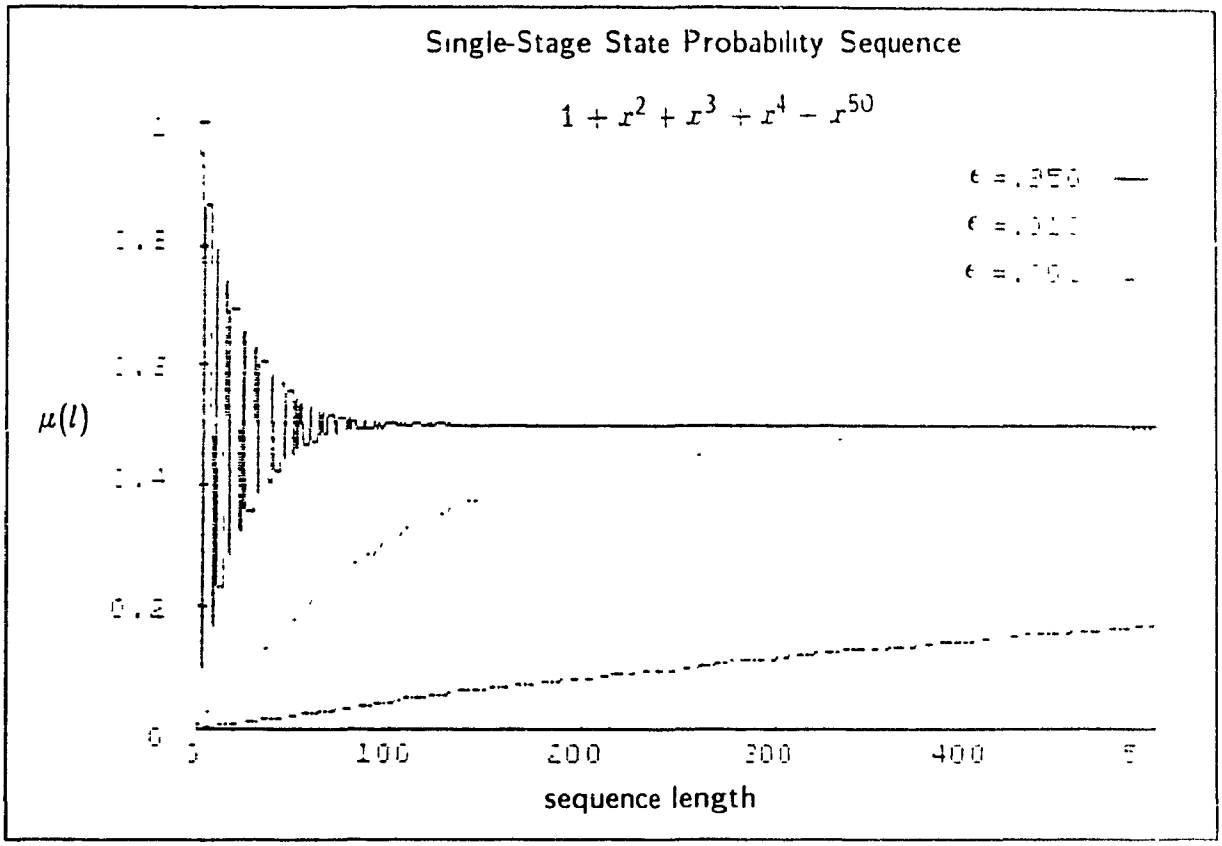

Figure 3.7 $\mu(n)$ for the polynomial $f(x) = 1 + x^2 + x^3 + x^4 + x^{50}$

In general, the m canonical equations $M_{n-m+1} \ldots M_n$ do not form decoupled equations. That is, the same error bit E_i may appear more than once in the m canonical equations. Therefore, in general, since the stage equations are not decoupled, the probability of the event of a stage being in either state 0 or 1 is not independent from the probability of the same event for the other stages. Therefore, for Z_n:

$$Z_n \neq \prod_{i=1}^{m} pr(stage\ i = 0) \tag{3.10}$$

or

$$Z_n \neq (1 - \mu_{n-m+1}) \cdots (1 - \mu_{n-1})(1 - \mu_n). \tag{3.11}$$

In general, given the canonical modulo-2 sum equations for the m stages at one instance in time, say n, and given the probability for individual input (error) bits to be one, i.e., given ϵ, solving for the probability of a particular state of the LFSR, e.g., "all-zero" state, at that instance in time can be considered as an instance of the

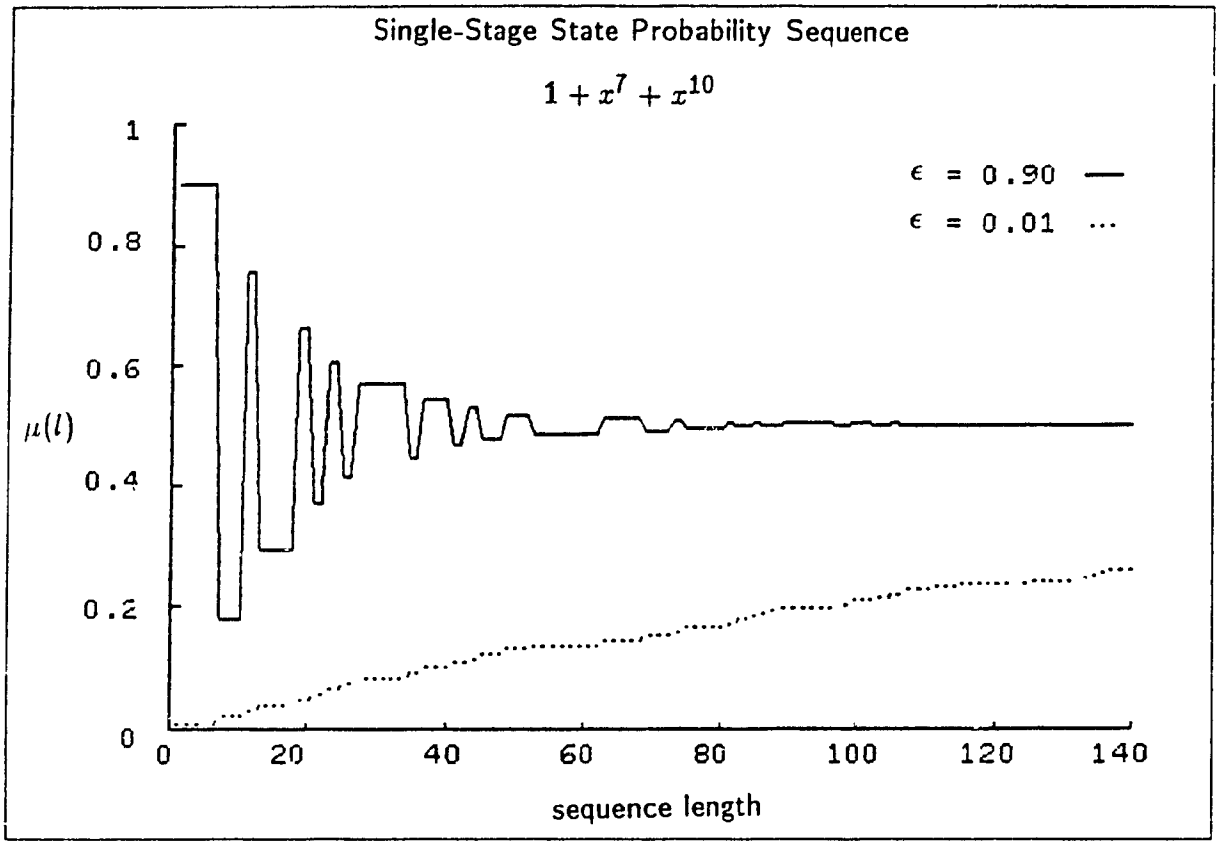

Figure 3.8 $\mu(n)$ for the polynomial $f(x) = 1 + x^7 + x^{10}$

standard problem of calculating a signal probability for a combinational circuit [Parker 75]. For example, solving for the probability of the "all-zero" state of the LFSR is equivalent to solving for the probability that the output G in the circuit depicted in Fig. 3.9 be 0, with the number of inputs to the circuit corresponding to the sequence length. Solving for the *all-ones* state would correspond to solving for the probability that the output G be one, with the OR gate of Fig. 3.9 replaced by an AND gate

Much work has been done in developing exact and approximate algorithms for estimating signal probabilities given boolean equations or gate-level descriptions of combinational circuits, e.g., [Parker 75] [Brglez 84] [Jain 85] [Seth 85]. For general networks, the problem of calculating the probability of a 1 or 0 at the circuit's output is part of a class of #P-complete problems [Valiant 79] [Krishnamurthy 86]. The complexity of #P-complete problems is conjectured to be worse than that of the well-known NP-complete problems, for which no polynomial-time algorithm is known to exist [Garey

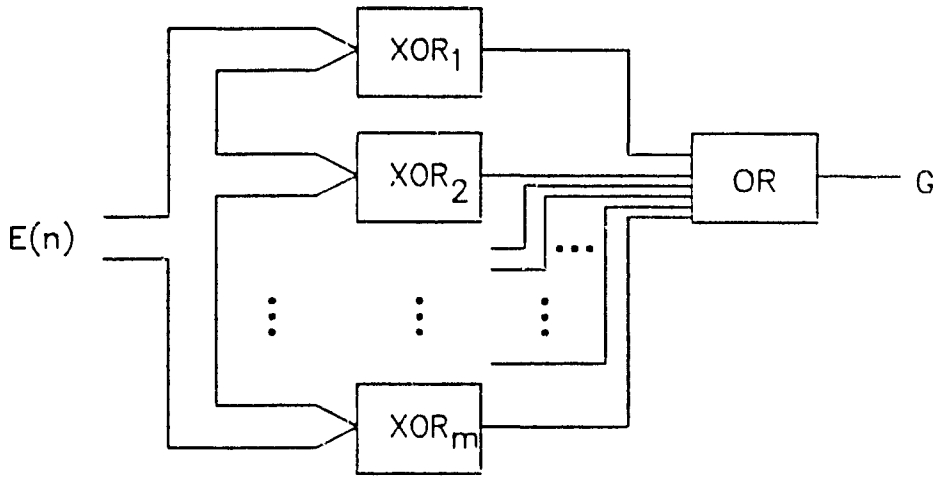

Figure 3.9 Example of the formulation of a state probability problem as a signal probability problem

78]. Hence, the #P-completeness of the signal probability problem for a combinational circuit implies that the best algorithms known for solving it require a time complexity at least exponential in the number of inputs to the circuit.

When formulated as a signal probability problem, solving for an element of the probability sequence $Z(n)$ requires finding the probability of a 1 or a 0 for a boolean formula expressed as a disjunction of m parity equations. The boolean satisfiability problem for general disjunctive formulae is solvable in polynomial time. Nevertheless, solving for the probability of a 1 or 0 for such formulae remains a #P-complete problem [Valiant 79] [Krishnamurthy 86]. † These observations could lead to conjecture that computing an element of $Z(n)$ is also #P-complete, and indeed, for the general case, the analysis technique presented in this chapter leads to solutions that are exponential in the sequence length. However, the solution described in the subsequent chapter of this dissertation proves that the problem can be solved with complexity less than

† As an alternative to the signal probability problem formulation of the problem of computing an element of $Z(n)$, given the modulo-2 sums expressions corresponding to each of the m stages for a sequence of length n, one straightforward solution approach is simply to expand the disjunction of these m expressions of $O(n)$ variables into a minimal sum of minterms. Since the minterms form mutually exclusive events, the probability that the disjunction of the m modulo-2 expressions will be 1 is simply the sum of the probabilities of the individual minterms. However, in general, the number of minterms required to express the disjunction is $O(2^n)$. Thus, this solution approach implies an $O(2^n)$ complexity.

exponential in the number of input variables, i.e., the sequence length (The solution described in Chapter 4 is exponential in the number of register stages and linear in the sequence length.) Thus, the solution presented in the subsequent chapter proves that the problem of computing an element of $Z(n)$ is not #P-complete.

Thus, a general and computationally feasible algorithm for the m-stage state probability sequences is described in Chapter 4. However, the solution in Chapter 4 does not follow directly from an extension of single-stage state probability sequences. Prior to describing this alternative analysis technique, in the subsequent subsections of this chapter, solutions for m-stage sequences (in particular $Z(n)$) that follow directly from extensions of single-stage state sequences are presented. These solutions are presented for two classes of polynomials: polynomials $1 + x^m$, and primitive polynomials.

3.4.2 Polynomials $1 + x^m$

LFSRs characterized by polynomials of the type $1 + x^k$ form an important class. Such LFSRs form the basis for a very interesting BIST scheme reported in [Krasniewski 87]. Such LFSRs turn out to be easy to analyze because of the simple structure that arises in the canonical equations for the m stages. The useful structure is the decoupling (independence) of the equations. In the gate-level signal probability formulation of the problem (Fig. 3.9), the decoupling of the equations corresponds to the inputs to the equivalent circuit not fanning out to several XOR gates.

A study of the use of LFSRs with characteristic polynomial $1 + x^m$ is reported in [David 78]. In the latter, one can find general expressions for the decoupled stage equations that arise for polynomials $1 + x^m$. Here, the form of the stage equations is illustrated through the matrix of coefficients C. For example, for the polynomial $1 + x^3$, assuming $n = 10$, C is:

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}. \quad (3.12)$$

From the above matrix, the following three equations are those required to analyze the 3-stage register after an error sequence of length $n = 10$ has been shifted in it:

$$M_8 = E_2 \oplus E_5 \oplus E_8,$$

$$M_9 = E_3 \oplus E_6 \oplus E_9,$$

$$M_{10} = E_1 \oplus E_4 \oplus E_7 \oplus E_{10}.$$

As mentioned earlier, consideration of the equations for the m stages at time n is equivalent to considering the equations for $M_{n-m+1} \ldots M_n$. This, in turn, is equivalent to considering the m adjacent rows $n \ldots n - m + 1$ in the matrix C. Examination of any m adjacent rows of the matrix C for the polynomials $1 + x^m$, reveals that there is only one "1" per sub-column of size m. Therefore, at any time, each bit E_i appears in the equation of only one of the m stages. It follows that for LFSRs characterized by polynomials $1 + x^m$, the joint probability of any state of the LFSR is simply the product of the state probabilities of the individual stages. Moreover, since the variables (error bits) are distributed evenly among the m stages, the number of variables in the equation for one stage may differ from that of another by at most one. More precisely, assuming n to be the length of the error sequence, the equation for $m - n \ mod \ m$ stages is a modulo-2 sum of exactly $\lfloor n/m \rfloor$ error bits, while that of $n \ mod \ m$ stages is the sum of $\lfloor n/m \rfloor + 1 = \lceil n/m \rceil$ bits, where $\lfloor\ \rfloor$ and $\lceil\ \rceil$ denote the mathematical floor and ceiling functions, respectively.

Assume the interest to be in calculating the probability of the "all-zero" state after a sequence of length n is shifted in the LFSR, i.e., the last term Z_n of the sequence

$Z(n)$ Given the number of error bits that appear in each stage's equation and given the probability ϵ that each of these bits is a 1, the next step to obtain an element of the sequence $Z(n)$ is to solve for the probability for one stage to be in the zero state. Because of the associative property of XOR functions, the probability that the logic value at the output Q of a q-input XOR gate be zero, given that the probability of any input being one is ϵ, can be solved recursively from the equation for a two-input case. Solving the recursion yields the following closed form expression:

$$pr(Q = 0) = \frac{1 + (1 - 2\epsilon)^q}{2}. \tag{3.13}$$

From the above expression and the independence property of the stage equations, it follows that for any polynomial $1 + x^m$, the general expression for the term Z_n is:

$$\begin{aligned} Z_n &= \prod_{i=1}^{m} pr(stage\ i = 0) \\ &= \prod_{j} pr(stage\ j = 0) \quad \prod_{k} pr(stage\ k = 0), \end{aligned} \tag{3.14}$$

where the product over j is over the $m - n \bmod m$ stages that have $\lfloor n/m \rfloor$ inputs, and the product over k is over the $n \bmod m$ stages that have $\lceil n/m \rceil$ inputs. Therefore, substituting eq. (3.13) into eq. (3.14),

$$\begin{aligned} Z_n &= \left[\frac{1 + (1 - 2\epsilon)^{\lfloor n/m \rfloor}}{2}\right]^{m-n \bmod m} \left[\frac{1 + (1 - 2\epsilon)^{\lceil n/m \rceil}}{2}\right]^{n \bmod m} \\ &= 2^{-m}\left[1 + (1 - 2\epsilon)^{\lfloor n/m \rfloor}\right]^{m-n \bmod m} \left[1 + (1 - 2\epsilon)^{\lceil n/m \rceil}\right]^{n \bmod m}. \end{aligned} \tag{3.15}$$

Here emphasis was placed on the computation of the probability of the "all-zero" state. However, because of the independence of the stage equations for polynomials $1 + x^m$, expressions for the probability of any of the 2^m possible states of the LFSR can easily be obtained.

3.4.3 Primitive Polynomials

LFSRs characterized by primitive polynomials [Peterson 72] form another interesting class, that are much used in BIST schemes both for test pattern generation

and compaction [Bardell 87]. The structure that arises in the canonical equations for the elements of $M(n)$ when the latter are generated from primitive polynomials permits the derivation of useful results. Primitive polynomials generate maximum-length sequences [Golomb 82]. Therefore, when initialized to the "100...000" state and cycled in the autonomous mode, the resulting sequence for all primitive polynomials of degree m has a period of length $2^m - 1$. Hence, the matrix of coefficients C of all primitive polynomials have a common property, tantamount to a considerable order in the canonical equations generated from primitive polynomials.

Assuming the length of an error sequence to be that of the maximum-length sequence of the LFSR, i.e., $n = 2^m - 1$, and assuming that the matrix C for this LFSR and for this n is generated, then, examination of the $2^m - 1$ sub-columns of length m that result from the m last rows of the matrix reveals that all these sub-columns are different. Examination of the corresponding scenario in the gate-level signal probability problem representation reveals that each of the $2^m - 1$ input bits E_i of the error sequence fans-out in a unique fashion. For example, consider the matrix C for the polynomial $f(x) = 1 + x^2 + x^3$, with $n = 7$, i.e., eq. (3.8). This polynomial is primitive and thus has a maximum-length period of 7. The sub-matrix constituted by the rows 5, 6, and 7 of C is reproduced below:

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}. \tag{3.16}$$

The seven columns of the above sub-matrix are all different. For this particular example, the corresponding gate-level signal probability representation of the canonical equations is shown in Fig. 3.10. Hence, as illustrated by this example, for primitive polynomials there exists a considerable order in the stage equations at times that are integral multiples of $2^m - 1$. This order is the basis for important results known in the context of coding theory; one such result being an important bound for the probability of undetected errors for cyclic Hamming codes [Lin 83] (see Section 5.3 of this dissertation for more discussion on coding theory). This bound for Hamming codes is used in [Gupta 88] to claim a result on the probability of aliasing in BIST for the cases where

the test sequence length corresponds to the natural length of the Hamming code. In the present context, this order enables an important upper bound on Z_n to be proven for any n. This bound is in turn useful for deriving an upper bound on the probability of aliasing for primitive polynomials. The bound on Z_n is expressed in the following theorem.

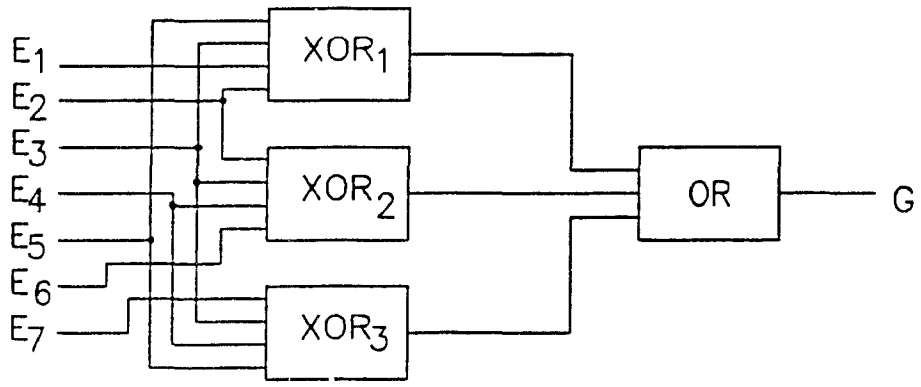

Figure 3.10 Gate-level representation for the polynomial $1 + x^2 + x^3$, $n = 7$

Theorem 3.3: Assuming a primitive feedback polynomial and an error sequence of length n, an upper bound for $Z_n = pr(\text{"all-zero" state at time } n)$ is:

$$Z_n \leq \left(\frac{1 + |1 - 2\epsilon|^{\lfloor n/m \rfloor}}{2}\right)^m. \tag{3.17}$$

where $|\ |$ denotes the "absolute value of". (For $n/m \gg 1$ this bound is essentially the envelope of eq. (3.15)).

Proof: See Appendix A. ∎

A useful application of this bound is discussed in the following subsection. Other applications are discussed in Chapter 5.

3.5 Probability of Aliasing

Aliasing occurs whenever the final signature of a faulty circuit is the same as that of the fault-free circuit. It can be shown that for this to occur, the final signature

of the faulty circuit must be the "all-zero" state when only the output *error* sequence of a CUT is considered [David 78]. Williams et al. used this observation in [Williams 86] to define the probability of aliasing at time n, PAL_n, as the probability of the signature analysis register starting in the "all-zero" state, leaving this state, and returning to it after an error sequence of length n is shifted in the signature register. Given the probability of the "all-zero" state, aliasing requires the exclusion of the case where the register never leaves this state [Williams 86], i.e., aliasing is not considered to occur if no errors occur in the output sequence. The probability of this occurring is $(1 - \epsilon)^n$. Therefore,

$$\begin{aligned} PAL_n &= pr(\text{"all-zero" state} \rightarrow \text{"all-zero" state}) - (1-\epsilon)^n \\ &= Z_n - (1-\epsilon)^n. \end{aligned} \tag{3.18}$$

The expression for Z_n derived for registers characterized by polynomials $1 + x^m$ (eq. (3.15)), was derived assuming that the register is initially in the "all-zero" state, i.e., $pr(Z_0 = 0) = 1$. Therefore, the equation (3.15) can be directly used for Z_n in eq. (3.18). Hence, the following exact equation for the aliasing probability for polynomials $1 + x^m$ results:

$$\begin{aligned} PAL_n = 2^{-m}[1 + (1-2\epsilon)^{\lfloor n/m \rfloor}]^{m-n \bmod m}\,[1 + (1-2\epsilon)^{\lceil n/m \rceil}]^{n \bmod m} \\ - (1-\epsilon)^n. \end{aligned} \tag{3.19}$$

Several plots of eq. (3.19) for different values of m and different values of ϵ were obtained. Fig. 3.11 contains plots for the polynomial $1 + x^3$, and Fig. 3.12 contains those for the polynomial $1 + x^8$. Such curves were reported in [Williams 86] and [Williams 87]. However, whereas the curves in [Williams 86] and [Williams 87] were obtained through simulations, given the closed-form analytic expressions, the generation of the curves reported here only required a computational effort linear in the test sequence length.

Regarding signature analysis registers characterized by primitive polynomials, from Theorem 3.3 and eq. (3.18), an upper bound for aliasing is:

$$PAL_n \leq 2^{-m}[1 + |1-2\epsilon|^{\lfloor n/m \rfloor}]^m - (1-\epsilon)^n \tag{3.20}$$

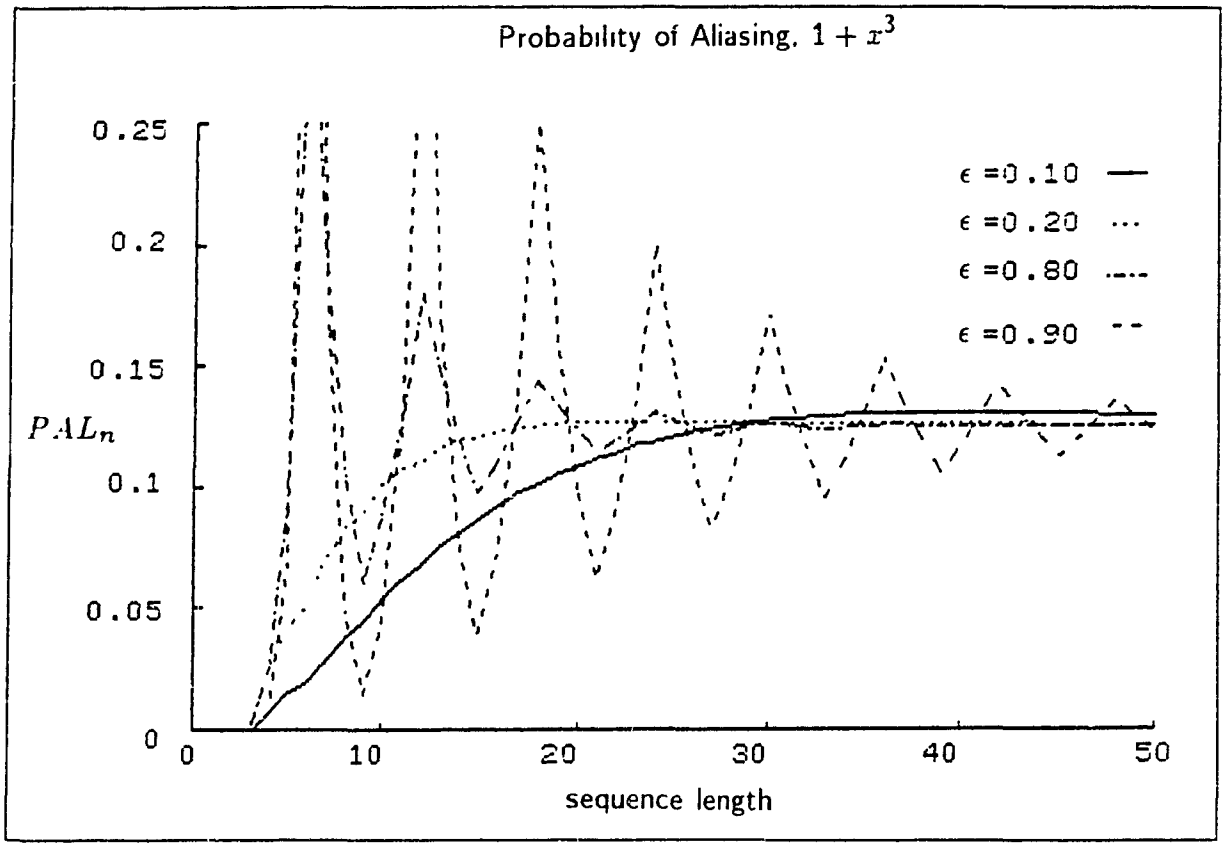

Figure 3.11 Probability of aliasing for polynomial $1 + x^3$.

In [Williams 87], the following bound for primitive polynomials is stated:

$$PAL_n \leq 1/2^m + (2^m - 1)(1 - 2\epsilon)^{n(1-1/(2^m-1))}. \tag{3.21}$$

Examples of Williams el al.'s bound (eq. (3.21)) and the one derived here (eq. (3.20)) were plotted. Bounds for primitive polynomials of degree 8 are shown in Fig. 3.13, and those for primitive polynomials of degree 16 and 15 appear in Figs. 3.14 and 3.15 respectively In Fig. 3.15, for purposes of comparison, the exact values for the probability of aliasing were plotted for the polynomial $1 + x + x^{15}$, along with the bounds although the technique used for obtaining the exact values is only described in the next chapter. Notice that in Fig 3.15, the logarithm to the base 10 of the probability of aliasing is given. In general, the bound derived here is tighter than Williams et al.'s bound for shorter test lengths, but is generally not as tight for longer test lengths. Equating eq. (3.21) with eq. (3.20), for $m >> 1$ and $\epsilon << 1$, the intersection point can be shown to be $n \approx 6m/10\epsilon$, and for ϵ approaching 1, the intersection point is

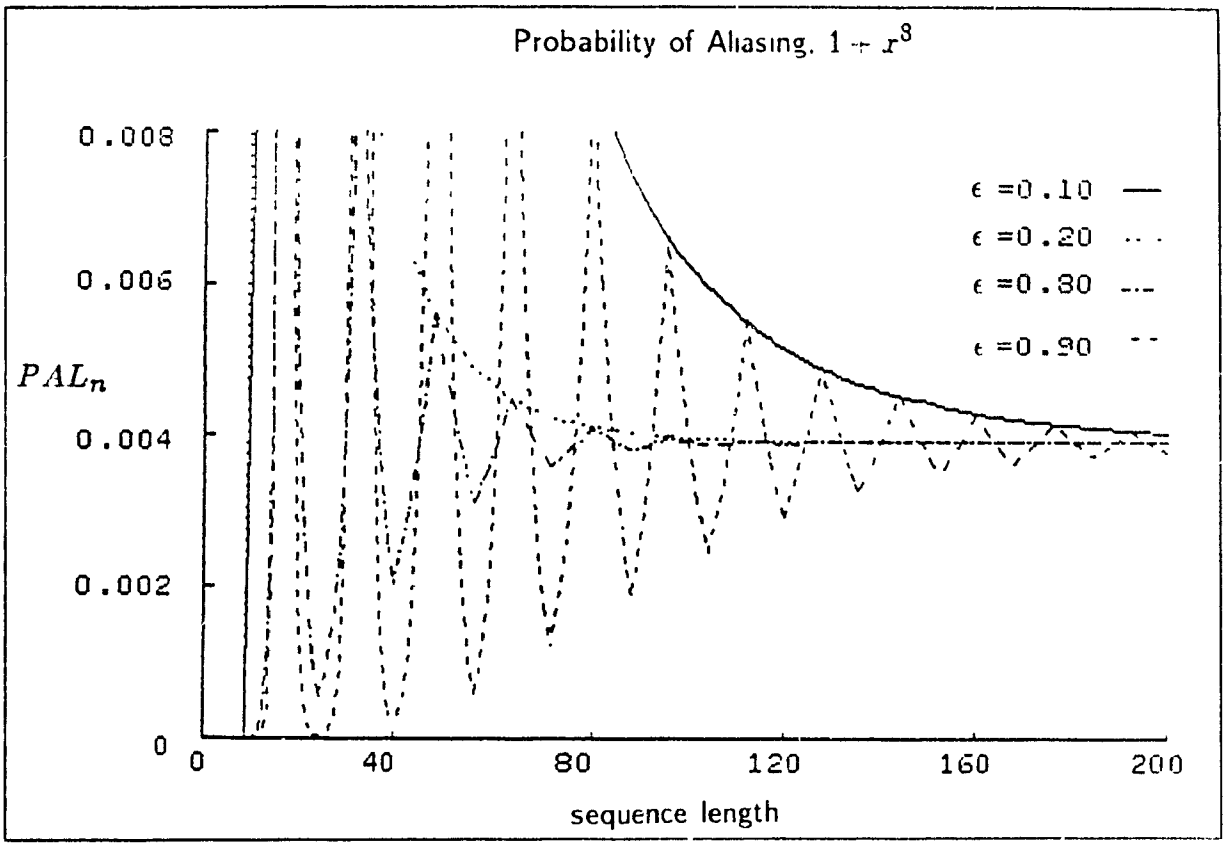

Figure 3.12 Probability of aliasing for polynomial $1 + x^8$

$n \approx 6m/10(1-\epsilon)$.

3.6 Summary

In this chapter, an analysis technique for studying the dynamic behaviour of LFSRs was proposed. The basis for the analysis is the single-stage state probability sequence of an LFSR. The computation of such sequences was shown to require a computational effort which is only linear in the sequence length. Though not presented here, it should be clear that because of the linearity of the operations performed by LFSRs, the simple superposition principle can be used to compute single-stage state probability sequences for multi-input LFSRs (MISRs) (See Section 5.2 for more details on multi-input LFSRs.)

Single-stage state probability sequences can serve as a first degree approxi-

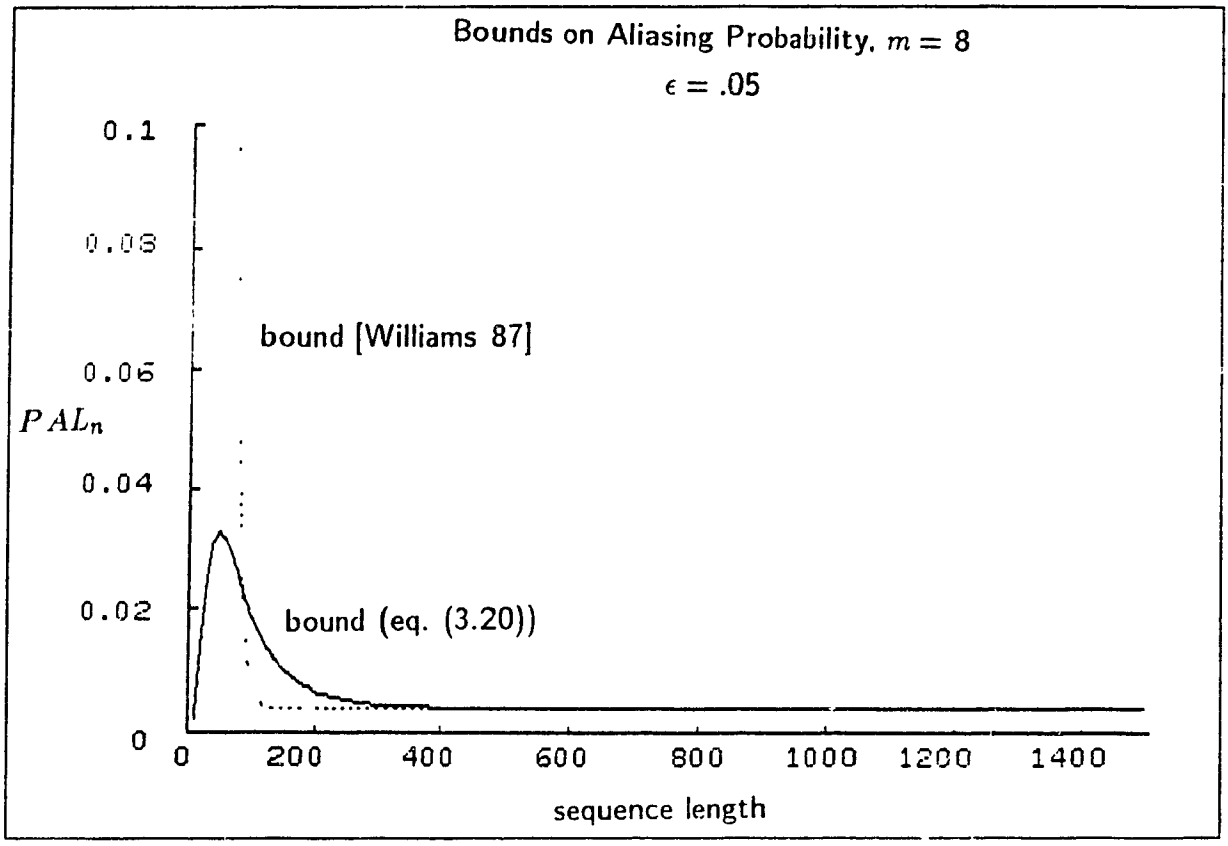

Figure 3.13 Bounds on the probability of aliasing for primitive polynomials of degree 8 "——" corresponds to eq (3.20) and " " corresponds to eq. (3.21).

mation for the behaviour of all m stages of an LFSR. Hence, single-stage state sequences may be useful in guiding several important BIST design decisions. The notion of single-stage state probability sequences was generalized to multi-stage state probability sequences. From this generalization, for LFSRs characterized by polynomials $1 + x^m$, exact closed-form expressions for the probability of aliasing were derived. In the case of LFSRs characterized by primitive polynomials, upper bounds were derived.

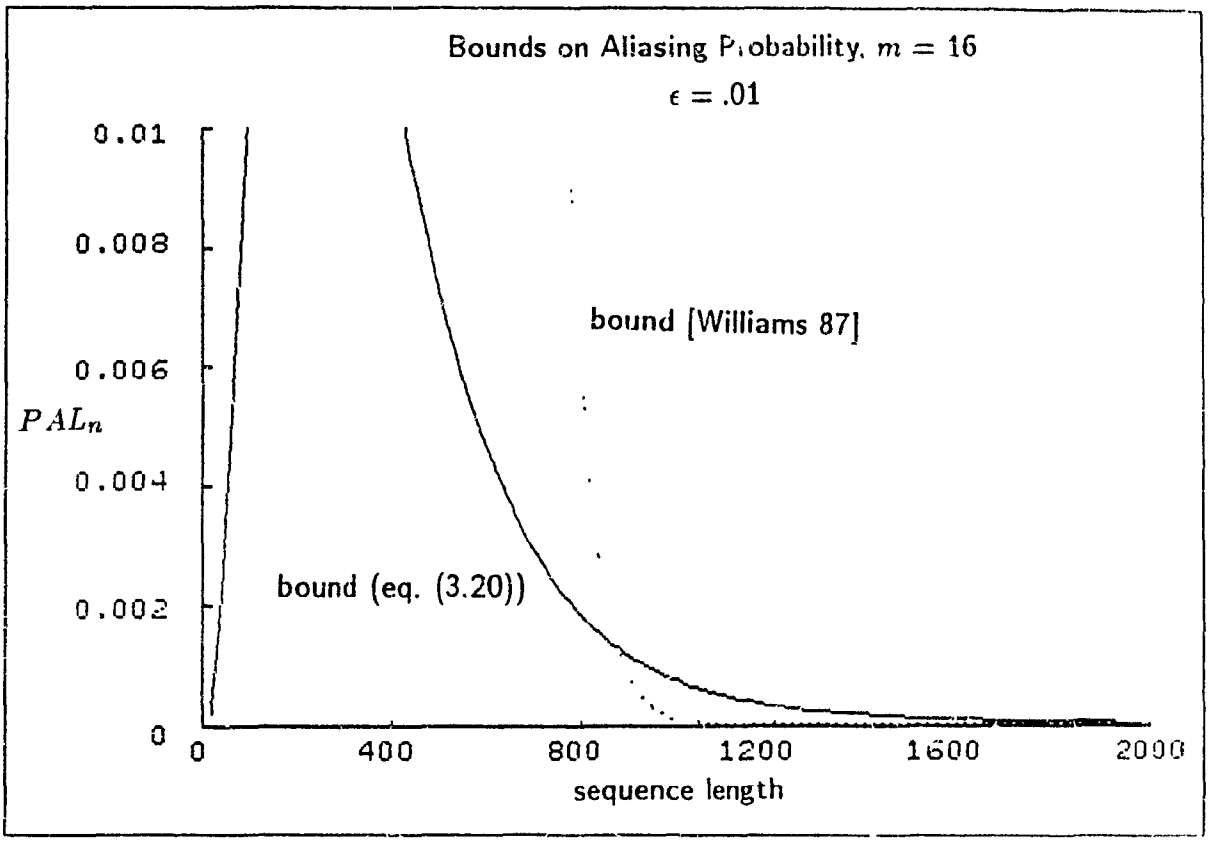

Figure 3.14 Bounds on the probability of aliasing for primitive polynomials of degree 16. "——" corresponds to eq. (3.20) and " " corresponds to eq. (3.21)

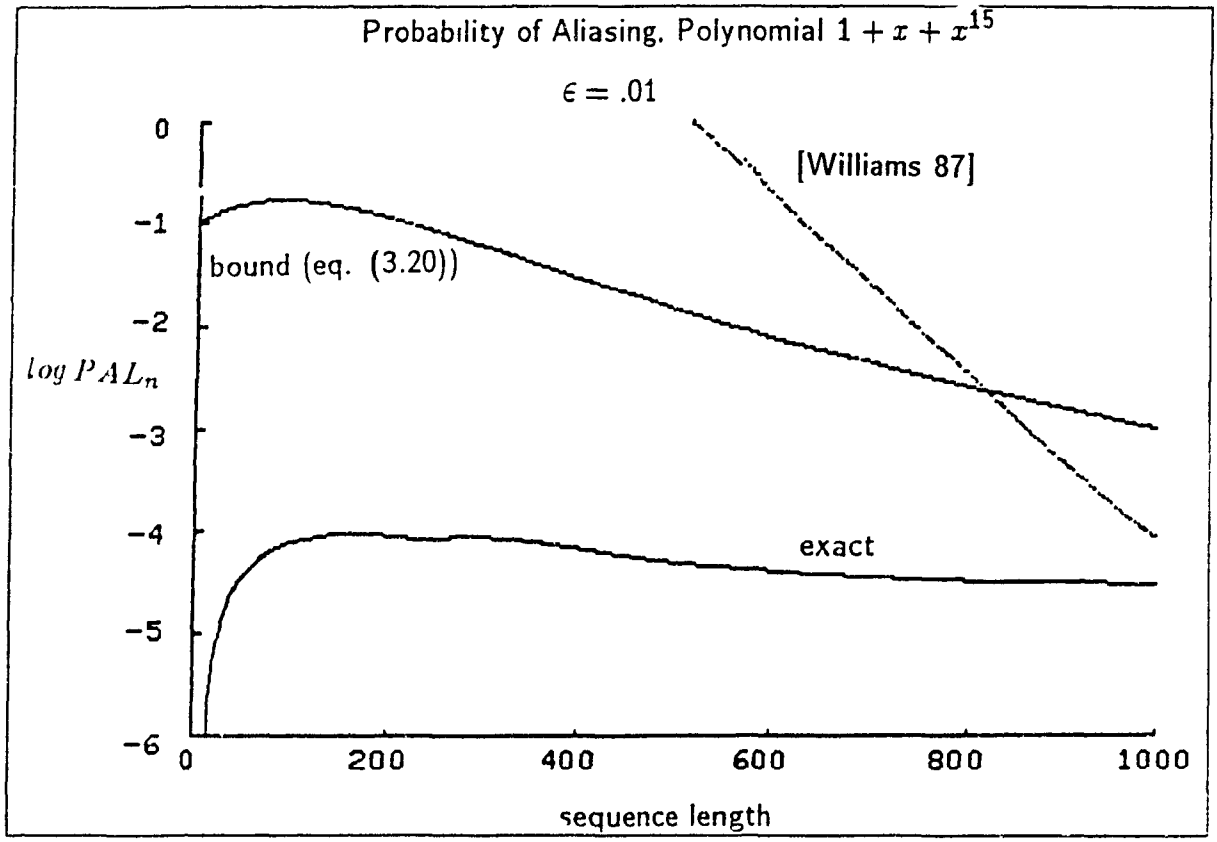

Figure 3.15 Bounds on the probability of aliasing for primitive polynomials of degree 15, and the exact probability of aliasing for the polynomial $1 + x + x^{15}$

Chapter 4

Iterative Technique for Calculating State Probabilities of LFSRs

4.1 Introduction

For LFSRs in general, the analytical technique developed in the preceding chapter has serious limitations. These are mostly due to the exponential (in sequence length) complexity that arises in the multi-stage analysis. In this chapter, a less complex technique for calculating the single- and multi-stage state probability sequences of LFSRs is presented. In essence, the technique is based on a recurrence for the probability of the "all-zero" state of an LFSR. From this recurrence, it is shown that the aliasing probability after a test sequence of length n is shifted in the LFSR can be calculated by exploiting the knowledge of the aliasing probability at length $n-1$. More precisely, the technique is based on a set of $2^m - 1$ double recurrences. The recurrence equations are double in that they express a recurrence in both the *time* and *space* dimensions, where the time dimension is already understood from the previous chapter, while the space dimension refers to the number of stages involved in the recurrence. Thus through these recurrence equations, it is shown that the probability that any set of up to m stages be simultaneously in the zero state at a given time step t can be calculated from the corresponding probability at the preceding time step $t-1$, and from that of a subset of the given set of stages at the same time step t. A compact notation is defined which enables each of these $2^m - 1$ recurrence equations to be expressed through only two

general equations. No closed-form solutions to these recurrences are given. Instead, the technique proceeds by performing exact numerical iterations from the set of recurrences.

For a register of size m, the technique requires $O(2^m)$ space and $O(n\ 2^m)$ time to yield all the $2^m - 1$ possible multi-stage state probability sequences. From these, the computation of the probability of aliasing for *all* values of test length from 1 to n follows directly. Thus, the technique in fact yields $2^m - 1$ solutions out of which only one (the "all-zero" state) is of particular interest in regard to the probability of aliasing. Hence, this technique provides a solution to the aliasing problem which is exponential in the number of stages, i.e., degree of the feedback polynomial, and linear in the sequence length, instead of being exponential in the sequence length. This is a very significant reduction in complexity compared to that of the technique developed in the preceding chapter which is exponential in the sequence length. The fact the the solution still requires a complexity exponential in the sequence length may appear to be a strong deterrent of its usefulness. This may be true, however, the computations required at each iteration step are simple enough to enable registers of sizes of interest in practice (e.g., 16) to be readily analyzed using today's computers, e.g., DEC's MicroVax.

Apart from its computational feasibility, another advantage of the solution presented in this chapter is that it is applicable to a very general error model. That is, the technique described in this chapter can handle distributions of error sequences other than the binomial or uniform distributions. More specifically, the iterative technique is applicable to the completely general error model wherein the probability of error on *each* of a CUT's output bit may vary. This enables the study of the performance of aliasing of signature analysis with respect to several interesting classes of errors, e.g., burst errors and dependent errors [Smith 80]. This facet of the technique also enables the study of several interesting BIST schemes. One example is the scheme where the probability of error changes as a result of changing the distribution of primary input probabilities of the random test pattern generator. Such a scheme has been shown to be very effective in reducing the test length required to achieve very high fault coverages [Lisanke 86] [Wunderlich 87] [Waicukauski 88].

This chapter describes the iterative technique in detail, and results obtained

using the technique are presented.

4.2 Preliminaries

The same notation as the one used in the preceding chapter is used in this chapter as well. For the sake of completeness of this chapter, the essentials of this notation are briefly repeated here, along with the introduction of further notation and preliminaries.

The feedback connections of an LFSR can be represented by polynomials in x with coefficients in the Galois field $GF(2)$ [Golomb 82] [Peterson 72]. For example, for the general LFSR depicted in Fig. 3.1, the *characteristic* polynomial $f(x)$ of the structure is:

$$f(x) = x^m \oplus h_{m-1}x^{m-1} \oplus h_{m-2}x^{m-2} \oplus \cdots \oplus 1, \tag{4.1}$$

where $h_i = 1$ if there is a feedback connection from stage i, and $h_i = 0$ otherwise.

For analysis purposes, only the error sequence $E(n)$ is treated explicitly. As shown in Fig. 3.1, let $E(n) = \{E_1, E_2, \ldots, E_n\}$ denote the incoming error sequence, $F(n) = \{F_1, F_2, \ldots, F_n\}$ denote the feedback sequence, and $M(n) = \{M_1, M_2, \ldots, M_n\}$ denote the modulo-2 sum sequence of $E(n)$ and $F(n)$, i.e., $M(n) = E(n) \oplus F(n)$. Assuming the "all-zero" state to be the initial condition of the LFSR, with E_i, F_i, and M_i denoting the i^{th} element of the respective sequences, M_i can be expressed as:

$$\begin{aligned} M_i &= E_i \oplus F_i \\ &= E_i \oplus \sum_{j=1}^{m} h_j M_{i-j}; \quad h_j = 0,1; \quad h_m = 1; \quad M_0 \ldots M_{-m+1} = 0; \quad i > 0; \end{aligned} \tag{4.2}$$

where the summation is modulo-2 addition.

Here the assumption is that the exact (deterministic) knowledge of the binary sequences $E(n)$, $M(n)$, and $F(n)$, is not available because such sequences can only be obtained through exact fault simulation. Instead, the assumption is that the bits E_i of the error sequence each have a probability ϵ_i of being equal to one, i.e., $pr(E_i = 1) = \epsilon_i$.

Recall that in Chapter 3, each error bit E_i was assumed to have the same probability of error, i.e., ϵ was assumed to be constant throughout the entire test sequence. Now however, each error bit is assumed to be characterized by a specific probability ϵ_i.

The following is new notation introduced because it facilitates the description of the iterative technique that will be described next.

$\mathbf{S}_i$ is defined as a vector representing the state of the LFSR at time i:

$$\mathbf{S}_i = [S_i^0, S_i^1, \ldots, S_i^{m-2}, S_i^{m-1}]. \tag{4.3}$$

where $S_i^j = M_{i+j-(m-1)}$. Therefore,

$$\mathbf{S}_i = [S_i^0, S_i^1, \ldots, S_i^{m-1}] = [M_{i-m+1}, M_{i-m+2}, \ldots, M_i]. \tag{4.4}$$

By the shifting property of the LFSR, for $0 \leq j < m-1$,

$$S_{i+1}^j = S_i^{j+1}; \tag{4.5}$$

and for $j = m-1$ (using eq.(4.2)),

$$\begin{aligned} S_{i+1}^{m-1} = M_{i+1} &= E_{i+1} \oplus \sum_{j=1}^{m} h_j M_{i+1-j} \\ &= E_{i+1} \oplus \sum_{j=1}^{m} h_j S_i^{m-j}. \end{aligned} \tag{4.6}$$

Prior to formally introducing the problem of aliasing and the iterative technique using the newly introduced notation, the following two lemmas are stated since they are used in the demonstration of the technique presented in Section 4.3.

Lemma 4.1: Let A and B be two boolean expressions, and let $\cup$ denote the boolean OR operator [†] and $\oplus$ denote modulo-2 addition, then

$$(A \oplus B) \cup A = A \cup B.$$

[†] The reason for using the symbol $\cup$ to denote the boolean OR operator is to save the symbol + to denote regular arithmetic addition, thereby avoiding confusion in the forthcoming equations where both boolean and arithmetic operations are expressed simultaneously.

Proof: From the definition of boolean connectives. ∎

Lemma 4.2: Let A and B be two boolean expressions, and let $\overline{A}$ denote the boolean complementation of A, then,

$$pr(\overline{A} \cup B = 1) = 1 - pr(A \cup B = 1) + pr(B = 1)$$

Proof: Follows from the corresponding Venn diagrams. ∎

4.3 Computing State Probability Sequences of LFSRs

4.3.1 Motivation and Basis for the Technique

Here the primary motivation for computing the state probabilities of LFSRs is to determine the probability of aliasing for a given test sequence length and a given LFSR. The problem of aliasing has already been discussed in preceding chapters. From before, the probability of aliasing PAL_n is defined as the probability of the signature register starting in the "all-zero" state, leaving this state, and returning to it after having shifted in the LFSR an error sequence of length n. With the new notation introduced in Section 4.2, starting in the state $S_0 = 0$ is synonymous with starting in the "all-zero state". Similarly, returning to this state at time n is expressed by $S_n = 0$. Assuming that the probability of error ϵ is constant, excluding the case where the register never leaves the initial "all-zero" state, then,

$$PAL_n = pr(S_n = 0) - (1 - \epsilon)^n. \tag{4.7}$$

Let $S(n)$ denote the sequence of states of the LFSR. That is, let the i^{th} element of $S(n)$, i.e., S_i denote the state that the LFSR is in after i error bits have been shifted in the LFSR. To find PAL_n, the interest is in the "all-zero" state. That is, the requirement is to find $pr(S_n = 0)$. To simplify notation, Z_i is defined such that $Z_i = pr(S_i = 0)$ and $Z_i' = 1 - Z_i$. Then, expressing $S_i = 0$ through its components yields,

$$Z_i = pr(M_i \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} = 0) \tag{4.8}$$

From the preceding chapter, finding $pr(M_i = 0)$ corresponds to the single-stage analysis and is hence easy for all feedback polynomials. Also from the previous chapter, finding $pr(S_i = 0)$, which corresponds to an m-stage analysis, is straightforward for polynomials $1 + x^m$. However, finding $pr(S_i = 0)$ for any polynomial, in particular primitive polynomials, is not straightforward due to the dependencies that exist between the sequence elements $M_i, M_{i-1}, \ldots, M_{i-m+1}$. However, in this chapter, it is shown how the multi-stage probability sequence $pr(S_1 = 0), pr(S_2 = 0), \ldots, pr(S_n = 0)$ can be obtained iteratively, for any polynomial. Before describing this technique in detail, a theorem which constitutes the basis for it is proven.

Theorem 4.3:

$$Z_i = \epsilon_i + Z_{i-1}(1 - 2\epsilon_i) - \epsilon_i pr(M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} = 1).$$

Proof: Using the definitions for the probability of boolean expressions from [Parker 75],

$$\begin{aligned} Z_i &= pr(M_i \cup M_{i-1} \cup \cdots \cup M_{i-m+1} = 0) \\ &= 1 - pr(M_i \cup M_{i-1} \cup \cdots \cup M_{i-m+1} = 1). \end{aligned} \tag{4.9}$$

By definition, $Z_i = 1 - Z_i'$, therefore,

$$Z_i' = pr(M_i \cup M_{i-1} \cup \cdots \cup M_{i-m+1} = 1). \tag{4.10}$$

Substituting eq. (4.2) into eq. (4.10) yields

$$Z_i' = pr\Big((E_i \oplus \sum_{j=1}^{m} h_j M_{i-j}) \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} = 1\Big) \tag{4.11}$$

where the summation is modulo-2. Using Lemma 4.1 repeatedly, i.e., as many times as there are non-zero h_i's in the characteristic polynomial of the LFSR, yields †

$$Z_i' = pr\Big((E_i \oplus M_{i-m}) \cup M_{i-1} \cup \cdots \cup M_{i-m+1} = 1\Big). \tag{4.12}$$

---

† Recently, an observation was made to the effect that eq. (4.12) could be obtained from first principles [Krishnamurthy 88]. The argument is the following. If any of the bits $M_{i-1} \ldots M_{i-m+1}$ is a one after the i^{th} bit is shifted in, then S_i will be one. Otherwise, i.e., if all $M_{i-1} \ldots M_{i-m+1}$ are zero, S_i will be one if $(E_i \oplus M_{i-m})$ is a one.

Given a boolean function f of n variables $x_1, x_2, \ldots, x_n$. Shannon's expansion theorem [Kohavi 70] states that $f(x_1, x_2, \ldots, x_n)$ can be expressed as a boolean sum (here denoted by the symbol $\cup$) of two terms:

$$\begin{aligned} f(x_1, x_2, \ldots, x_n) &= \overline{x}_i f(x_1, \ldots, x_i, \ldots, x_n)|_{x_i=0} \cup x_i f(x_1, \ldots, x_i, \ldots, x_n)|_{x_i=1} \\ &= \overline{x}_i f(x_1, \ldots, 0, \ldots, x_n) \cup x_i f(x_1, \ldots, 1, \ldots, x_n). \end{aligned} \tag{4.13}$$

Therefore, performing such an expansion for the expression (function)

$$\left((E_i \oplus M_{i-m}) \cup M_{i-1} \cup \cdots \cup M_{i-m+1} = 1 \right) \tag{4.14}$$

yields the following for eq. (4.14):

$$\begin{aligned} &\overline{E}_i \left((E_i \oplus M_{i-m}) \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m-1} \right)|_{E_i=0} \\ &\cup \\ &E_i \left((E_i \oplus M_{i-m}) \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} \right)|_{E_i=1} \end{aligned} \tag{4.15}$$

The expression in eq. (4.15) can be rewritten as

$$\begin{aligned} &\overline{E}_i \left((0 \oplus M_{i-m}) \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} \right) \\ &\cup \\ &E_i \left((1 \oplus M_{i-m}) \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} \right) \end{aligned} \tag{4.16}$$

Since the two terms of eq. (4.16) constitute mutually exclusive events, their probabilities may be summed. Therefore, from eq. (4.12) and the above,

$$\begin{aligned} Z_i' = &\, pr\left[\overline{E}_i \left((0 \oplus M_{i-m}) \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} \right) = 1 \right] \\ &+ pr\left[E_i \left((1 \oplus M_{i-m}) \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} \right) = 1 \right] \end{aligned} \tag{4.17}$$

Furthermore, since E_i is independent of $M_{i-1} \ldots M_{i-m}$, eq. (4.17) can be rewritten as

$$\begin{aligned} Z_i' = &\, pr(E_i = 0)\ pr\left[(0 \oplus M_{i-m}) \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} = 1 \right] \\ &+ pr(E_i = 1)\ pr\left[(1 \oplus M_{i-m}) \cup M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m+1} = 1 \right] \end{aligned} \tag{4.18}$$

Since $pr(E_i = 1) = \epsilon_i$ and $pr(E_i = 0) = 1 - \epsilon_i$,

$$\begin{aligned} Z_i' = &\, (1 - \epsilon_i)\ pr(M_{i-1} \cup M_{i-2} \cup \cdots \cup M_{i-m} = 1) \\ &+ \epsilon_i\ pr(M_{i-1} \cup \cdots \cup M_{i-m+1} \cup \overline{M}_{i-m} = 1). \end{aligned} \tag{4.19}$$

Using Lemma 4.2 and eq. (4.10),

$$\begin{aligned} Z'_t &= (1-\epsilon_t)Z'_{t-1} \\ &\quad + \epsilon_t\Big[1 - Z'_{t-1} + pr(M_{t-1} \cup \cdots \cup M_{t-m+1} = 1)\Big] \\ &= \epsilon_t + Z'_{t-1}(1-2\epsilon_t) + \epsilon_t pr(M_{t-1} \cup \cdots \cup M_{t-m+1} = 1). \end{aligned} \tag{4.20}$$

Substituting $Z'_t = 1 - Z_t$ in the above yields:

$$Z_t = \epsilon_t + Z_{t-1}(1-2\epsilon_t) - \epsilon_t pr(M_{t-1} \cup M_{t-2} \cup \cdots \cup M_{t-m+1} = 1). \tag{4.21}$$

■

Theorem 4.3 expresses a recursion in two dimensions: in time and in space, where time refers to the sequence length while the space dimension refers to the number of stages involved. Theorem 4.3 states that it is possible to calculate Z_t from ϵ_t, Z_{t-1} and $pr(M_{t-1} \cup \cdots \cup M_{t-m+1} = 1)$. Expressing Z_t from Z_{t-1} constitutes the recursion in the time dimension, while expressing Z_t (which involves the m stages) from $pr(M_{t-1} \cup \cdots \cup M_{t-m+1} = 1)$ constitutes the recursion in the space dimension since $pr(M_{t-1} \cup \cdots \cup M_{t-m+1} = 1)$ is a subproblem of the original problem, i.e., $pr(M_{t-1} \cup \cdots \cup M_{t-m+1} = 1)$ is the probability that only $m-1$ instead of m stages not be simultaneously zero at time t. Just as Theorem 4.3 expresses Z_t in terms of itself at time $t-1$, and the probability of a disjunction of a fewer number of stages at time t, it is possible to express $pr(M_{t-1} \cup \cdots \cup M_{t-m+1} = 1)$ in terms of itself at time $t-1$ and the probabilities of different disjunctions at time $t-1$, and similarly for the other possible disjunctions of m stages. Thus, if the probabilities of all the possible disjunctions that can be formed from the set of the m states of the corresponding m stages of an LFSR are known at time $t-1$, then it is possible to compute the corresponding probabilities for the subsequent time t. This is shown formally in the next subsection.

4.3.2 Calculating Probability Sequences of Disjunctions of States of LFSR Stages

In this subsection a general iterative technique for calculating the probability sequences of all the possible disjunctions of the states of the m stages of an LFSR for all

sequence lengths is presented. Using Theorem 4.3 and eq. (4.7), this technique enables the exact probability of aliasing to be calculated for all test sequence lengths

If each component S_t^j of the state vector S, of an LFSR of length m is considered as a boolean variable, then there are 2^m-1 possible non-null disjunctions that can be formed with these variables. To distinguish the different possible disjunctions using a compact notation, let U_t^a, $1 \leq a \leq 2^m - 1$ denote the $2^m - 1$ particular disjunctions, where U_t^a is defined as follows:

$$U_t^a = \bigcup \alpha_j S_t^j, \quad 0 \leq j \leq m-1; \tag{4.22}$$

where $\alpha_j = 0,1$ depending on whether the particular variable S_t^j is part of the disjunction or not, and the space index (superscript) a of U_t^a is formed according to the following:

$$a = \sum_{j=0}^{m-1} \alpha_j 2^j, \tag{4.23}$$

where the summation and exponentiation of 2 are the regular operations over the field of real numbers. Therefore,

$$\begin{aligned} U_t^1 &= S_t^0; \\ U_t^2 &= S_t^1; \\ U_t^3 &= S_t^0 \cup S_t^1; \\ U_t^4 &= S_t^2; \\ U_t^5 &= S_t^0 \cup S_t^2; \\ \vdots &= \vdots \\ U_t^{2^m-1} &= S_t^0 \cup S_t^1 \cup S_t^2 \cup \ldots \cup S_t^{m-1}. \end{aligned} \tag{4.24}$$

This assignment of space indices (superscripts) enables efficient bitwise logic operations to be performed on their binary representations. For example,

$$U_t^3 \cup U_t^5 = (S_t^0 \cup S_t^1) \cup (S_t^0 \cup S_t^2) = S_t^0 \cup S_t^1 \cup S_t^2 \tag{4.25}$$

Letting 0011 and 0101 be the binary representations of the space indices of U_t^3 and U_t^5, respectively, and letting $\vee$ denote the bitwise OR operation, then the index of $U_t^3 \cup U_t^5$

is simply $0011 \cup 0101 = 0111$ which is the binary representation of 7. Therefore,

$$U_i^3 \cup U_i^5 = U_i^7. \tag{4.26}$$

Similarly to U_i^a, X_i^a is defined as the modulo-2 sums of the same variables. The $2^m - 1$ possible such sums are denoted in the following way:

$$X_i^a = \sum \alpha_j S_i^j, \quad 0 \le j \le m-1; \tag{4.27}$$

where the summation is modulo-2 and where $\alpha_j = 0, 1$ depending on whether the particular variable S_i^j is part of the modulo-2 sum or not, and the space index a is formed according to eq. (4.23). Therefore,

$$\begin{aligned} X_i^1 &= S_i^0; \\ X_i^2 &= S_i^1; \\ X_i^3 &= S_i^0 \oplus S_i^1; \\ X_i^4 &= S_i^2; \\ X_i^5 &= S_i^0 \oplus S_i^2; \\ \vdots &= \vdots \\ X_i^{2^m-1} &= S_i^0 \oplus S_i^1 \oplus S_i^2 \oplus \cdots \oplus S_i^{m-1}. \end{aligned} \tag{4.28}$$

Clearly, there is one X_i^a which corresponds to the characteristic equation of the LFSR (see eqs. (4.2) and (4.6)). Let that particular X_i^a be labeled X_i^G. Therefore, eq. (4.6) can be rewritten as:

$$S_{i+1}^{m+1} = E_{i+1} \oplus X_i^G. \tag{4.29}$$

The particular values that are desired are the probabilities P_i^a defined as follows.

$$P_i^a = pr(U_i^a = 1); \quad 1 \le a \le 2^m - 1. \tag{4.30}$$

More specifically, it is desired to compute the P_{i+1}^a's from the P_i^a's. The following two theorems describe how this can be done.

Theorem 4.4: For $1 \le a < 2^{m-1}$,

$$P_{i+1}^a = P_i^{2a},$$

where $2a$ implies regular multiplication over the field of real numbers. For $2^{m-1} \leq a \leq 2^m - 1$,

$$P_{i+1}^a = \epsilon_{i+1}(1 + P_i^u) + (1 - 2\epsilon_{i+1})\ pr(X_i^x \cup U_i^u = 1),$$

where $u = 2a - 2^m$, and $x = G \wedge u'$, where $\wedge$ denotes the bitwise boolean AND operation, and $'$ denotes ones complementation.

Proof: The case where $1 \leq a < 2^{m-1}$ implies that S_{i+1}^{k-1} is not part of U_{i+1}^a. By the shifting property of the LFSR, for $0 \leq j < k-1$ (eq. (4.5),

$$S_{i+1}^j = S_i^{j+1}. \tag{4.31}$$

Therefore, an increment in time from i to $i+1$ implies a decrement in the space index of S from $j+1$ to j. In turn, from the definition of a, a decrement of one for each of the values of the space indices j implies the division of a by 2. Hence, for $1 \leq a \leq 2^{m-1}$,

$$P_{i+1}^a = P_i^{2a},$$

where $2a$ implies regular multiplication over the field of real numbers.

For $2^{m-1} \leq a \leq 2^m - 1$, since S_{i+1}^{k-1} is part of U_{i+1}^a, the latter can be rewritten as:

$$U_{i+1}^a = S_{i+1}^{k-1} \cup U_{i+1}^{a-2^{k-1}}. \tag{4.32}$$

Therefore,

$$P_{i+1}^a = pr(U_{i+1}^a = 1) = pr(S_{i+1}^{m-1} \cup U_{i+1}^{a-2^{m-1}} = 1). \tag{4.33}$$

From this theorem for $1 \leq a < 2^{m-1}$,

$$P_{i+1}^a = pr(S_{i+1}^{m-1} \cup U_i^{2a-2^m} = 1) \tag{4.34}$$

Using eq. (4.29),

$$P_{i+1}^a = pr\Big((E_{i+1} \oplus X_i^G) \cup U_i^{2a-2^m} = 1\Big). \tag{4.35}$$

Using Lemma 4.1,

$$P_{i+1}^a = pr\Big((E_{i+1} \oplus X_i^x) \cup U_i^u = 1\Big), \tag{4.36}$$

where $u = 2a - 2^m$, and $x = G \cdot u'$. Therefore,

$$\begin{aligned} P^a_{i-1} &= pr\Big((E_{i+1} \oplus X^x_i) \cup U^u_i = 1\Big) \\ &= 1 - pr\Big((E_{i+1} \oplus X^x_i) \cup U^u_i = 0\Big). \end{aligned} \tag{4.37}$$

The boolean expression $\Big((E_{i-1} \oplus X^x_i) \cup U^u_i = 0\Big)$ can be expanded into a product of maxterms:

$$(E_i \cup X^x_i \cup U^u_i)(\overline{E}_i \cup \overline{X}^x_i \cup U^u_i). \tag{4.38}$$

Since the two maxterms cannot both be satisfied simultaneously, they constitute mutually exclusive events. Therefore,

$$\begin{aligned} P^a_{i+1} = 1 - [&pr(E_{i+1} \cup X^x_i \cup U^u_i = 0) \\ &+ pr(\overline{E}_{i+1} \cup \overline{X}^x_i \cup U^u_i = 0)]. \end{aligned} \tag{4.39}$$

In turn, since X^x_i and U^u_i are independent of E_{i+1}.

$$\begin{aligned} P^a_{i-1} &= 1 - [(1 - \epsilon_{i+1})\ pr(X^x_i \cup U^u_i = 0) \\ &\quad + \epsilon_{i+1}\ pr(\overline{X}^x_i \cup U^u_i = 0)] \\ &= 1 - \Big[(1 - \epsilon_{i+1})\ \Big(1 - pr(X^x_i \cup U^u_i = 1)\Big) \\ &\quad + \epsilon_{i+1}\ \Big(1 - pr(\overline{X}^x_i \cup U^u_i = 1)\Big)\Big]. \end{aligned} \tag{4.40}$$

Using Lemma 4.2, the above easily simplifies to:

$$P^a_{i+1} = \epsilon_{i+1}(1 + P^u_i) + (1 - 2\epsilon_{i+1})\ pr(X^x_i \cup U^u_i = 1). \tag{4.41}$$

■

The equation for P^a_{i+1} for $2^{m-1} \le a \le 2^m - 1$ in Theorem 4.4 contains the term $pr(X^x_i \cup U^u_i = 1)$. The next theorem may be used to resolve this term into P^a_i's.

Theorem 4.5:

$$\begin{aligned} pr(X^x_i \cup U^u_i = 1) = &\Big((|x| + 1)\ mod\ 2\Big) P^u_i \\ &+ (-1)^{|x|-1} \sum_{s=1}^{|x|} (-2)^{s-1} \sum_t P^t_i, \end{aligned}$$

where $|x|$ denotes the number of ones in the binary representation of the index x, and where the indices t correspond to all the possible bitwise logical OR of the index u with the indices of s other components of X_i^x.

Proof: See Appendix B. ∎

Examples of equations described by Theorem 4.5 follow.

$|x| = 1$:

$$pr(S_i^q \cup U_i^u = 1) = pr(U_i^{u \vee q} = 1) = P_i^{u \vee q}. \tag{4.42}$$

$|x| = 2$:

$$pr((S_i^q \oplus S_i^r) \cup U_i^u = 1) = P_i^u - P_i^{u \vee q} - P_i^{u \vee r} + 2P_i^{u \vee q \vee r}. \tag{4.43}$$

$|x| = 3$:

$$\begin{aligned} pr((S_i^q \oplus S_i^r \oplus S_i^s) \cup U_i^u = 1) = {} & P_i^{u \vee q} + P_i^{u \vee r} + P_i^{u \vee s} \\ & - 2(P_i^{u \vee q \vee r} + P_i^{u \vee q \vee s} \\ & + P_i^{u \vee r \vee s}) + 4P_i^{u \vee q \vee r \vee s}. \end{aligned} \tag{4.44}$$

Summarizing what has been proven in Theorems 4.3, 4.4, and 4.5 is that it is possible to calculate the probability that any combination of up to m stages not be simultaneously in the zero state at time i from the corresponding set of probabilities at time $i-1$. Hence, the implementation of these theorems constitutes an algorithm that can be thought of as a dynamic programming solution approach to the problem [Aho 83]. At every time step, the $2^m - 1$ entries (probabilities) of a table are computed from the entries of the corresponding table at the previous time step. From this iterative procedure, it is possible to calculate the exact probability of aliasing for any length n and for any constant or varying value of ϵ in time $O(n\,2^m)$. A detailed analysis of the time and space complexities of this algorithm follows in the next subsection.

4.3.3 Detailed Space and Time Complexity Analysis of the Iterative Algorithm

In the following, the $2^m - 1$ probabilities that are calculated at each time step of the algorithm described above constitute the entries of a table denoted by T

Theorem 4.6: For an LFSR of size m, the space complexity of the iterative algorithm is $O(2^m)$.

Proof: For a register of size m, at every time step, the iterative algorithm described above computes $2^m - 1$ new probabilities from those from the previous time step. Hence, the space complexity of the algorithm is $O(2^m)$. ∎

In regard to the time complexity of the iterative algorithm, the following three theorems state upper bounds on the time complexity. The second theorem is a refinement of the first, and the third a refinement of the second. These are refinements in that they consider the number of feedback taps as a variable.

Theorem 4.7: For a register of size m and a sequence of length n, if the number of feedback taps, i.e., $|G|$, of the LFSR is considered a constant, then the time complexity of the iterative algorithm is $O(n\ 2^m)$.

Proof: At each time step, the iterative algorithm computes $2^m - 1$ new probabilities. If each of these $2^m - 1$ computations is assumed to require a constant amount of time, then for a sequence of length n, the algorithm requires $O(n\ (2^m - 1)) = O(n\ 2^m)$ time.

∎

From Theorem 4.5, it is clear that the time complexity is not the same for all the $2^m - 1$ probabilities computed at every time step. A worst-case analysis yields the upper bound stated in the following theorem.

Theorem 4.8: For an LFSR of size m with $g = |G|$ feedback taps, and a sequence of length n, the time complexity of the iterative algorithm is upper bounded by $O(ng\ 2^{m+g})$

Proof: From Theorems 4.4 and 4.5, the computation of the $2^m - 1$ new probabilities is dominated by the computation of the term $pr(X_i^x \cup U_i^u = 1)$. Notice that in actuality the computation of this term only arises for 2^{m-1} entries of **T**. From Theorem 4.4 the remaining entries of **T** do not require floating point calculations, but only shifts of indices. However, for the derivation of an upper bound, one can assume that the term $pr(X_i^x \cup U_i^u = 1)$ is computed for all $2^m - 1$ entries of **T**. Hence, $O(2^m)$ computations of the term $pr(X_i^x \cup U_i^u = 1)$ are required.

From Theorem 4.5, the number of terms in the expression for computing $pr(X_i^x \cup U_i^u = 1)$ is $2^{|x|}$ or $2^{|x|} - 1$. In turn, the indices for the individual terms of these expressions have to be computed as well. The latter computation is upper bounded by $O(|x|)$ time since up to $|x| + 1$ indices have to be ORed together to yield a new index. Hence, the evaluation of a term $pr(X_i^x \cup U_i^u = 1)$, requires $O(|x|\ 2^{|x|})$ time. From Theorem 4.4,

$$\begin{aligned} |x| &= G \setminus u' \\ &\leq |G| = g \end{aligned} \tag{4.15}$$

Therefore, from Theorem 4.5 and the above, for a polynomial with $g = |G|$ feedback taps, the computation of $pr(X_i^x \cup U_i^u = 1)$ is upper bounded by $O(f\ 2^g)$ computation steps. Since there are $O(2^m)$ such computations required, an upper bound on the time complexity of the algorithm is $O(ng\ 2^{m+g})$ for a sequence of length n. ∎

As mentioned in the proof of the preceding theorem, from Theorems 4.4 and 4.5, the computation of the $2^m - 1$ new probabilities of **T** is dominated by the computation of the term $pr(X_i^x \cup U_i^u = 1)$. In actuality, the computation of such terms only arises for 2^{m-1} entries. From Theorem 4.4, the remaining entries do not require floating point calculations, but only shifts of indices. For establishing the complexity of the computation involved, a refinement to the upper bound derived in the preceding theorem is not to assume that all the $2^m - 1$ entries involve a computation effort of $O(g\ 2^g)$. Such assumption yields the following upper bound on the time complexity.

Theorem 4.9: For an LFSR of size m with $g = |G|$ feedback taps, the time complexity of the iterative algorithm, for a sequence of length n, is upper bounded by $O(ng\ 2^m(3/2)^g) = O(ng2^{m+\ 585g})$.

Proof: See Appendix C. ∎

Consequently, the worst-case complexity for the iterative algorithm is $O(nq2^{1.585m})$, which arises when all the stages are fed back to the first stage, i.e., when $q = m$. In practice however, the configurations with few feedback taps are preferred since they result in less area overhead [Bardell 87]. Moreover, generally, the primitive feedback polynomials are the preferred ones, and for most degrees, there exist primitive feedback configurations such that $q = 2$ [Bardell 87].

4.4 Experimental Results

The iterative technique for computing exact probabilities of aliasing based on the equations given in Section 4.3 was implemented. In Fig. 4.1 and 4.2, the exact probabilities of aliasing for the non-primitive polynomial $f(x) = 1 + x^8 + x^{10}$ are shown for different values of error probability and different test sequence lengths. In Figs. 4.3 and 4.4, the probabilities of aliasing are plotted for the primitive polynomial $1 + x^7 + x^{10}$. For all the values of probability of error ϵ for which the probability of aliasing is plotted, the latter approaches its steady-state value 2^{-m} faster, i.e., at shorter sequence lengths, in the case of the primitive polynomial $1 + x^7 + x^{10}$ compared to the non-primitive polynomial $1 + x^8 + x^{10}$. Figs. 4.5 and 4.6 are plots of the logarithm (base 10) of the probability of aliasing for a register characterized by the primitive polynomial $f(x) = 1 + x + x^{15}$. In this case, the bounds from [Williams 87] and those from the previous chapter have been plotted as well. These plots show that for relatively short sequence lengths, the bound proposed in [Williams 87] is overly pessimistic, whereas the one from Chapter 3 of this dissertation is much more informative. Finally, in Fig. 4.7, the probability of aliasing is plotted for the polynomial $1 + x^7 + x^{10}$ for the case where the value of ϵ is 0.01 for the first 250 bits and is then 0.90 for the remainder of the test sequence. In practice, such change in the value of ϵ could occur when changing the distribution of test patterns, which is done in biased random testing [Lisanke 86] [Wunderlich 87] [Waicukauski 88].

The principal purpose for including these curves here is to demonstrate the

feasibility of the technique. To date, obtaining such probabilities for a register of length, say 8, appeared virtually impossible [Williams 87]. However, using the technique described here, generating the values for a sequence length of 1000 for a polynomial of degree 10 takes less than two minutes of CPU time on a MicroVax

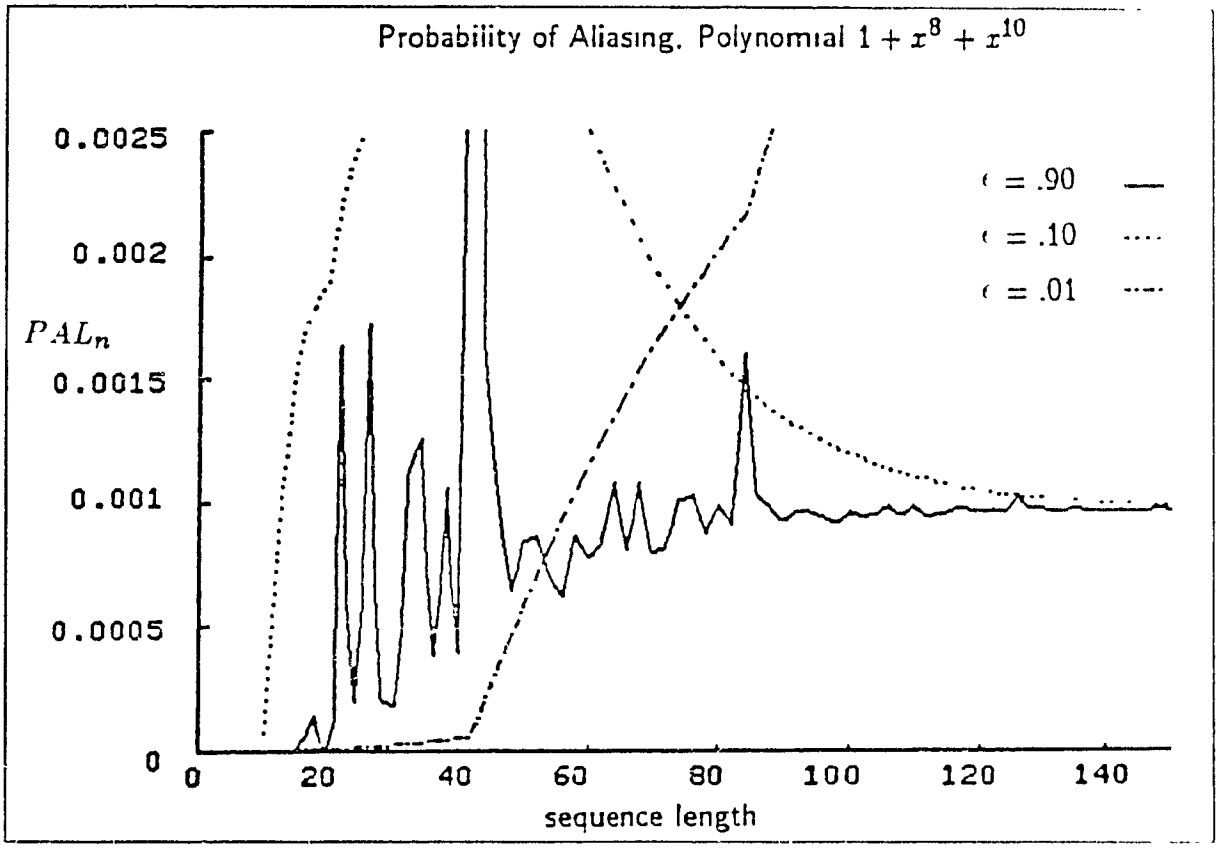

Figure 4.1 Probability of aliasing, $f(\tau) = 1 + x^8 + x^{10}$

4.5 Summary

This chapter described a completely new approach for calculating the probability of aliasing for LFSRs characterized by any feedback polynomials. A double recursion equation for the probability of the "all-zero" state of an LFSR is given. The recursion is double in that the probability of the "all-zero" state at time i is expressed in terms of the corresponding probability at time $i-1$ and in terms of the state probability of a disjunction of fewer stages at time i. This recursion equation forms the basis of an

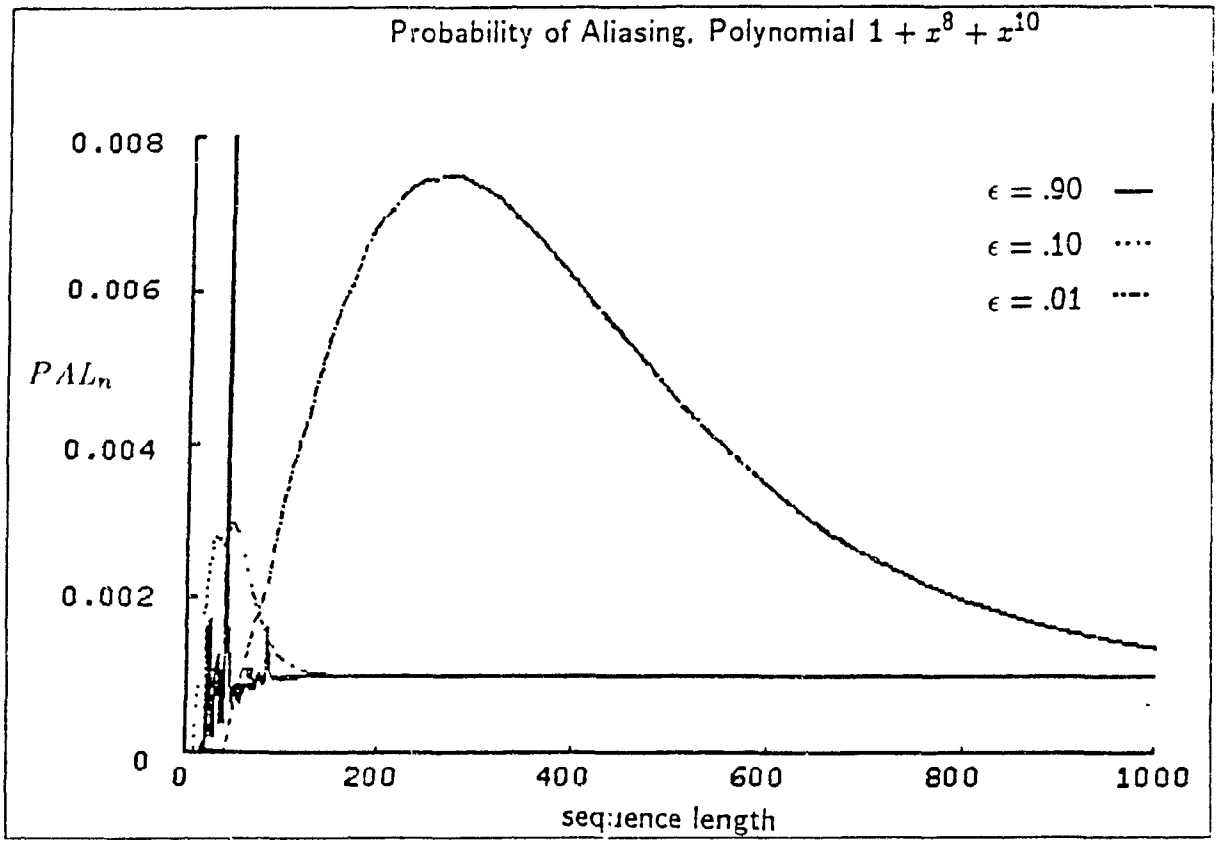

Figure 4.2 Probability of aliasing, $f(x) = 1 + x^8 + x^{10}$

iterative technique for computing the probability of aliasing for LFSRs characterized by any feedback polynomial and for any test sequence length. The complete algorithm involves $2^m - 1$ equations, however, two theorems are proven that show how to express these $2^m - 1$ equations through only two general equations. Results using the technique are presented and the complexity of the algorithm is analyzed in detail.

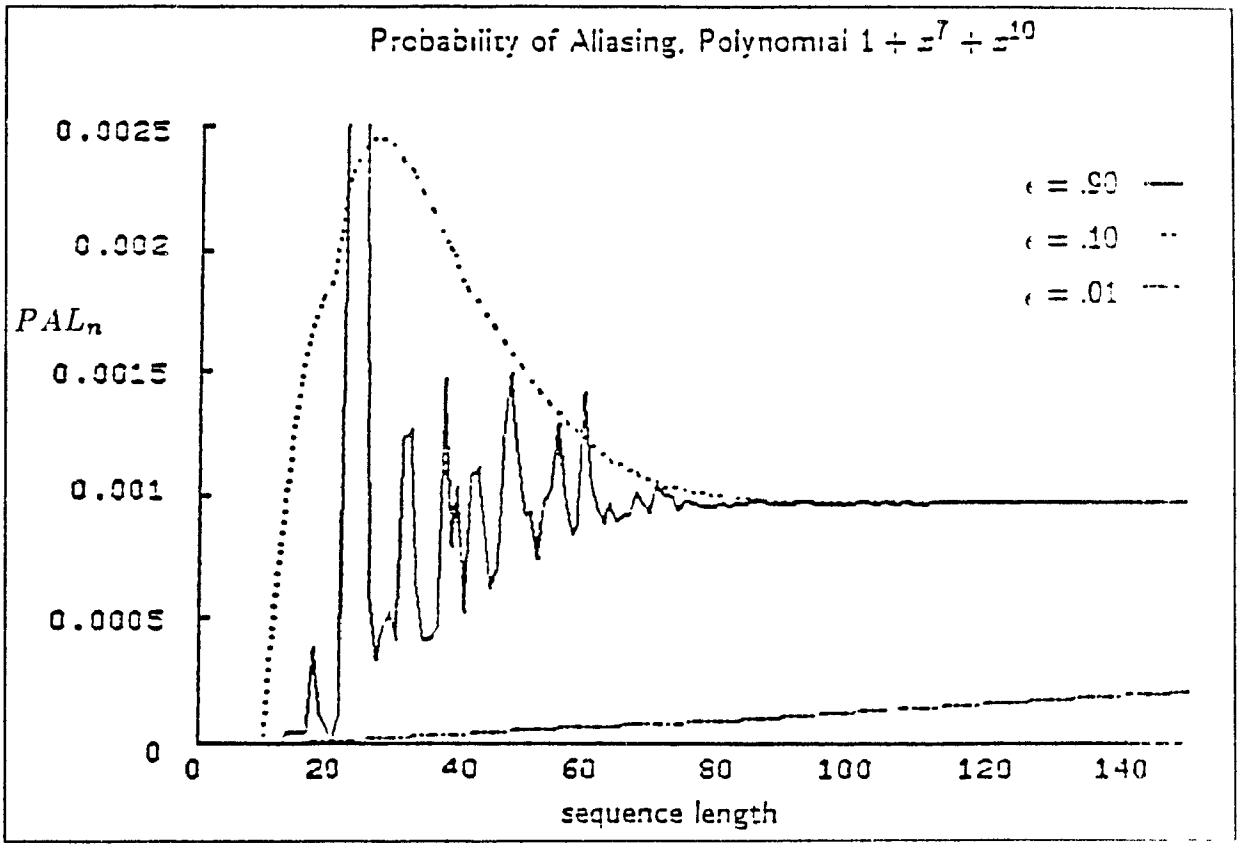

Figure 4.3 Probability of aliasing, $f(x) = 1 + x^7 + x^{10}$

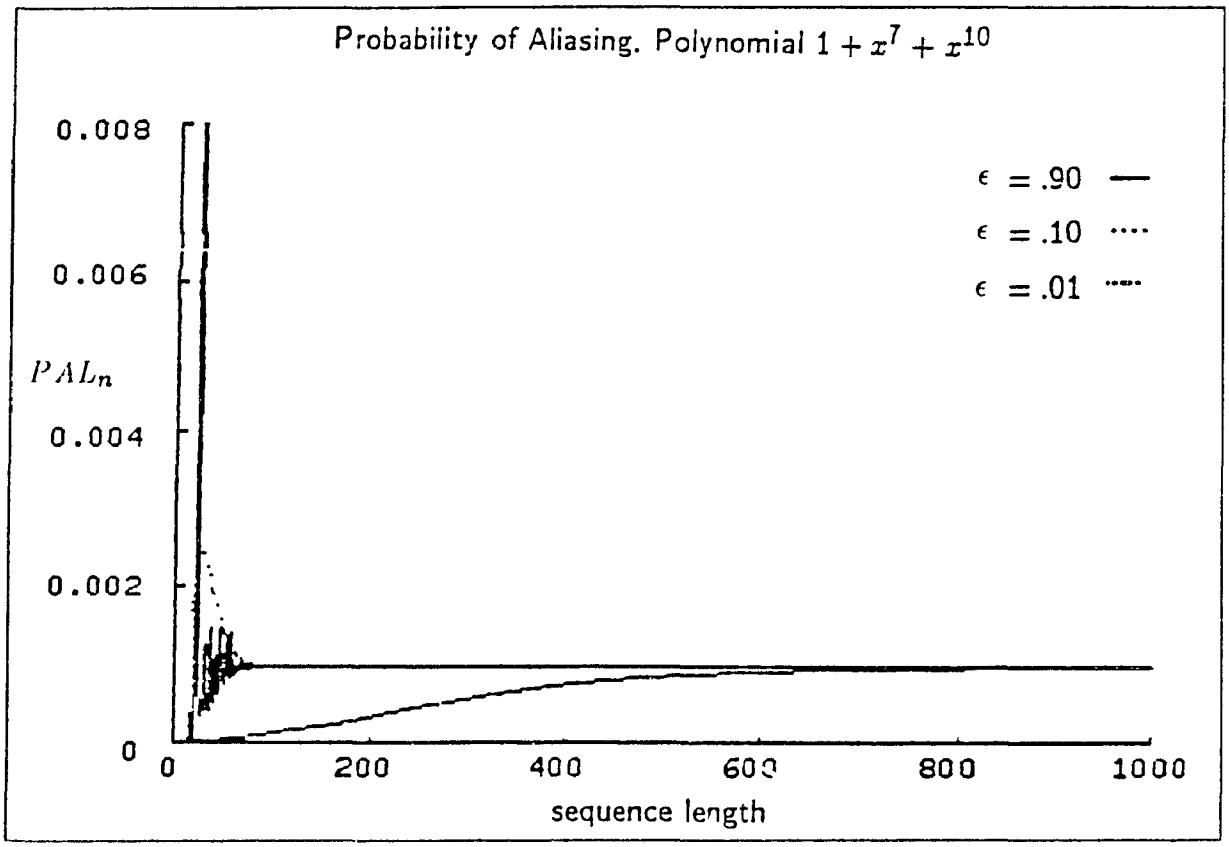

Figure 4.4 Probability of aliasing, $f(x) = 1 + x^7 + x^{10}$.

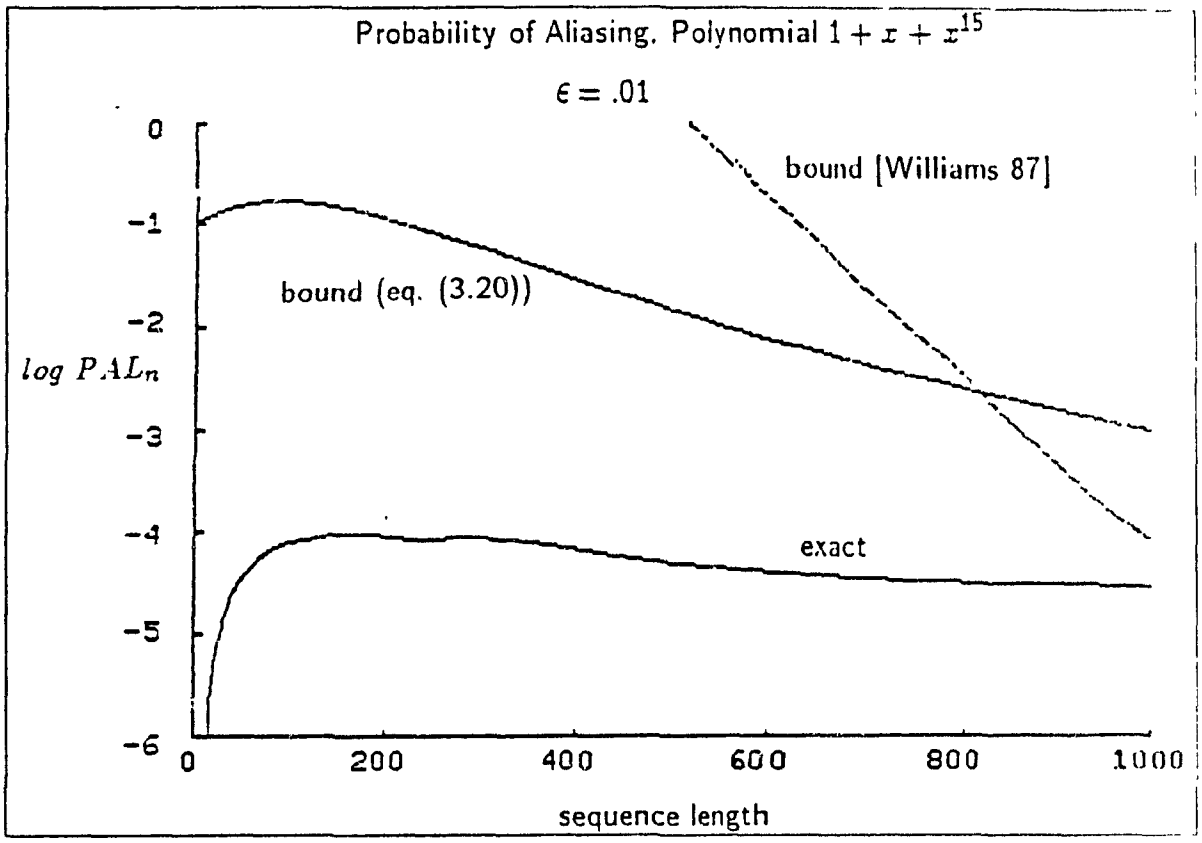

Figure 4.5 Probability of aliasing, $f(x) = 1 + x + x^{15}$, $\epsilon = .01$

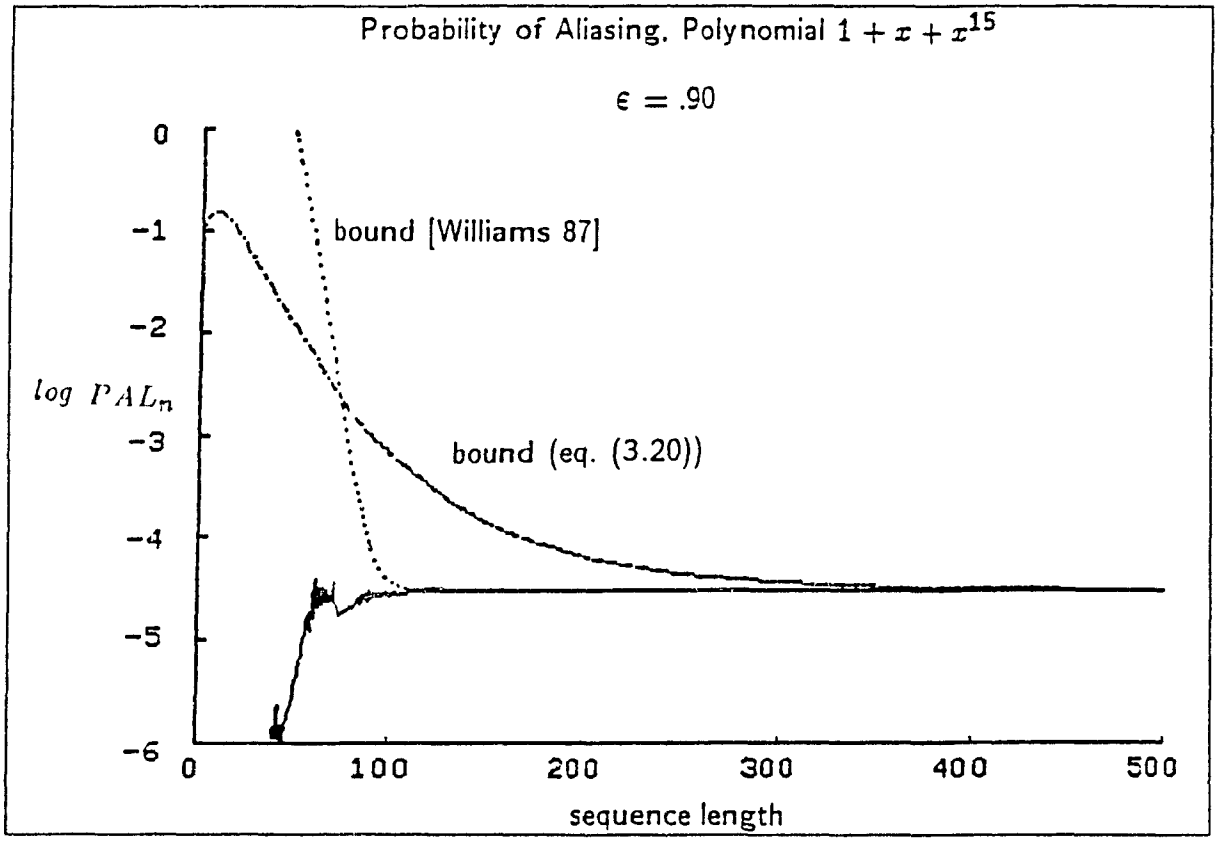

Figure 4.6 Probability of aliasing, $f(x) = 1 + x + x^{15}$, $\epsilon = .90$

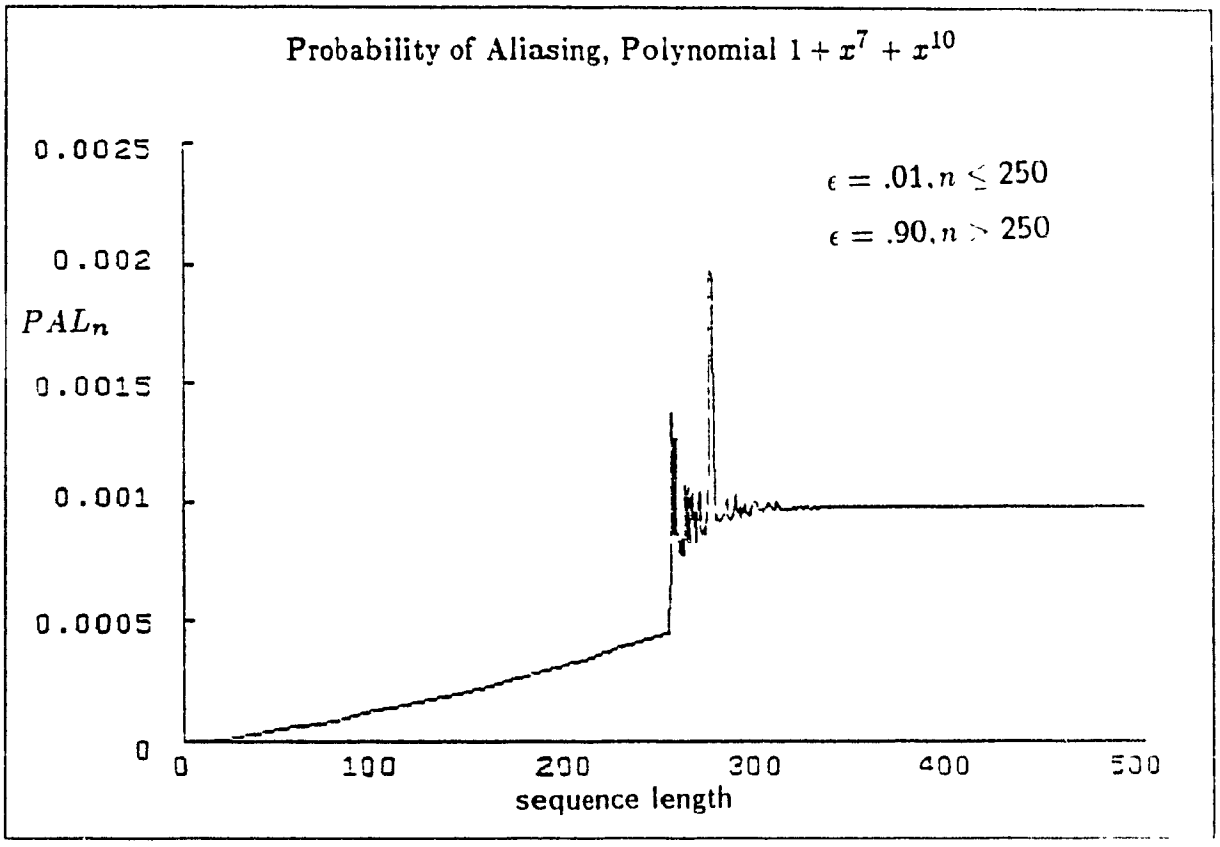

Figure 4.7 Probability of aliasing, $f(x) = 1 + x^7 + x^{10}$, $n \leq 250 \cdot \epsilon = .01$, $n > 250 \cdot \epsilon = .90$

Chapter 5 Applications and Extensions

5.1 Detection Confidence and Test Length Calculations

Several recent research efforts have been aimed at the analysis of random or pseudorandom testing in view of establishing the required test length to obtain a certain test quality, e.g., a given fault coverage with a certain degree of confidence. An excellent reference that summarizes the results to date is [Wagner 87]. However, none of the reported analyses are truly aimed at built-in self-testable circuits since they do not assume a compaction stage at the output of the CUT. Probably the principal reason for this lack of analyses that include the effects of compaction is the lack of efficient analysis tools that enable the effects of compaction to be studied under reasonable error models, i.e., models other than the uniform distribution of error sequences. Using the techniques for calculating the probability of aliasing for signature analyzers developed in Chapters 3 and 4 of this dissertation, the previous work on test confidence and test length calculations for random testing can be extended. i.e., new expressions for test confidence and test length that take into account compaction can be derived. This is done in subsequent subsections

The assumptions under which the results are obtained are the following. The CUT is assumed to be a single-output circuit. In regard to the types of test patterns that are applied to the CUT, these are assumed to be random, i.e., to come from a source where the sampling is done with replacement, as opposed to pseudorandom patterns, i.e., coming from a source where the sampling is done without replacement. The

assumption of true random patterns enables the fault detectability ϵ value to be considered constant for the entire test sequence. Normally, pseudorandom testing [Wagner 87] is the applicable model when the test patterns applied to a circuit are generated from a maximal-length LFSR. However, in the usual case where the number of test patterns applied to the CUT is relatively small compared to the period of the LFSR, then the assumption of random testing is a good approximation [McCluskey 88]. In regard to the single-output assumption, a generalization to multi-output circuits is discussed briefly in a later subsection.

5.1.1 Detection confidence before compaction

There exist different probabilistic measures of random or pseudorandom test quality (see [Wagner 87]). Some are based on *sets* of faults while others are based on only *single* (often "worst-case") faults. To avoid any loss of generality, here the focus of the analysis is on a single fault. The results for single faults can be extended to measures that consider sets of faults.

Since the assumption is that random patterns are applied to the CUT, in the presence of a particular fault the probability of an error at the output of the CUT when applying one pattern is equal to the probability of detection of that fault. Let this probability of error be ϵ. Given a fault's probability of detection, and assuming no compaction at the output, the required test length n for detecting the fault with a certain confidence is easily calculated. Let the detection confidence C be the probability of detecting the fault when applying n test patterns. Let $C = 1 - e$, then e is the probability of not detecting the fault when applying the n test patterns and is called the *escape probability* [Savir 84]. Therefore, assuming n to be the number of test patterns applied to the CUT and ϵ to be the fault's single pattern detection probability, then $e = (1 - \epsilon)^n$. Solving for n yields the following lower bound to obtain the desired detection confidence:

$$n \geq \lceil \frac{ln\ e}{ln(1-\epsilon)} \rceil. \tag{5.1}$$

Since the next assumption is that compaction of the CUT's output response

is done, let the above detection confidence C, the escape probability e, and the calculated test length n, be subscripted by "*bc*" to denote "before compaction". Therefore, e_{bc} is the specified escape probability before compaction and n_{bc} is the calculated required test length for achieving the specified escape probability before compaction.

5.1.2 Detection confidence and test length after compaction

In the previous subsection, an expression for the required test length to achieve a certain detection confidence assuming no compaction, is given. Now assume that the output sequence of the CUT is fed to a signature analysis register that compacts the output. It is desired to estimate the effect of compaction on the detection confidence and the test length.

Let C_{ac} and e_{ac} denote the detection confidence and the escape probability *after* compaction, respectively. Given that the circuit is faulty, C_{ac} is the probability that a faulty signature results, and e_{ac} is the probability that the resultant signature is that of the fault-free circuit. Because of the possibility of aliasing, for a fixed test sequence length, $C_{ac} < C_{bc}$. Alternatively, $e_{ac} > e_{bc}$. If the CUT's output sequence is error-free, then, assuming a correctly functioning compaction network, the resulting signature will be that of the fault-free circuit. Hence, for a fault to be detected *after* compaction, i.e., for a faulty signature to result, the fault must first be detected *before* compaction.

In addition to being detected before compaction, for a fault to be detected after compaction, no aliasing can occur. From the definition of the probability of aliasing given in Chapters 3 and 4, the case where the LFSR starts in the "all-zero" state and returns to this state after a test sequence of length n has been applied to the CUT is precisely the situation where a fault would not be detected after compaction. Hence, eq (3.18) can be rewritten as:

$$PAL = e_{ac} - e_{bc}, \tag{5.2}$$

or,

$$e_{ac} = e_{bc} + PAL. \tag{5.3}$$

Alternatively, using the relationship $C = 1 - e$,

$$C_{ac} = C_{bc} - PAL. \tag{5.4}$$

Subsequently, two different values for e_{ac} are distinguished depending on whether the aliasing probability is assumed to be the 2^{-m} asymptotic value or a more precise non-asymptotic value. Since the asymptotic 2^{-m} aliasing probability is the aliasing probability in the limit as the sequence length tends to infinity, let e_{ac}^{∞} correspond to the case where the asymptotic value is assumed, and e_{ac} correspond to the more precise case.

Assuming the 2^{-m} asymptotic probability of aliasing yields the following expression for the escape probability after compaction:

$$e_{ac}^{\infty} = e_{bc} + 2^{-m}. \tag{5.5}$$

Instead of assuming the asymptotic value for the probability of aliasing, if the signature register is assumed to have a characteristic polynomial $1 + x^m$, then eq (3.19) can be used for PAL in eq. (5.3) to obtain an exact expression for e_{ac}. From eq. (3.19), and eq. (5.3),

$$e_{ac} = 2^{-m}\lfloor 1 + (1-2\epsilon)^{\lfloor n/m \rfloor}\rfloor^{m-n \bmod m}\ \lfloor 1 + (1-2\epsilon)^{\lceil n/m \rceil}\rfloor^{n \bmod m}. \tag{5.6}$$

Moreover, assuming that $n/m \gg 1$, eq. (5.6) simplifies to:

$$e_{ac} = 2^{-m}\lfloor 1 + (1-2\epsilon)^{n/m}\rfloor^{m}. \tag{5.7}$$

Some numerical results obtained from the above derivations are reported in Table 5.1. Assuming two different values of ϵ (0.01 and 0.001), using eq (5.1), values for n_{bc} were calculated for different chosen values of e_{bc}. From these, in turn, using eqs (5.5) and (5.6), values for e_{ac}^{∞} and e_{ac} were obtained for different values of m. What is most significant in the presented results is that in some cases there is an order of magnitude difference between e_{ac}^{∞} and e_{ac}, e.g., when $m = 16$ and $\epsilon = .001$.

Escape Probability and Test Length Polynomials: $1 + x^m$					
m	e_{bc}	n_{bc}	e_{ac}^{∞}	e_{ac}	n_{ac}
$\epsilon = .01$					
16	10^{-1}	230	1.00×10^{-1}	1.17×10^{-1}	249
	10^{-2}	459	1.00×10^{-2}	1.90×10^{-2}	551
	10^{-4}	917	1.15×10^{-4}	12.15×10^{-4}	1648
	10^{-6}	1375	16.32×10^{-6}	205.49×10^{-6}	--
32	10^{-1}	230	1.00×10^{-1}	1.08×10^{-1}	238
	10^{-2}	459	1.00×10^{-2}	1.37×10^{-2}	496
	10^{-4}	917	1.00×10^{-4}	3.58×10^{-4}	1100
	10^{-6}	1375	1.00×10^{-6}	17.36×10^{-6}	1915
$\epsilon = .001$					
16	10^{-1}	2302	1.00×10^{-1}	1.18×10^{-1}	2495
	10^{-2}	4603	1.00×10^{-2}	1.92×10^{-2}	5541
	10^{-4}	9206	1.15×10^{-4}	12.36×10^{-4}	16640
	10^{-6}	13809	16.26×10^{-6}	208.95×10^{-6}	-
32	10^{-1}	2302	1.00×10^{-1}	1.09×10^{-1}	2391
	10^{-2}	4603	1.00×10^{-2}	1.39×10^{-2}	4989
	10^{-4}	9206	1.00×10^{-4}	3.69×10^{-4}	11087
	10^{-6}	13809	1.00×10^{-6}	17.99×10^{-6}	19311

Table 5.1 Escape Probability and Test Length: Polynomials $1 + x^m$

The results indicate that for LFSRs with characteristic polynomial $1 + x^m$, compaction may in some cases significantly decrease the confidence of detecting a fault. Hence, one natural question to ask is what happens to the escape probability e_{ac} after compaction if the test length is increased. Solving for n in eq. (5.7) yields

$$n \approx \frac{m \ ln(2e_{ac}^{1/m} - 1)}{ln(1 - 2\epsilon)}. \tag{5.8}$$

Therefore, similarly to eq. (5.1) which relates test length and escape probability assuming no compaction, eq. (5.8) relates the same quantities assuming compaction by an LFSR with characteristic polynomial $1 + x^m$. In Table 5.1, the column n_{ac} is the value of test length obtained using eq. (5.8) taking $e_{ac} = e_{bc}$. The entries marked by a " "

indicate that the confidence is not achievable, i.e., even if n is increased without bound. $e_{ac} < e_{hc}$.

In the case of LFSRs characterized by polynomials other than $1 + x^m$, no closed-form expressions can readily be obtained for the probability of aliasing. Hence, no closed-form expressions can readily be obtained for e_{ac}. However, for primitive polynomials, closed-form expressions for bounds on e_{ac} can be derived using the expressions for the upper bounds on aliasing obtained for primitive polynomials (Section 3.5). In general however, i.e., for any type of feedback polynomials, the iterative technique described in the previous chapter can be used to calculate e_{ac} directly, since the basis for the technique is precisely the calculation of pr("all-zero" state $\rightarrow$ "all-zero" state) which is also precisely e_{ac}.

Some values for e_{ac} and n_{ac} were calculated assuming that the LFSR was characterized by the primitive polynomial $f(x) = 1 + x^7 + x^9 + x^{12} + x^{16}$ adopted by Hewlett Packard for its signature analysis schemes [Frohwerk 77]. Results are presented in Table 5.2. Clearly, in regard to test confidence and test length, the use of such primitive polynomial yields much more desirable effects compared to that of polynomials $1 + x^{16}$. The values calculated for e_{ac} using exact techniques are very near those obtained assuming the asymptotic probability of aliasing

5.1.3 Summary

Much work has already been aimed at deriving expressions that can be used to estimate the number of test patterns that should be applied to a CUT to achieve a certain degree of confidence. Most of these techniques are based on the detection probabilities of the different faults of a particular CUT. However, until now, none of these expressions had been derived taking into account the effect of compaction. From, the techniques developed in the preceding chapters for calculating the probability of aliasing, test length and test confidence calculations that specifically take into account compaction by a signature analysis register can now be performed. This was briefly illustrated in the above subsection. In turn, based on the analysis for single faults, using

Escape Probability and Test Length Polynomial: $1 + x^7 + x^9 + x^{12} + x^{16}$					
m	e_{bc}	n_{bc}	e_{ac}^{∞}	e_{ac}	n_{ac}
$\epsilon = .01$					
16	10^{-1}	230	1.00×10^{-1}	1.00×10^{-1}	230
	10^{-2}	459	1.00×10^{-2}	1.00×10^{-2}	459
	10^{-4}	917	1.15×10^{-4}	1.15×10^{-4}	933
	10^{-6}	1375	16.32×10^{-6}	17.36×10^{-6}	–
$\epsilon = .001$					
16	10^{-1}	2302	1.00×10^{-1}	1.00×10^{-1}	2302
	10^{-2}	4603	1.00×10^{-2}	1.00×10^{-2}	4605
	10^{-4}	9206	1.15×10^{-4}	1.16×10^{-4}	9371
	10^{-6}	13809	16.26×10^{-6}	16.26×10^{-6}	–

Table 5.2 Escape Probability and Test Length HP Polynomial $1 + x^7 + x^9 + x^{12} + x^{16}$

the measures discussed in Chapter 2, different probabilistic fault coverage measures can be calculated. Nonetheless, mostly due to the difference between fault domain and error domain deception volumes, the quality of the derived probabilistic fault coverage measures remains an open issue. Finally, an immediate apparent drawback of the above analysis is its limitation to single-output circuits. The following subsection discusses an extension to multi-output circuits.

5.2 Extensions to Multi-Output Circuits

Most applications of BIST would normally be intended for multi-output circuits. Several different schemes have been proposed for the compaction of the response of multi-output circuits. Readers interested in the various propositions made in the past are referred to [Zorian 87] or [Bardell 87] for discussions and numerous references.

In the case of signature analysis, one straightforward option is to have a register compacting each one of a circuit's outputs. In [David 84], such a scheme as well as some variations of it are studied for the case where the signature analysis registers

are characterized by polynomials $1 + x^m$. Of course, the overhead of such schemes is high. Another alternative is to multiplex a single-input signature analyzer to the CUT's various outputs, one at a time, and to repeat the test sequence for each output.

An interesting alternative to multiplexing is possible in the case of boundary scan designs [Maunder 87] [Lagemaat 87]. In such designs, in test mode, each of a circuit's inputs and outputs may be connected to form a shift register. For a chip with boundary scan, a BIST scheme such as the one shown in Fig 5.1 is a possibility. In that scheme, a test consists of loading the scan register with a pseudo-random test pattern, applying the pattern to the CUT, collecting the CUT's output response in the scan register, and finally shifting out the response in a signature analysis register. Hence, for such a scheme, the analysis techniques developed in Chapters 3 and 4 for single-input signature registers can be used for multi-output circuits. The iterative technique of Chapter 4 is particularly suited since it can handle variable values of ϵ. Such would be the case here since the sequence fed to the signature analyzer would originate from several different outputs of a CUT. The techniques developed to estimate fault detectabilities [Brglez 84] [Jain 85] [Seth 85], i.e., ϵ, can generally be used with multi-output circuits as well. However, for the multi-output circuits, these algorithms calculate multiple single-output probabilities of fault detection, but do not calculate truly multiple-output (joint) fault detection probabilities. Hence, given this drawback, the simplest assumption to make is to assume that the errors appearing on any given output are independent from those appearing on any other output, hence are uncorrelated. Thus, assuming of independence or errors at a CUT's outputs, and assuming that a particular fault produces an error at output 1 with probability ϵ_1, at output 2 with probability ϵ_2, ..., at output m with probability ϵ_m, the sequence of error probabilities corresponding to the error sequence shifted in the signature analyzer would periodic of the form:

$$\epsilon_1, \epsilon_2, \ldots, \epsilon_m, \epsilon_1, \epsilon_2, \ldots, \epsilon_m, \epsilon_1, \epsilon_2, \ldots, \epsilon_m \ldots . \tag{5.9}$$

Such a type of sequence of values of ϵ is readily handled by the iterative technique of Chapter 4. If the independence assumption were not considered reasonable, other types of multi-output error models could also be studied with the iterative technique since it

can handle any value of ϵ on any bit of a sequence.

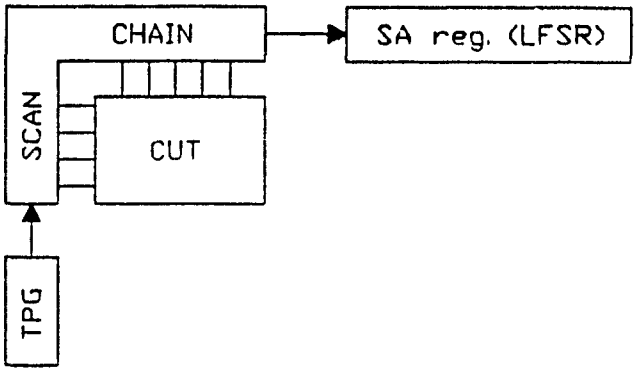

Figure 5.1 Multi-output circuit with a scan chain and an LFSR for SA

Another BIST scheme for multi-output circuits that uses signature analysis for compaction is one where the outputs of the circuit are fed directly to a parallel input signature register or *multi-input shift register* (MISR), as shown in Fig. 5.2. For such cases, the analysis techniques developed in the preceding chapters for single-input signature analyzers can also be used since it has been shown by several researchers [Sridhar 82] [Hassan 83] [Bardell 87] that the parallel compaction of the response of a multi-output CUT by a MISR can be reduced to the compaction of an equivalent serial sequence fed to a single-input LFSR characterized by the same feedback configuration as that of the MISR. In each of the cited references, the reduction is shown for the internal type of MISR (and hence LFSR), i.e., the type where the last stage of the LFSR feeds back to several EXOR gates that lie between the stages. The internal type of MISR is the most attractive in regard to area overhead. However, the analysis techniques developed in Chapters 3 and 4 apply to the external types of feedback configurations. However, as mentioned in Chapter 3, isomorphic transformations between the two types have been demonstrated [Illawiczka 86]. Hence, using the two types of transformations, the process of parallel compaction by an internal type of MISR can be reduced to the serial compaction by an external type of LFSR.

The aforementioned formal reductions are not presented here. However, an example is given to illustrate the nature of the transformation that reduces the

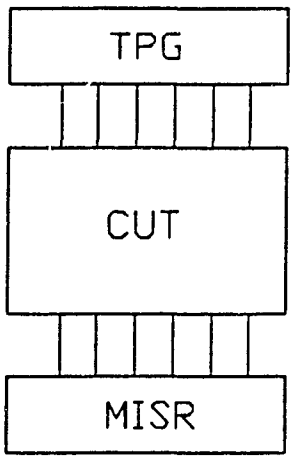

Figure 5.2 Multi-output circuit with a MISR for SA.

parallel compaction process by an internal MISR to the compaction of an equivalent serial sequence by a single input LFSR with the same feedback configuration as that of the MISR. Fig 5.3a illustrates the compaction of six 5-bit vectors by a 5-bit MISR with a feedback polynomial $f(x)$, while Fig. 5.3b illustrates the functionally equivalent compaction of a serial bit sequence, derived from the six vectors, by a single-input LFSR characterized by the same feedback polynomial $f(x)$. The equivalent serial sequence is simply a bit-wise modulo-2 sum of shifted versions of the output vectors from the CUT. Hence, assuming that each output bit from a given output i of the CUT has a constant probability of error ϵ_i, and is independent of every other output bit, then the probability of error of each of the bits in the equivalent sequence can be calculated using eq (3.3) recursively Given the equivalent probability of error, ϵ_{eq} for the equivalent serial sequence, the techniques developed in the preceding chapters can be used to analyze the effect of compaction with multi-output circuits.

Hence, the generalization of the analysis of compaction of the response of multi-output circuits by LFSRs or MISRs can be considered rather straightforward since the analysis techniques developed for analyzing single-input signature registers can be applied to the multi-output cases. Based on the assumption of the independence of errors at the outputs of multi-output CUTs, techniques to find equivalent ϵ values have been suggested above. The problem that remains open is determining whether such assumptions are reasonable.

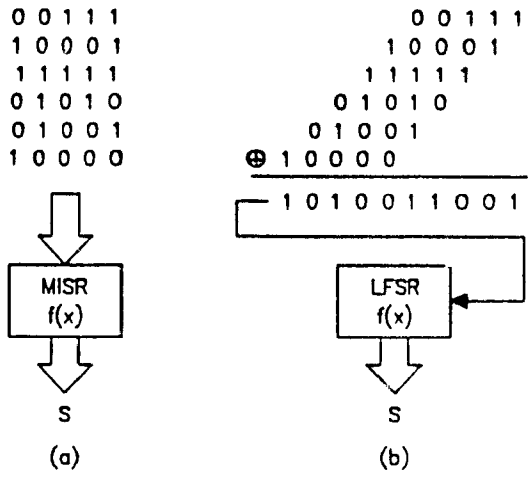

Figure 5.3 (a) A MISR, and (b) its equivalent single-input LFSR [Sridhar 82]

5.3 Coding Theory Applications

5.3.1 Introduction

The aliasing problem in the context of BIST is closely related to the problem of an undetected error in coding theory. Though testing experts have often exploited the results from coding theory for making claims in regard to their testing problems, few have explicitly stated this relationship, and consequently, equivalent results have occasionally been derived independently. For example, the result by Williams et al [Williams 86] stating that the probability of aliasing tends to 2^{-m} as the sequence length tends to infinity, for all LFSR configurations, and for all values of the error probability, had previously been derived in the context of coding theory, i.e., in [Witzke 85]. Recently, in [Gupta 88], Gupta and Pradhan explicitly stated the relationship between the testing and coding theory problems. Gupta and Pradhan used a well-known expression for the probability of an undetected error for cyclic Hamming codes, and derived from it an expression for the probability of aliasing in BIST. However, their result applies only to the aliasing probability of test sequences whose lengths correspond to natural lengths of Hamming codes. In the following, after describing the relationship between the probability of aliasing in testing and the probability of an undetected error in coding theory, the iterative technique developed for computing the probability of

aliasing is shown to be useful for computing the probability of undetected errors for any cyclic code and any shortened version of the latter.

5.3.2 Probability of an Undetected Error

Cyclic codes, as well as *shortened* cyclic codes, also known as *polynomial* codes or *cyclic redundancy check* (CRC) codes [Lin 83], are widely used for error detection in data communication systems. The particular appeal of shortened cyclic codes is that their encoding and decoding may be performed using the same simple circuits as the ones employed for the original code. A number of cyclic codes have been adopted as standards, e.g., the CRC-12, CRC-ANSI, and CRC-CCITT codes [Tanenbaum 81].

An (n, k) binary cyclic code C is a set of 2^k code words of n bits defined as a subspace of a vector space over $GF(2)$. For all code words $c \epsilon C$, the corresponding code polynomial $c(x)$ is divisible by a polynomial $g(x)$ of degree $m = n - k$, called the generator polynomial of C. Supposing that a code word is transmitted, let $r(x)$ denote the code polynomial of the received word. Because of the channel noise, the received code polynomial may be different from the transmitted code polynomial. In the decoding of a linear code, the first step is to compute the syndrome. The syndrome computation can be performed by dividing $r(x)$ by the generator polynomial $g(x)$. Hence, an LFSR whose divisor polynomial corresponds to the code generator polynomial $g(x)$ can be used to compute the syndrome. If the syndrome is zero, $r(x)$ is a code polynomial and the decoder accepts $r(x)$ as the transmitted code polynomial. If the syndrome is not zero, then $r(x)$ is not a code polynomial, and the presence of error is detected.

However, in the case where the syndrome is zero, there is a possibility that the presence of errors is undetected. This occurs when the error polynomial $e(x)$ corresponds to a code word of C. That is, if the error polynomial is also divisible by the code generator polynomial $g(x)$, then the computed syndrome will be zero despite the presence of errors in the received word. Usually, the possibility for such undetected errors is analyzed for the binary symmetric channel (BSC), and is characterized by a probability, denoted here by P_u.

The probability of an undetected error can be computed if the weight distribution of the code is known. Theoretically, the weight distribution of an (n,k) linear code can be computed by examining its 2^k code words or by examining the 2^{n-k} code words of its dual and then applying the MacWilliams identity [Lin 83]. However, as stated in [Lin 83], except for some short linear codes and a few classes of linear codes the weight distributions for many known linear codes are still unknown because finding such distributions is computationally infeasible. Consequently, it is generally very difficult, if not impossible, to compute P_u for many known codes.

Nevertheless, the undetected error probability for various (n,k) linear codes used soleiy for error detection on a BSC channel with bit error rate (or transition probability) ϵ, $\epsilon \leq 1/2$, has been discussed in numerous recent papers [Leung-Yan-Cheong 76] [Leung-Yan-Cheong 79] [Wolf 82] [Kasami 83] [Witzke 85] [Fujiwara 85] [Miller 85] [Fujiwara 86]. Earlier, Korznik [Korznik 65] proved that there exists (n,k) linear codes whose probability P_u of an undetected error satisfies the following upper bound:

$$P_u \leq 2^{-(n-k)}[1-(1-\epsilon)^k], \tag{5.10}$$

for all n, k, and ϵ such that $0 \leq \epsilon \leq 1/2$. Korznik's proof is an existence proof. Since then no general method has been found for generating codes that satisfy eq. (5.10). However, a few classes of known codes have been shown to satisfy a weaker bound on P_u, namely:

$$P_u \leq 2^{-(n-k)}. \tag{5.11}$$

Examples of such codes are Hamming single-error correcting codes, perfect binary codes, and double-error-correcting primitive BCH codes of natural length [Leung-Yan-Cheong 76] [Leung-Yan-Cheong 79].

Hence, Hamming codes of natural length do satisfy the upper bound $P_u \leq 2^{-(n-k)}$. Moreover, for small values of ϵ, P_u is much smaller than $2^{-(n-k)}$. However, shortened Hamming codes do not necessarily obey this bound. As mentioned in [Fujiwara 85], the reason for the interest in the performance, measured through P_u, of shortened codes (not necessarily Hamming codes) is that the natural length of

some of the standard codes may be quite long, e.g., the length of the CCITT code is $n = 2^{15} - 1 = 32\ 767$. However, in practice, the length of a data packet is often no more than a few thousand bits, which may be much shorter than the natural length of the adopted code. Consequently, shortened versions of a code must often be used [Fujiwara 85]. Since the length of a data packet also often varies from say a few hundred to a few thousands bits, codes must also be shortened by various degrees. Already in [Leung-Yan-Cheong 76], shortening has been proven to affect the performance of a cyclic code with respect to the probability of an undetected error. This performance is precisely the focus of numerous recent papers, e.g., [Fujiwara 85] [Witzke 85], and presentation abstracts, e.g., [Miller 85], [Fujiwara 86].

In [Fujiwara 85], P_u as a function of ϵ for shortened versions of the CCITT codes obtained from two different polynomials is discussed, while in [Witzke 85], the shortened versions of the CRC-12, CRC-ANSI and the CCITT codes are examined. In both cases, the results are obtained by computing the weight distribution of the dual code and then making use of the MacWilliams identity. In [Witzke 85], so-called "direct" methods were used to compute the weight distributions of interest. Only eight different shortened versions (k = 5, 10, 20, 50, 100, 200, 500, 1000) of the codes were studied for different values of ϵ.

In [Fujiwara 85], two iterative methods that are generally more efficient than the direct method are proposed for computing the weight distribution of shortened Hamming codes. One of the major strengths of these iterative methods is that their iterative nature does not require the lengths of the shortened codes of interest to be fixed beforehand, i.e., the methods enable the weight distributions for the q codes of lengths n for all $1 \leq n \leq q$ to be obtained. However, once the weight distributions for $1 \leq n \leq q$ have been obtained, the MacWilliams identity then has to be used to compute the desired probability P_u. Using their iterative methods, the authors reported curves for P_u as a function of ϵ for a dozen different lengths from $n = 24$ to $n = 32\ 767$. They also reported the peak values of P_u for different values of n and different values of ϵ. Using their iterative technique, the order of the computation time to compute P_u for q different code lengths, for a generator polynomial of degree m, is $O(q\ 2^m)$ (considering

the number, p, of non-zero coefficients of the generator polynomial to be a constant, otherwise, the complexity is linear in p). However, the iterative techniques described in [Fujiwara 85] rely on properties of maximal-length sequences, i.e., on specific properties of sequences generated by primitive polynomials. Hence, the techniques do not apply to any code generator polynomials.

In the context of coding theory, for an undetected error to occur, the case where no errors occur must be excluded †, and the polynomial divider (LFSR) must have started in the "all-zero" state and returned to this state after the n bits of the code word have been shifted in the LFSR. This condition for an undetected error corresponds exactly to the definition of aliasing (eq. (4.7)), hence, $P_u = PAL$. Therefore, the novel iterative technique described in the preceding chapter for computing the aliasing probability PAL can also be used for computing P_u for cyclic codes of any lengths and generated by any polynomial. If the iterative technique described in Chapter 4 is used to compute P_u, the time complexity of the algorithm will be $O(q\,2^m)$ for computing P_u for q different code lengths from 1 to q (see Section 4.3.3 for more details on the algorithm's complexity). Using the iterative technique of Chapter 4 to compute P_u for various codes (polynomials) and various code lengths constitutes a fundamentally different approach from all the known previous attempts in that this iterative technique does not imply an explicit computation of the weight distribution of the codes, nor that of the dual codes. Moreover, the iterative technique from Chapter 4 enables the computation of P_u for channels other than the BSC, e.g., channels for which the probability of error ϵ may be different for every transmitted bit.

5.3.3 P_u for Various Code Generator Polynomials

The plots of aliasing probabilities as a function of test sequence length, reported in the previous chapter for different feedback polynomials, can be interpreted as plots of P_u as a function of code length for the codes generated by the corresponding generator polynomials.

---

† The probability of no errors occurring for a code word of n bits is $(1-\epsilon)^n$

In [Witzke 85] and [Fujiwara 85], the curves for the probability of an undetected error for various shortened versions of standard cyclic codes are reported as a function of the bit error probability ϵ, with the code length as a parameter. It would have been possible to do similarly here, but to illustrate an alternative way of reporting the performance of a shortened code, instead, the probability of an undetected error for shortened versions of the CCITT code as a function of the code length, with ϵ as a parameter, is reported. These curves for P_u for the shortened versions of the CCITT codes generated by the polynomial $1 + x^5 + x^{12} + x^{16}$ appear in Figs. 5.4 and 5.5, for various values of ϵ

In [Fujiwara 85], the values of ϵ which maximize P_u for a given code length are reported. Here, the length for which P_u is a maximum, for a given ϵ, can be found directly from the curves.

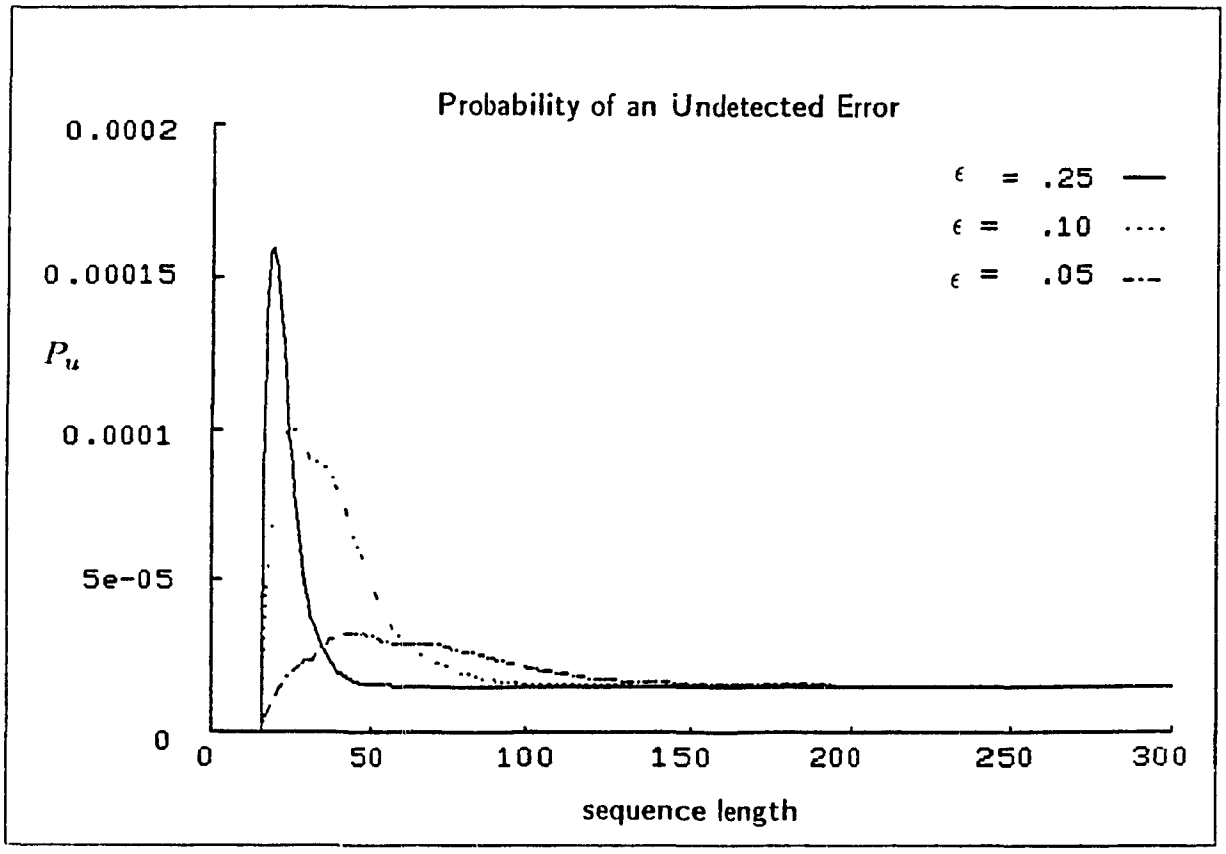

Figure 5.4 Probability of an undetected error; CCITT code, $g(x) = 1 + x^5 + x^{12} + x^{16}$

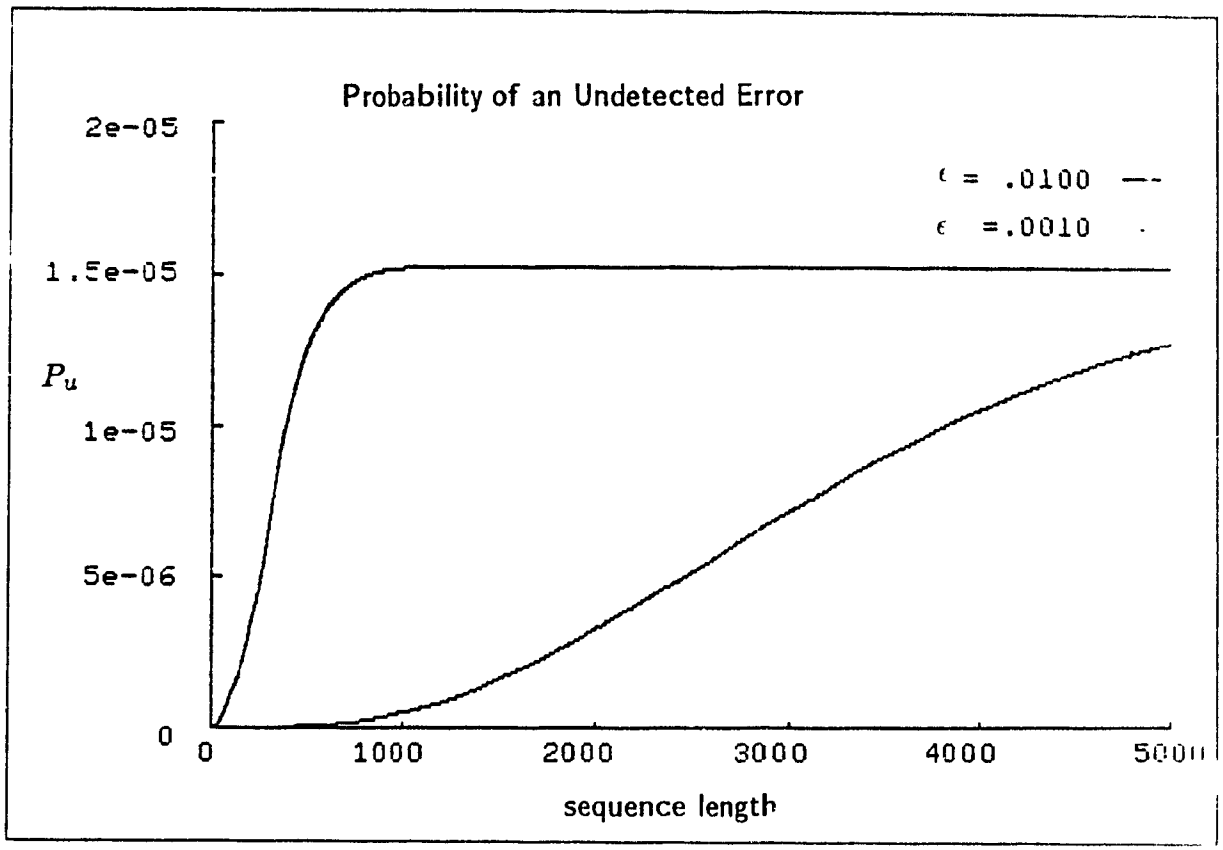

Figure 5.5 Probability of an undetected error, CCITT code, $g(x) = 1 + x^5 + x^{12} + x^{16}$.

These plots of P_u are all curves of the dynamic behaviour of m parity check equations. That is, for the CCITT generator polynomial, P_u is plotted assuming that 16 parity check bits are appended to the original information bits. However, one could also be interested in finding, also for various code lengths, the value of P_u that results if different numbers and combinations of parity check bits are appended to the original information bits, while keeping the same code generator polynomial. For example, in the case of CCITT, if only say 8 parity check bits were to be appended to the original information bits, one could be interested in generating those bits using the same CCITT polynomial instead of generating them using a polynomial of degree 8. In that case, without any additional computational efforts, the iterative technique proposed here enables the analysis of all $\binom{16}{8}$ ways of choosing 8 parity equations out of the 16 that arise with the given generator polynomial. That is, in the case of a CCITT code generator polynomial, in time complexity $O(q\ 2^{16})$, for each code length from 1 to q, the iterative

technique proposed here yields $2^{16} - 1$ values of P_u where each value corresponds to one one of the possible non-trivial combinations of the 16 parity equations that the given code polynomial generates. Given all the possible choices, the combination yielding the minimum P_u could be adopted.

5.3.4 Summary

Shortened cyclic codes are much used in communication systems. Unfortunately, because of their variable lengths, shortened codes are part of a class of codes for which the code distribution cannot be known in general. However, from the recent literature, there is a need and thus interest in computing the probability of undetected error for various shortened cyclic codes. In fact, such probability is used as a measure of the performance of a particular code. However, to date, no particularly effective method for computing such probabilities had been proposed. This thesis contributes an efficient technique for computing the probability of an undetected error for any shortened version of a cyclic code.

The fundamental difference between the technique proposed here and every other known approach for achieving the same end is that the technique proposed here is not based on an explicit calculation of the weight distribution of the code for calculating the probability of an undetected error. On the other hand, if one is indeed interested in finding the weight distribution of a code for a given length, it can be done from the calculated values of P_u for different values of ϵ. That is, given the values of P_u for a given code length for l different values of ϵ, a system of l linear equations can be set up and solved to find the weight distribution of the code for that length. Other than its simplicity and efficiency, the technique proposed here is also attractive for its capability of handling channels more complex than BSC, i.e., channels with varying probabilities of error may be handled at no extra analysis cost. Moreover, this control on the probability of error ϵ of individual bits enables different code polynomials to be very easily evaluated for their performance in regard to the detection of burst errors

Finally, the proposed iterative technique possesses another important facet (mentioned only briefly). Given a code generator polynomial, this important facet is

that the technique does not only yield the full dynamic behaviour of P_u resulting from increasing the code length in the case where the number of parity check bits is kept constant, i.e., equal to the degree of the generator polynomial. Given the code generator polynomial, the proposed iterative technique yields the complete dynamic behaviour of P_u resulting from increasing the code length for the situations where different numbers of parity check bits are appended to the original information bits. Hence, for a given code generator polynomial, the technique enables two dimensions to be readily analyzed, the code length n itself, and the number and choice of parity check bits. This facet of the technique seems very promising for its potential usefulness in designing good codes, and hence deserves to be explored further.

Chapter 6 Conclusion

Several different design for testability (DFT) techniques aimed at reducing the soaring costs associated with testing large and complex logic circuits have recently been proposed. Built-in self test (BIST) is one particular DFT approach that is gaining increasing support from the design, test, and manufacturing communities. In turn, several different BIST schemes have been proposed. One popular BIST scheme for unstructured combinational logic is one where pseudorandom test patterns generated by a maximal-length linear feedback shift register (LFSR) are applied to the circuit under test (CUT), and where the latter's response is compacted also using an LFSR. LFSR-based compaction is well-known under the name of "signature analysis". This dissertation focused on such BIST schemes. More specifically, the assessment of the quality of BIST schemes that use signature analysis constituted the global motivation for the work presented in this dissertation.

Depending on the test strategy adopted and the analytical and/or simulation tools available, different measures of the quality of a test strategy are possible. One frequently used measure of quality is the fault coverage. Hence, as for any other test strategy, the establishment of the fault coverage achieved by a given BIST scheme is crucial. For BIST circuits, it is desirable to report the test quality using the same fault coverage measures as those used in more conventional testing strategies. Unfortunately, the compaction stage in BIST complicates the matter. In some cases it prevents the usual fault coverage measures from being used as they would be in non-BIST cases.

The reasons why the usual fault coverage measures cannot always be used

was explained in Chapter 2. The possible aliasing in BIST is the source of the problem. The amount of aliasing must be measured before any claim on fault coverage may be made. In the same way that several definitions of fault coverage exist, several measures of aliasing also exist. In the past however, these measures of aliasing have often been misunderstood and hence misused. The intention in Chapter 2 was to provide clear definitions of aliasing measures. Two broad classes of measures were defined; those based on the fault domain deception volume, and those based on the error domain deception volume.

The measures of aliasing based on the fault domain deception volume are those that may be directly used with usual fault coverage measures. However, measuring the fault domain deception volume generally requires the use of full fault simulation. The high cost of the fault simulation of large circuits may in some cases preclude this approach entirely. Proposed alternative techniques to fault simulation are probabilistic in nature. In the case of BIST, probabilistic fault coverage measures imply measures of aliasing based on the error domain deception volume. Typically, a particular type of statistical assumption was made on the error domain deception volume, and from this assumption, claims on the fault coverage followed. Unfortunately, this assumption, i.e., the assuming of a uniform distribution of error sequences, is generally not justifiable. More justifiable is the assumption of a binomial distribution of error sequences. Such an assumption had already been used for calculating the probabilistic fault coverage in testing strategies that do not include circuit response compaction. However, at the time of the initiation of the research reported in this dissertation, only very few research attempts had been aimed at analyzing the problem of aliasing under the binomial distribution of error sequences. For signature analysis, an asymptotic result had recently been proven, along with other qualitative results. However, no computationally feasible technique existed to measure the aliasing of signature analysis schemes under the assumption of a binomial distribution of error sequences. Such techniques form the core of this dissertation.

Two basic techniques for calculating the probability of aliasing of signature analysis registers were developed. In Chapter 3, the first one was presented. The tech-

nique essentially consists of capturing the behaviour of the signature analysis register through a boolean equation for each of its stages. Such equations can be obtained in linear time and space complexity. The equations for a single stage were shown to constitute good first-degree approximations to the joint behaviour of the multiple stages of the signature register. Therefore, the analysis of the behaviour of single stages of a signature register provides heuristics that are useful for making important design decisions, e.g., the type of feedback that should be adopted, or what number of test patterns should be applied to the CUT for the signature analysis register to approach its asymptotic behaviour.

For making quantitatively more accurate and stronger claims on the aliasing problem, the analysis of only one stage is insufficient. To make such claims, the joint behaviour of all the stages of a signature analysis register must be studied. Given the boolean equations for the individual stages of a signature analyzer, solving for the probability of aliasing was shown to be transformable to a standard signal probability problem for a combinational circuit where the number of inputs to the circuit corresponds to the sequence length. For the general case, i.e., for signature registers characterized by any feedback polynomials, solutions to this problem require a computational complexity that is exponential in the sequence length. However, for feedback configurations characterized by polynomials $1+x^m$, where m is the number of stages of the signature register, closed-form expressions were obtained for the probability of aliasing, as a function of three parameters: the sequence length, the number of stages of the signature register, and the probability of error. In the case of registers characterized by primitive polynomials, expressions for upper bounds on the probability of aliasing, also as a function of the sequence length, the number of stages, and the probability of error, were derived.

In Chapter 4, a completely different approach for the computation of the probability of aliasing was proposed. This approach is iterative in nature. It is based on recursion equations in two dimensions; one dimension being the sequence length (time), and the second being the number of stages of the register involved (space). The complete technique proposed for iteratively computing the probability of aliasing involves $2^m - 1$ recursion equations, for a register with m stages. However, using

the proposed notation, this exponential number of equations was reduced to only two relatively simple general equations, valid for any feedback polynomial. These general equations make the software implementation of the technique particularly easy. Because of its iterative nature and the exponential number of recursion equations involved, the technique yields a solution for the aliasing probability linear in the sequence length, and exponential in the signature register's number of stages. Of course, the exponential complexity prevents signature registers of any size from being analyzed. Nonetheless, the relative simplicity of the calculations involved enables registers of sizes of interest in practice to be analyzed readily. Examples were given in Chapter 4.

In regard to their respective computational complexities, an interesting parallel can be drawn between the techniques developed in Chapters 3 and 4 and the MacWilliams identity, a well-known identity for code weight distributions [Lin 83]. As seen in Chapter 5, the problem of aliasing in the context of BIST parallels the probability of an undetected error in the context of coding theory. The usual methods for determining the probability of an undetected error require the determination of the weight distribution of the code determined by the feedback configuration of the LFSR. Interestingly, while the technique in Chapter 3 requires a complexity that is exponential in the sequence length i.e., $O(2^n)$, that of the technique in Chapter 4 is exponential in the size of the signature register, i.e., $O(2^m)$. The MacWilliams identity permits the same sort of reduction for determining the weight distribution of codes. That is, while the direct approach for determining the weight distribution of an (n, k) code, where $m = n - k$, is to analyze the 2^k code words, the alternative is to analyze the $2^{n-k} = 2^m$ code words of the dual code and then use the MacWilliams identity. Therefore, the MacWilliams identity permits a reduction in complexity of the same order that the iterative technique procures.

Other than its computational feasibility when compared to any other known approach, the advantage of the iterative technique developed in Chapter 4 is that it enables distributions of error sequences other than the binomial distribution to be analyzed at no extra computational cost. That is, the iterative nature of the technique permits every bit of an error sequence to be characterized by its own particular proba-

bility of error. This enables several types of errors or distributions to be analyzed, e.g., different dependent errors, or burst errors. This variability also enables the study of aliasing in biased random testing applications.

Applications of the analysis techniques in regard to test length calculations and test confidence were discussed in Chapter 5. The lack of efficient analysis techniques for calculating the probability of aliasing under error sequence distributions, other than the unreasonable uniform distribution, previously prevented the effects of LFSR-based compaction from being justifiably taken into account. Thus, this dissertation contributes analysis tools that are essential for establishing the fault coverage using probabilistic approaches, i.e., when deterministic methods such as full fault simulation are not possible or feasible.

The analysis techniques developed in Chapters 3 and 4 were essentially for single-input signature registers, and hence apparently more particularly suited for single-output circuits. However, extensions of the proposed analysis techniques to multi-output circuits were discussed in Chapter 5.

Finally, the techniques developed for computing the probability of aliasing in the context of BIST have important ramifications in the context of coding theory. That is, the iterative technique for computing the probability of aliasing may be used as an efficient technique for computing the probability of an undetected error for shortened cyclic codes. The technique enables the computation of the probability of an undetected error for the standard binary symmetric channel, as well as for more general channel models if need be. Finding efficient algorithms for computing the probability of undetected errors of shortened codes continues to attract several researchers' attention. In this respect, this dissertation makes an important contribution. Other coding theory applications of the technique remain to be explored.

In regard to other suggestions for further work, it may be possible to find ways of reducing the complexity of the iterative technique for all or only certain types of feedback configurations. Also, finding different types of approximations that would reduce the complexity of the iterative technique from exponential to polynomial may

be possible. For BIST applications, much interest has recently been expressed for non-linear compaction structures like cellular automata [Hortensius 87], or the compaction structure that arises in the circular self test scheme [Krasniewski 87] Thus, determining if and how the iterative technique of Chapter 4 could be generalized for handling non-linear compaction structures would also constitute an interesting pursuit of the work presented here. With respect to more global but perhaps also more useful issues more empirical work should be performed to establish the validity of the assumption of the binomial distribution of error sequences Such empirical results, combined with analytical work of the type presented in this dissertation should perfect probabilistic fault coverage analysis tools, and consequently make them trustworthy alternatives to costly deterministic approaches such as those based on fault simulation

References

[Aho 83] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *Data Structures and Algorithms*. Addison Wesley, 1983.

[Aitken 88] R.C. Aitken and V. K. Agarwal, "Aliasing Probability of Non-Exhaustive Randomized Syndrome Tests," *Proc. Int. Conf. on Computer-Aided Design*, Nov. 1988.

[Ando 80] H. Ando, "Testing VLSI with Random Access Scan," *Digest of Papers, COMPCON 80*, Feb. 1980, pp. 50-52.

[Armstrong 72] D. B. Armstrong, "A Deductive Method for Simulating Faults in Logic Circuits," *IEEE Trans. on Comp.*, Vol. C-21, No. 5, May 1972, pp. 464-471.

[Bardell 87] P. H. Bardell, W. H. McAnney, and J. Savir, *Built-In Self Test for VLSI: Pseudorandom Techniques*, New York: John Wiley & Sons Inc., 1987

[Benowitz 75] N. Benowitz, D. F, Calhoun, G.E. Alderson, J. E. Bauer, and C.T. Joeckel, "An Advanced Fault Isolation System for Digital Logic", *IEEE Trans. on Comp.*, Vol. C-24, No. 5, May 1975, pp. 489-497.

[Bhavsar 84] D. Bhavsar and B. Krishnamurthy, "Can we Eliminate Fault Escape in Self-Testing by Polynomial Division (Signature Analysis)?", *Proc. Int. Test Conf.*, Oct. 1984, pp 134-139

[Bloom 79] D. M. Bloom, *Linear Algebra and Geometry*, Cambridge: Cambridge University Press, 1979.

[Bose 82] A. K. Bose et al. "A Fault Simulator for MOS LSI Circuits," *Proc. 19th Design Automation Conf.*, June 1982, pp. 400-409.

[Bozorgui-Nesbat 80] S. Bozorgui-Nesbat and E.J. McCluskey, "Structured Design for Testability to Eliminate Test Pattern Generation," *Proc. 10th Int. Symp. on Fault-Tolerant Computing*, June 1980, pp. 158-163.

[Brglez 84] F. Brglez, P Pownall, and R. Hum, "Applications of Testability Analysis: From ATPG to Critical Delay Path Tracing", *Proc. Int. Test Conf.*, Oct. 1984, pp. 705-712.

[Carter 82] W. C. Carter, "The Ubiquitous Parity Bit", *Proc. 12th Int. Symp. on Fault-Tolerant Computing*, June 1982, pp. 289-296.

[Chang 70] H. Y. Chang, E.G. Manning and G. Metze, *Fault Diagnosis of Digital Systems*. Wiley-Interscience, N.Y., 1970.

[Chin 87] C. Chin, and E. J. McCluskey, "Test Length for Pseudorandom Testing", *IEEE Trans. on Comp*., Vol C-36, No 2, Feb 1987, pp 252-256.

[Cox 88] H. Cox, A. Ivanov, J. Rajski, and V. K. Agarwal, "On Multiple Fault Coverage Measures and Aliasing Probability Measures", *Proc Int Test Conf.*, Sept. 1988.

[David 78] R. David, "Feedback Shift Register Testing", *Proc 8th Int Symp on Fault-Tolerant Computing*, June 1978, pp 103-107

[David 80] R. David, "Testing by Feedback Shift Register", *IEEE Trans on Comp.*, Vol C-29, No 7, July 1980, pp 669-673

[David 84] R. David, "Signature Analysis of Multi-Output Circuits", *Proc 14th Int. Symp. on Fault-Tolerant Computing*, June 1984, pp 366-371.

[Design & Test 85] *IEEE Design & Test Magazine*, Vol. 2, No. 2, April 1985

[Eichelberger 78] E. B. Eichelberger and T.W. Williams, "A Logic Design Structure for LSI Testability," *J. Design Automat Fault Tolerant Computing*, Vol. 2, No. 2, pp 165-178.

[Eichelberger 83] E. B. Eichelberger and E. Lindbloom, "Random Pattern Coverage Enhancement for LSSD Logic Self-Test," *IBM J. Res Develop*, Vol. 27, May 1983, pp 265-272

[Frohwerk 77] R.A. Frohwerk, "Signature Analysis: A New Digital Field Service Method", *Hewlett- Packard Journal*, May 1977, pp 2-8

[Fujiwara 83] H. Fujiwara and T. Shimono, "On the Acceleration of Test Generation Algorithms," *IEEE Trans on Comp*., Vol C-32, No 12, Dec 1983, pp. 1137-1144.

[Fujiwara 85] T. Fujiwara, T. Kasami, A. Kitai, and S Lin, "On the Undetected Error Probability for Shortened Hamming Codes, *IEEE Trans on Comm.*, Vol. COM-33, June 1985, pp 570-574

[Fujiwara 86] T. Fujiwara, and T. Kasami, "Error Detecting Capabilities of the Shortened Hamming Codes adopted for Error Detection in IEEE Standard 802 3", *IEEE Int. Symp on Information Theory*, 1986, p 65.

[Funatsu 75] S. Funatsu, N Wakatsuki, and T Arima, "Test Generation Systems in Japan," *Proc. 12th Design Automation Symposium*, June 1975, pp. 114-122.

[Galiay 80] J. Galiay, Y. Crouzet, and M. Vergniault, "Physical vs. Logical Fault Models of MOS LSI Circuits: Impact on their Testability," *IEEE Trans. on Comp.*, Vol. C-29, No. 6, June 1980, pp. 527-531.

[Garey 78] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, New York: W. H. Freeman and Company, 1978.

[Goel 80] P. Goel, "Test Generation Costs Analysis and Projections", *Proc. of 17th Design Automation Conf.*, June 1980, pp. 77-84.

[Goel 81] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," *IEEE Trans. on Comp.*, Vol. C-30, No. 3, Mar 1981, pp. 215-222.

[Golomb 82] S. W. Golomb, *Shift Register Sequences*, Calif.: Aegean Park Press, 1982.

[Gupta 88] S. K. Gupta and D. K. Pradhan, "A Framework for Designing & Analyzing BIST Techniques," *Proc. Int. Test Conf.*, Sept. 1988.

[Hassan 83] S. Z. Hassan, D. J Lu, E. J. McCluskey, "Parallel Signature Analyzers – Detection Capability and Extensions," *Proc. 26th IEEE Computer Society Int. Conf.*, COMPCON Spring 1983, pp. 440-445.

[Hassan 84] S. Z. Hassan and E. J. McCluskey, "Increased Fault Coverage Through Multiple Signatures," *Proc. 14th Int. Symp. on Fault-Tolerant Computing*, June 84, pp. 354-359.

[Hayes 76] J. P. Hayes, "Check Sum Test Methods", *Proc. 6th Int. Symp. on Fault-Tolerant Computing*, June 1976, pp. 114-119.

[Hlawiczka 86] A. Hlawiczka, "Compression of Three-State Data Serial Streams by Means of a Parallel LFSR Signature Analyzer," *IEEE Trans. on Comp.*, Vol. C-35, No. 8, Aug. 1986, pp. 732-741.

[Hortensius 87] P. D Hortensius, "Parallel Computation of Non-Deterministic Algorithms in VLSI, *Ph. D. thesis*, University of Manitoba, Winnipeg, Canada, 1987.

[Hurst 85] S. L. Hurst, D. M. Miller, and J. C. Muzio, *Spectral Techniques in Digital Logic*, London: Academic Press, 1985.

[Ivanov 87] A. Ivanov, and V. K Agarwal, "On A Fast Method to Monitor the Behaviour of Signature Analysis Registers", *Proc. Int. Test Conf.*, Sept. 1987, pp. 645-655.

[Ivanov 88] A. Ivanov, and V. K. Agarwal, "An Iterative Technique for Calculating Aliasing Probability of Linear Feedback Signature Registers,"

Proc. 18th Int. Symp. on Fault-Tolerant Computing, June 1988, pp. 70-75.

[Jain 85] S. K. Jain and V. D. Agrawal, "Statistical Fault Analysis," *IEEE Design & Test Magazine*, Feb. 1985, pp. 38-44

[Kasami 83] T. Kasami, T. Klove, and S. Lin, "Linear Block Codes for Error Detection," *IEEE Trans. on Information Theory*, Vol. IT-29, Jan 1983, pp 131-136.

[Koenemann 79] B. Koenemann, H. Mucha, and G. Zwiehoff, "Built-In Logic Block Observation Technique," *Proc. Int. Test Conf.*, Oct. 1979, pp 37-41.

[Kohavi 70] , Z. Kohavi, *Switching and Finite Automata Theory*, New York McGraw-Hill Book Co., 1970.

[Korznik 65] V. I. Korznik, "Bound on Undetected Error Probability and Optimum Group Codes in a Channel with Feedback, "*Radiotekhnika*, Vol. 1, 1965, pp 27-33 (English translation: *Telecommun Radio Eng.*, Vol 2, Jan. 1965, pp 87-92)

[Krasniewski 87] A. Krasniewski and S. Pilarski, "Circular Self-Test Path: A Low Cost BIST Technique," *Proc. 24th ACM/IEEE Design Automation Conference*, June 1987, pp 407-415.

[Krishnamurthy 86] B. Krishnamurthy, and I G. Tollis, "Improved Techniques for Estimating Signal Probabilities", *Proc. Int Test Conf*, Sept. 1986, pp. 244-251

[Krishnamurthy 88] B. Krishnamurthy, *private communication*, June 1988

[Lagemaat 87] D van de Lagemaat and H. Bleeker, "Testing a Board with Boundary Scan," *Proc. Int. Test Conf*, Sept. 1987, pp 724-729

[Leung 76] S. K. Leung-Yan-Cheong, and M E. Hellman, "Concerning a Bound on Undetected Error Probability", *IEEE Trans on Information Theory*, Vol. IT-22, March 1976, pp. 235-237

[Leung 79] S. K. Leung-Yan-Cheong, E. R. Barnes, and D. U Freidman, "On Some Properties of the Undetected Error Probability of Linear Codes," *IEEE Trans. on Information Theory*, Vol IT-25, Jan 1979, pp 110-112

[Levendel 81] Y. H. Levendel and P R. Menon, "Fault Simulation Methods Extensions and Comparisons", *Bell Sys. Tech. Jour.*, Vol. 60, No 9, Nov. 1981, pp. 2235-2258.

[Lin 83] S. Lin, and D J. Costello Jr., *Error Control Coding: Fundamentals and Applications*, Englewood Cliffs, New Jersey: Prentice-Hall, 1983.

[Lisanke 86] R. Lisanke, F. Brglez, A. de Geus, and D. Gregory, "Testability-Driven Random Pattern Generation," *Proc. Int. Conf. on Integrated Circuit Design*, Nov. 1986.

[Maamari 88] F. Maamari and J. Rajski, "A Fault Simulation Method Based on Stem Regions," *Proc. Int. Conf. on Computer-Aided Design*, Nov 1988.

[Mann 80] W. R. Mann, "Microelectronic Device Test Strategies – A Manufacturer's Approach", *Proc. Int. Test Conf.*, Nov. 80, pp. 195-202.

[Maunder 87] C. Maunder and F. Beenker, "Boundary Scan: A Framework for Structured Design-for-Test," *Proc. Int. Test Conf.*, Sept. 1987, pp. 714-723.

[McCluskey 84] E. J McCluskey, "Verification Testing – A Pseudo-Exhaustive Test Technique," *IEEE Trans. on Comp.*, Vol. C-33, No. 6, June 1984, pp. 541-546.

[McCluskey 85] E. J. McCluskey, "Built-in Self Test Techniques", *IEEE Design & Test Magazine*, Vol. 2, No. 2, April 1985, pp. 21-28.

[McCluskey 86] E.J. McCluskey, *Logic Design Principles with Emphasis on Testable Semicustom Circuits*, Prentice-Hall, Englewood Cliffs, N.J., 1986.

[McCluskey 87] E. J. McCluskey, S. Makar, S. Mourad, and K. D. Wagner, "Probability Models for Pseudorandom Test Sequences," *Proc. Int. Test Conf.*, Sept. 1987, pp 471-479.

[Miller 84] D.M. Miller and J. C. Muzio, "Spectral Fault Signatures for Single Stuck-at Faults in Combinational Networks," *IEEE Trans. on Comp.*, Vol. C-33, No. 8, Aug. 1984, pp. 765-769.

[Miller 85] M. J. Miller, "The Probability of Undetected Error for Shortened Cyclic Codes", *IEEE Int. Symp. on Information Theory*, 1985, p. 111.

[Muzio 83] J.C. Muzio and D.M. Miller, "Spectral Fault Signatures for Internally Unate Combinational Networks," *IEEE Trans. on Comp.*, Vol. C-32, 1983, pp 1058-1062.

[Myers 83] M.A. Myers, "An Analysis of the Cost and Quality Impact of LSI-VLSI Technology on PCB Test Strategies", *Proc. Int. Test Conf.*, Oct 1983, pp. 382-395.

[Parker 75] K. P Parker, and E. J. McCluskey, "Probabilistic Treatment of Combinational Networks", *IEEE Trans. on Comp.*, Vol. C-24, No. 6, June 1975, pp. 668-670.

[Parker 76] K. P. Parker. "Compact Testing: Testing with Compressed Data", *Proc. 6th Int. Symp. on Fault-Tolerant Computing*, June 1976, pp 93-98.

[Peterson 72] W. W. Peterson, and E. J. Weldon, *Error Correcting Codes*, 2nd ed., Cambridge: MIT Press, 1972.

[Rajski 87] J. Rajski and H. Cox, "A Method of Test Generation and Fault Diagnosis in Very Large Combinational Circuits," *Proc. Int Test Conf.*, Sept. 1987, pp. 932-943.

[Roth 67] J. P. Roth, W. G. Bouricuis, and P. R. Schneider, "Programmed Algorithms to Compute Tests to Detect and Distinguish Between Failures in Logic Circuits", *IEEE Trans. on Comp*, Vol C-16, No 10, Oct. 1967, pp. 567-589.

[Savir 80] J. Savir, "Syndrome-testable Design of Combinational Circuits", *IEEE Trans. on Comp*, Vol. C-29, No. 6, June 1980, pp. 442-451

[Savir 83] J. Savir, G. Ditlow, and P. H. Bardell, "Random Pattern Testability", *Proc. 13th Int. Symp. on Fault-Tolerant Computing*, June 1983, pp. 80–89

[Savir 84] J. Savir, G Ditlow, and P. H. Bardell, "On Random Pattern Test Length", *IEEE Trans. on Comp.*, Vol. C-33, No. 6, June 1984, pp 467-474.

[Saxena 86] N. R. Saxena and J. P. Robinson, "Accumulator Compression Testing," *IEEE Trans. on Comp.*, Vol. C-35, No. 4, April 1986, pp. 317-321.

[Schulz 88] M. H. Schulz, E. Trishler, and T. M. Sarfert, "SOCRATES: A Highly Efficient Automatic Test Pattern Generation System", *IEEE Trans. on CAD of ICAS*, Jan. 1988, pp. 125-137

[Seth 85] S. C. Seth, L. Pan, and V. D. Agrawal,"PREDICT: Probabilistic Estimation of Digital Circuit Testability," *Proc. 15th Int. Symp on Fault-Tolerant Computing*, June 1985, pp 220-225

[Shedletsky 77] J. J. Shedletsky, "Random Testing: Practicality versus verified Effectiveness", *Proc. 7th Int. Symp. on Fault-Tolerant Computing*, June 1977, pp. 175–179.

[Smith 80] J E. Smith, "Measures of the Effectiveness of Fault Signature Analysis", *IEEE Trans. on Comp.*, Vol. C-29, No 6, June 1980, pp 510–514.

[Sridhar 82] T. Sridhar, D. S. Ho, T. J. Powell, and S. M. Thatte, "Analysis and Simulation of Parallel Signature Analyzers", *Proc. Int. Test Conf*, Oct. 1982, pp. 656-661.

[Stewart 77] J. H. Stewart, "Future Testing of Large LSI Circuit Cards," *Proc. 1977 Semiconductor Test Symposium,* Oct., 1977, pp. 6-15.

[Susskind 81] A. K Susskind, "Testing by Verifying Walsh Coefficients", *Proc. 11th Int. Symp. on Fault-Tolerant Computing,* June 1981, pp. 206-208.

[Tanenbaum 81] A. S. Tanenbaum, A.S., *Computer Networks.* Englewood Cliffs, New Jersey: Prentice-Hall, 1981.

[Trivedi 82] K. S. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications,* Englewood Cliffs, New Jersey: Prentice-Hall, 1982

[Tucker 80] A. Tucker, *Applied Combinatorics,* New York: John Wiley & Sons, 1980.

[Valiant 79] L. G. Valiant "The Complexity of Enumeration and Reliability Problems", *SIAM Jour. Comput.,* Vol. 8, No. 3, Aug. 1979, pp. 410-421.

[Wadsack 78] R. L. Wadsack, "Fault Modelling and Logic Simulation of CMOS and MOS Integrated Circuits," *Bell Syst. Tech. Jour.,* May-June 1978, pp. 1149-1475.

[Wagner 87] K. D. Wagner, C. K. Chin, and E. J. McCluskey, "Pseudorandom Testing", *IEEE Trans. on Comp.,* Vol. C-36, No. 3, March 1987, pp. 332-343.

[Waicukauski 88] J. A. Waicukauski and E. Lindbloom, "Fault Detection Effectiveness of Weighted Random Patterns," *Proc. Int. Test Conf.,* Sept. 1988, pp. 245-255.

[Williams 82] T.W. Williams and K. P. Parker, "Design for Testability – A Survey", *IEEE Trans. on Comp.,* Vol. C-31, No. 1, Jan. 1982, pp. 2-15.

[Williams 85] T. W. Williams, "Test Length in a Self-Testing Environment," *IEEE Design & Test Magazine,* Vol. 2, No. 2, April 1985, pp. 59-63.

[Williams 86] T. W. Williams, W. Daehn, M. Gruetzner, and C. W. Starke, "Comparison of Aliasing Errors for Primitive and Non-Primitive Polynomials", *Proc. Int. Test Conf.,* Sept. 1986, pp. 282-288.

[Williams 87] T. W. Williams, W. Daehn, M. Gruetzner, and C.W. Starke, "Aliasing Errors in Signature Analysis Registers", *IEEE Design & Test Magazine,* April 1987, pp. 39-45, or Williams et. al., "Aliasing Errors with Primitive and Non-Primitive Polynomials" *Proc. Int. Test Conf.,* Sept. 1987, pp. 637-644.

[Witzke 85] K. A. Witzke, and C. Leung, "A Comparison of Some Error Detecting CRC Code Standards," *IEEE Trans. on Comm.*, Vol. COM-33, Sept. 1985, pp. 996-998.

[Wolf 82] J. K. Wolf, A. M. Michelson, and A. H. Levesque, "On the Probability of Undetected Error for Linear Block Codes," *IEEE Trans on Comm.*, Vol. COM-30, Feb 1982, pp 317-324.

[Wunderlich 87] H-J. Wunderlich, "On Computing Optimized Input Probabilities for Random Tests", *Proc. 24th ACM IEEE Design Automation Conference*, June 1987, pp. 392-398.

[Zorian 86] Y. Zorian, and V. K. Agarwal, "A General Scheme to Optimize Error Masking in Built-In Self-Testing", *Proc. 18th Int. Symp on Fault-Tolerant Computing*, July 1986, pp. 410-415.

[Zorian 87] Y. Zorian, "Optimized Error Coverage in Built-In Self-Test by Output Data Modification", *Ph. D. thesis*, McGill University, Montreal, Canada, Sept. 1987.

Appendix A. Proof of Theorem 3.3

Prior to proving Theorem 3.3, two lemmas are proven. The statement and proof of these first requires a few preliminaries.

Assuming a particular LFSR characterized by a primitive polynomial of degree m, and assuming a test sequence of length n, the $n \times n$ matrix of coefficients C defined in Chapter 3 can easily be found. For example, for the polynomial $f(x) = 1 + x^2 + x^3$, with $n = 6$, C is:

$$\mathbf{C} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}. \tag{A.1}$$

Given C, for $n \geq m$, consider only the $m \times n$ submatrix of C formed by taking the last m rows of C, i.e., rows $n, n-1, \ldots, n-m+1$. Let this resultant matrix of m rows and n columns be denoted by $\mathbf{C}_{mn}$. For $n < m$, $\mathbf{C}_{mn}$ is not defined. For C given in eq. (A.1), $\mathbf{C}_{mn}$ is:

$$\mathbf{C}_{mn} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}. \tag{A.2}$$

Define a two dimensional *window* W to be any $m \times m$ submatrix of $\mathbf{C}_{mn}$ formed by taking any m consecutive columns of $\mathbf{C}_{mn}$. For a matrix $\mathbf{C}_{mn}$, there is a total of $n - m + 1$ possible windows. Of these $n - m + 1$ windows, only $\lfloor n/m \rfloor$ non-overlapping (that have no common columns) windows can be found. For $\mathbf{C}_{mn}$ given in eq. (A.2), there are $6 - 3 + 1 = 4$ possible windows. These are:

$$\mathbf{W}_1 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} \tag{A.3}$$

$$\mathbf{W}_2 = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \tag{A.4}$$

$$\mathbf{W}_3 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \tag{A.5}$$

$$\mathbf{W}_4 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}. \tag{A.6}$$

Among $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3, \mathbf{W}_4$, only $\lfloor 6/3 \rfloor = 2$ non-overlapping windows can be found, i.e., $\mathbf{W}_1$ and $\mathbf{W}_4$. For the polynomial of this particular example, the sequence of states (of periodicity 7) that the LFSR goes through when initialized to the "100" state and cycled in the autonomous mode is:

100, 010, 101, 110, 111, 011, 001, 100, ...

Notice that from the properties of C that follow from its definition given in Chapter 3, the m rows of any window $\mathbf{W}$ of C_{mn} obtained from a primitive polynomial correspond to m consecutive states of the LFSR when the latter is cycled in the autonomous mode. Moreover, for primitive polynomials, these m states are distinct because the period of the sequence generated by a primitive polynomial is $2^m - 1$. Hence, no state can repeat within only m shifts.

Consider each row of a window $\mathbf{W}$ to be an m-tuple over a field F. Since every window $\mathbf{W}$ is made up of m rows, every window $\mathbf{W}$ of C_{mn} is constituted by a set of m m-tuples. Now the following lemma can be proven.

Lemma A.1: The set of m m-tuples that constitute every window $\mathbf{W}$ of any C_{mn} obtained from a primitive polynomial forms a basis for an m-dimensional vector space over a field F.

Proof: Consider the first m states that any LFSR characterized by a primitive polynomial goes through when the latter is initialized to the "1000...00" state. Because of the

shifting property of the LFSR, the latter sequence of states $s_1, s_2, s_3, \ldots, s_m$, in matrix form, will be:

$$\begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_m \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 \\ X & 1 & 0 & \ldots & 0 \\ X & X & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ X & X & X & \ldots & 1 \end{pmatrix}. \qquad (A.7)$$

where X denotes 0 or 1. Clearly, because of the ones in the main diagnonal, the m m-tuples form a basis for an m-dimensional vector space

In the autonomous mode, at every shift, the current state of the LFSR is mapped onto a unique successor state. This one-to-one and onto mapping from an element in a space onto another element in the same space is a linear transformation that may be described by an $m \times m$ matrix, where m is the number of stages of the LFSR [Golomb 82]. Letting that matrix be A, if the state of the LFSR at time i is s_i, then the state at time $i+1$ is $s_{i+1} = s_i A$.

If the linear transformation described by the matrix A is applied j times to each of the states $s_1, s_2, \ldots, s_m$, the set of m consecutive states that results is: $(s_1A^j, s_2A^j, s_3A^j, \ldots, s_mA^j)$, where A^j denotes the j^{th} power of A. In general, any window W of C_{mn} is constituted by such m consecutive states, i.e., for each window, there exists a j such that the m rows of W correspond to the states: $(s_1A^j, s_2A^j, s_3A^j, \ldots, s_mA^j)$. From linear algebra [Bloom 79], applying a one-to-one linear transformation to each of the m elements that form a basis for an m-dimensional vector space yields a set of m unique elements that also form a basis for an m-dimensional vector space. Therefore, since the set of states $s_1, s_2, \ldots, s_m$ form a basis for an m-dimensional vector space, so does the set of states $(s_1A^j, s_2A^j, s_3A^j, \ldots, s_mA^j)$. Hence, the m m-tuples from every window W of C_{mn} obtained from a primitive polynomial constitute a basis for an m-dimensional vector space. ∎

Define a *cover* for a particular window W to be a set of elements w_{ij} of W such that $w_{ij} = 1$ and *exactly* one element from each column of W is selected. For

example, for $\mathbf{W}_2$ above, the elements w_{11}, w_{22} and w_{13} constitute a cover. Now define a *perfect cover* as a cover that contains *exactly* one element from each row of $\mathbf{W}$. For the window $\mathbf{W}_2$ above, w_{11}, w_{22} and w_{13} does not constitute a perfect cover because no elements come from the third row, while two come from the first row. However, the set of elements w_{11}, w_{22} and w_{33} does constitute a perfect cover for $\mathbf{W}_2$. †

Lemma A.2: For all n and m such that $n \geq m$, for the submatrix $\mathbf{C}_{mn}$ formed by taking the last m rows, i.e., rows $n, n-1, n-2, \ldots, n-m+1$, from the matrix of coefficients $\mathbf{C}$ generated from a primitive polynomial, a perfect cover is guaranteed to exist for each of the $n-m+1$ possible windows $\mathbf{W}$ of $\mathbf{C}_{mn}$.

Proof: For the lemma to hold, n must be greater or equal to m since the definition of the matrix $\mathbf{C}_{mn}$, and hence also the definition of a window $\mathbf{W}$, requires this condition to be true. From Lemma A.1 every window $\mathbf{W}$ contains a set of m-tuples that constitutes a basis for an m-dimensional vector space. By the definition of a basis for a vector space, it follows that at least one perfect cover for every window is guaranteed to exist [Bloom 79]. For example, by performing elementary row operations, i.e., simply interchanging rows of $\mathbf{W}$, the main diagonal of $\mathbf{W}$ can be made to be all ones, i.e., $w_{11} = w_{22} = \cdots = w_{mm} = 1$. The elements of this diagonal constitute a perfect cover for $\mathbf{W}$. ∎

Theorem 3.3: Assuming a primitive feedback polynomial and an error sequence of length n, an upper bound for $Z_n = pr(\text{all-zero state at time } n)$ is:

$$Z_n \leq \left(\frac{1 + |1-2\epsilon|^{\lfloor n/m \rfloor}}{2}\right)^m, \tag{A.8}$$

where $|\ |$ denotes "absolute value of." (For $n/m \gg 1$ this bound is essentially the envelope of eq. (3.15)).

The proof relies on the Full Range Cutting Algorithm [Savir 83]. Once the all-zero state probability problem is formulated as a signal probability problem, the

---

† Finding a *perfect cover* is equivalent to the problem of finding a *set of distinct representatives* from a collection of subsets, or finding a *maximal matching* in a bipartite graph [Tucker 80].

objective is to find an upper bound on the probability of a zero at the output G of the circuit in Fig. 3.9. For primitive polynomials, the circuit in Fig. 3.9 contains reconvergent fanout lines. For such circuits, the Cutting Algorithm yields bounds for the circuit's output signal probabilities. The procedure of the Cutting Algorithm is to leave uncut only one of the branches from a stem line and to cut all the remaining ones. The uncut branch lines retain the signal probability of the stem, and the cut branches are assigned the full range probability interval $[0, 1]$. Once a circuit is cut it becomes a tree. Hence, from then on, the signal probability bound calculations are straightforward. The rules for basic gates are given in [Savir 83].

Concentrating on only one of the m stages, assume the scenario shown in Fig. A.1 where the XOR gate has as its inputs q uncut lines and r cut lines. The q uncut lines each carry a 1-probability of ϵ, and the r cut lines carry the signal probability intervals $[0, 1]$. From the associative property of the XOR gate, this $(q+r)$-input gate is equivalent to a two-input XOR gate where one of the two inputs has the 0-probability $pr(0) = \frac{1+(1-2\epsilon)^q}{2}$ (from eq. (3.13)), and the other input has the 0-probability interval $[0, 1]$.

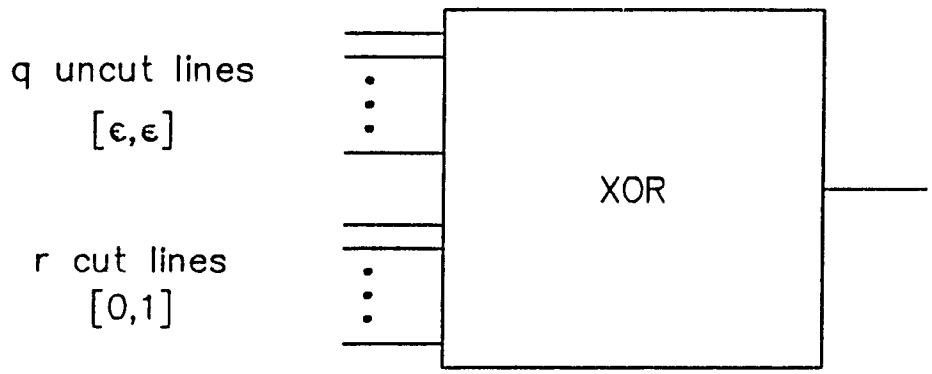

Figure A.1 XOR gate with q uncut lines and r cut lines.

From [Savir 83], one can calculate the following 0-probability interval at the output of the two-input equivalent gate: $pr(0) = [\frac{1+(1-2\epsilon)^q}{2}, 1 - \frac{1+(1-2\epsilon)^q}{2}] = [\frac{1+(1-2\epsilon)^q}{2}, \frac{1-(1-2\epsilon)^q}{2}]$. Assuming that all m stages have such 0-probability intervals and that they feed into an

OR gate (as in Fig. 3.9), the latter has the following 0-probability bounds at its output

$$\left[\left(\frac{1+(1-2\epsilon)^q}{2}\right)^m, \left(\frac{1-(1-2\epsilon)^q}{2}\right)^m\right]. \tag{A.9}$$

Assuming the length of the input error sequence to be n, then according to the Cutting Algorithm, n lines (branches) can remain uncut and thus assume their true signal probability. For a sequence of length n, if the matrix C_{mn} is formed, then the positions of the ones in any row j of C_{mn} correspond to the E_i's that feed into stage j. Alternatively, the position of the ones in the column i of C_{mn} corresponds to the fanout pattern for each error bit E_i. Consider for example C_{mn} given in eq. (A.2)

$$C_{mn} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}. \tag{A.10}$$

Letting the last row of C_{mn} correspond to the first (leftmost) stage of the LFSR, the second row of C_{mn} correspond to the second stage of the LFSR, and the first row of C_{mn} correspond to the third (rightmost) stage of the LFSR, then this particular C_{mn} reveals that for the test sequence length $n = 6$, the error bit E_1 is an input to stages 2 and 3, E_2 an input to stages 1, 2, and 3, E_3 an input to stages 1 and 2, E_4 an input to stages 1 and 3, E_5 an input to stages 2, and E_6 an input to stage 1 only.

Each window W of C_{mn} corresponds to a subset of m E_i's that fanout to the m stages of the LFSR. The Cutting Algorithm states that m lines (branches) can remain uncut. From earlier, a cover corresponds to choosing exactly one element for each column of a window W. Therefore, a cover for W is equivalent to selecting which branch of E_i not to cut. Then finding a *perfect cover* for W corresponds to finding a cutting pattern such that *each* of the m stages gets *exactly* one uncut line E_i. A total of $\lfloor n/m \rfloor$ windows over the matrix C_{mn} can be formed. From Lemma A.2, a perfect cover can be found for each of these windows when C_{mn} is generated by a primitive polynomial. Therefore, n uncut lines can be distributed among the m stages such that each stage receives at least $\lfloor n/m \rfloor$ uncut lines.

As an example, consider the circuit in Fig. 3.10. In the latter, inputs E_1, E_6 and E_7 do not fanout and hence do not have to be cut. This yields one uncut line for each stage. Then, for example, the input branch stemming from E_2 of gate XOR_1 could remain uncut, and similarly for E_3 for XOR_2, and E_4 for XOR_3. Hence, in this case, each XOR gate can have at least $\lfloor 7/3 \rfloor = 2$ uncut lines as inputs. This example illustrates the maximal length case, i.e., $n = 2^m - 1$. However, for non-maximal lengths, Lemma A.1 guarantees that at least $\lfloor n/m \rfloor$ lines per stage can retain their specific signal probabilities, i.e., remain uncut.

Consequently, if each stage receives $\lfloor n/m \rfloor$ uncut lines, from the equations for a two-input XOR gate [Savir 83], the 0-probability interval for the output of the circuit (output of OR gate) is:

$$\left[\left(\frac{1+(1-2\epsilon)^{\lfloor n/m \rfloor}}{2}\right)^m, \left(\frac{1-(1-2\epsilon)^{\lfloor n/m \rfloor}}{2}\right)^m\right]. \qquad (A.11)$$

Hence, an upper bound for the probability of the "all-zero" state for primitive polynomials is:

$$Z_n \leq max\left[\left(\frac{1+(1-2\epsilon)^{\lfloor n/m \rfloor}}{2}\right)^m, \left(\frac{1-(1-2\epsilon)^{\lfloor n/m \rfloor}}{2}\right)^m\right]. \qquad (A.12)$$

The above can be simplified to:

$$Z_n \leq \left(\frac{1+|1-2\epsilon|^{\lfloor n/m \rfloor}}{2}\right)^m. \qquad (A.13)$$

■

Appendix B. Proof of Theorem 4.5

Prior to giving the statement of Theorem 4.5 and providing a proof for it, three lemmas are proven.

Lemma B.1: Let A and B be two boolean variables, and let $\oplus$ denote the modulo-2 addition and $\cup$ denote the boolean OR operator, then

$$pr(A \oplus B = 1) = 2pr(A \cup B = 1) - pr(A = 1) - pr(B = 1).$$

Proof: From the definitions in [Parker 75],

$$pr(A \oplus B = 1) = 1 - pr(A \oplus B = 0). \tag{B.1}$$

The function $A \oplus B = 0$ can be expressed as a product of boolean sums:

$$(A \oplus B = 0) = (A \cup B = 0)(\overline{A} \cup \overline{B} = 0). \tag{B.2}$$

Since the two terms are mutually exclusive events, i.e., both cannot be satisfied simultaneously,

$$pr(A \oplus B = 0) = pr(A \cup B = 0) + pr(\overline{A} \cup \overline{B} = 0). \tag{B.3}$$

Also,

$$pr(A \cup B = 0) = 1 - pr(A \cup B = 1), \tag{B.4}$$

and

$$pr(\overline{A} \cup \overline{B} = 0) = 1 - pr(\overline{A} \cup \overline{B} = 1). \tag{B.5}$$

Using Lemma 4.2,

$$pr(\overline{A} \cup \overline{B} = 1) = 1 - pr(A \cup \overline{B} = 1) + pr(\overline{B} = 1). \tag{B.6}$$

Since $pr(\overline{B} = 1) = 1 - pr(B = 1)$, eq. (B.6) can be rewritten as:

$$pr(\overline{A} \cup \overline{B} = 1) = 2 - pr(B = 1) - pr(A \cup \overline{B} = 1). \tag{B.7}$$

Using Lemma 4.2 again for the term $pr(A \cup \overline{B} = 1)$ yields:

$$\begin{aligned} pr(\overline{A} \cup \overline{B} = 1) &= 2 - pr(B = 1) - [1 - pr(A \cup B = 1) + pr(A = 1)] \\ &= 1 - pr(B = 1) - pr(A = 1) + pr(A \cup B = 1). \end{aligned} \tag{B.8}$$

Hence, using eqs. (B.1)-(B.8),

$$\begin{aligned} pr(A \oplus B = 1) &= 1 - pr(A \oplus B = 0) \\ &= 1 - [pr(A \cup B = 0) + pr(\overline{A} \cup \overline{B} = 0)] \\ &= 1 - [1 - pr(A \cup B = 1)] - [1 - pr(\overline{A} \cup \overline{B} = 1)] \\ &= 1 - [1 - pr(A \cup B = 1)] - [1 - (1 - pr(B = 1) \\ &\qquad - pr(A = 1) + pr(A \cup B = 1))] \\ &= 2pr(A \cup B = 1) - pr(A = 1) - pr(B = 1). \end{aligned} \tag{B.9}$$

■

Let $S^1, S^2, \ldots S^n$ and be a set of boolean variables, with $0 \leq pr(S^i = 1) \leq 1$. Also let U be a boolean variable with $0 \leq pr(U = 1) \leq 1$. To simplify the notation, let $pr(S^i = 1)$ simply be denoted by $pr(S^i)$, and let $\sum$ represent modulo-2 addition. Then the following lemmas can be proven.

Lemma B.2:

$$pr\left[\left(\sum_{i=1}^{n} S^i\right) \cup U\right] = \begin{cases} pr\left(\sum_{i=1}^{n}(S^i \cup U)\right) & \text{for odd n;} \\ pr(U) + pr\left(\sum_{i=1}^{n}(S^i \cup U)\right) & \text{for even n.} \end{cases}$$

Proof: Assume $U = \emptyset$. Clearly, the function $\sum_{i=1}^{n} S^i = 0$ can be written as a product of 2^{n-1} boolean sums (maxterms) of n variables, i.e.,

$$\left[\sum_{i=1}^{n} S^i = 0\right] = [M_0 M_1 \cdots M_{2^{n-1}-1} = 0], \tag{B.10}$$

where M_i denotes a maxterm. If $U \neq \emptyset$, the expansion for $[(\sum_{i=1}^{n} S^i) \cup U] = 0$ in terms of a set of different maxterms M_i' or in terms of the original maxterms M_i simply becomes

$$\begin{aligned} \left[\left[\left(\sum_{i=1}^{n} S^i\right) \cup U\right] = 0\right] &= [M_0' M_1' \cdots M_{2^{n-1}-1}' = 0] \\ &= [(M_0 \cup U)(M_1 \cup U) \cdots (M_{2^{n-1}-1} \cup U) = 0]. \end{aligned} \tag{B.11}$$

By definition, the maxterms in eq. (B.11) form mutually exclusive events. Therefore, $pr\left[\left(\sum_{i=1}^{n} S^i\right) \cup U = 1\right]$ can be written as:

$$\begin{aligned} pr\left[\left(\sum_{i=1}^{n} S^i\right) \cup U\right] &= 1 - pr\left[\left(\sum_{i=1}^{n} S^i\right) \cup U = 0\right] \\ &= 1 - \sum_{i=0}^{2^{n-1}-1} pr(M_i' = 0) \\ &= 1 - \sum_{i=0}^{2^{n-1}-1} pr[(M_i \cup U) = 0]. \end{aligned} \tag{B.12}$$

When n is odd, the product of sums expansion for $(\sum_{i=1}^{n} S^i) \cup U = 0$, does not contain the term $(U \cup \overline{S}^1 \cup \overline{S}^2 \cup \cdots \cup \overline{S}^n)$. That is, for n odd, all the maxterms that are part of the expansion have at least one uncomplemented variable. Hence, since all the terms contain at least one uncomplemented variable, a new variable $\underline{S}^i = S^i \cup U$ can replace one or any number of the original variables S^i, and the expressions for $pr(M_i' = 0) = pr[(M_i \cup U) = 0]$, in terms of the variables $\underline{S}^i$ remain the same as those for $pr(M_i = 0)$. Note that changing only one variable is equivalent to changing all of them because of the associative property of the sum operation ($\cup$). For example, in the case of two variables S^i and S^j, transforming only S^i is equivalent to transforming both S^i and S^j because

$$\begin{aligned} \underline{S}^i \cup S^j &= (U \cup S^i) \cup S^j \\ &= (U \cup S^i) \cup (U \cup S^j) \\ &= \underline{S}^i \cup \underline{S}^j. \end{aligned} \tag{B.13}$$

Therefore, the final expression for $pr[(\sum_{i=1}^{n} S^i) \cup U]$ in the case where n is odd is:

$$\begin{aligned} pr\left[\left(\sum_{i=1}^{n} S^i\right) \cup U\right] &= pr\left(\sum_{i=1}^{n} \underline{S}^i\right) \\ &= pr\left(\sum_{i=1}^{n} (S^i \cup U)\right). \end{aligned} \tag{B.14}$$

For n even, the product of sums expansion for $[(\sum_{i=1}^{n} S^i) \cup U] = 0$, contains the term $M_0' = (U \cup \overline{S}^1 \cup \overline{S}^2 \cup \cdots \cup \overline{S}^n)$. All other maxterms, M_i', contain at least one

uncomplemented variable other than U. Hence, as in the case where n is odd, the expressions for $pr(M_i' = 0)$ for $i = 0$, in terms of the variables $\underline{S}^i = S^i \cup U$, remain the same as those for $pr(M_i = 0)$ in terms of the original variables S^i. However, for the term $M_0' = (U \cup \overline{S}^1 \cup \overline{S^2} \cup \cdots \cup \overline{S^n})$, which, apart from U, contains only complemented variables, the expression for $pr(M_0' = 0)$ in terms of the transformed variables $\underline{S}^i$ is the same as that for $pr(M_0 = 0)$ in terms of the original variables S^i except for an additional term. Let $A = \overline{S}^1 \cup \overline{S}^2 \cup \overline{S}^3 + \cdots + \overline{S}^n$. Therefore, $M_0' = U \cup A$, and $pr(M_0' = 0) = pr[(U \cup A) = 0]$ Using Lemma 4.2,

$$pr(U \cup A = 0) = 1 - [1 - pr(U \cup A = 1) + pr(U = 1)]. \tag{B.15}$$

Therefore,

$$pr(U \cup A = 0) = pr(U \cup A = 1) - pr(U = 1).$$

Hence, the expression for $pr(M_0' = 0)$ in terms of the variables $\underline{S}^i$ is the same as that for $pr(M_0 = 0)$ in terms of the variables S^i, except for the additional term $pr(U = 1)$. Consequently, for n even, the final expression for $pr[(\sum_{i=1}^{n} S^i) \cup U]$ is:

$$\begin{aligned} pr\Big[\Big(\sum_{i=1}^{n} S^i\Big) \cup U\Big] &= pr(U) + pr\Big(\sum_{i=1}^{n} \underline{S}^i\Big) \\ &= pr(U) + pr\Big(\sum_{i=1}^{n} (S^i \cup U)\Big). \end{aligned} \tag{B.16}$$

■

Lemma B.3:

$$\begin{aligned} pr\Big(\sum_{i=1}^{n} S^i\Big) = {} & 2^{n-1} pr(S^1 \cup S^2 \cup \cdots \cup S^n) \\ & + \cdots + 4(-1)^{n-3} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{n} pr(S^i \cup S^j \cup S^k) \\ & + 2(-1)^{n-2} \sum_{\substack{i,j=1 \\ i<j}}^{n} pr(S^i \cup S^j) \\ & + (-1)^{n-1} \sum_{i=1}^{n} pr(S^i). \end{aligned}$$

Proof: The proof is by induction. Clearly, the theorem holds for $n = 2$, i.e., it results in Lemma B.1. Assume that the theorem is true for $n = l$; it will be shown that it is also true for $n = l + 1$.

It is assumed that

$$\begin{aligned} pr\Big(\sum_{i=1}^{l} S^i\Big) &= 2^{l-1} pr(S^1 \cup S^2 \cup \cdots \cup S^l) \\ &\quad + \cdots + 4(-1)^{l-3} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{l} pr(S^i \cup S^j \cup S^k) \\ &\quad + 2(-1)^{l-2} \sum_{\substack{i,j=1 \\ i<j}}^{l} pr(S^i \cup S^j) \\ &\quad + (-1)^{l-1} \sum_{i=1}^{l} pr(S^i) \end{aligned} \tag{B.17}$$

is true, and the objective is to show that

$$\begin{aligned} pr\Big(\sum_{i=1}^{l+1} S^i\Big) &= 2^{l} pr(S^1 \cup S^2 \cup \cdots \cup S^{l+1}) \\ &\quad + \cdots + 4(-1)^{l-2} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{l+1} pr(S^i \cup S^j \cup S^k) \\ &\quad + 2(-1)^{l-1} \sum_{\substack{i,j=1 \\ i<j}}^{l+1} pr(S^i \cup S^j) \\ &\quad + (-1)^{l} \sum_{i=1}^{l+1} pr(S^i). \end{aligned} \tag{B.18}$$

Using Lemma B.1, the following can be written:

$$\begin{aligned} pr\Big(\sum_{i=1}^{l+1} S^i\Big) &= pr\Big[\Big(\sum_{i=1}^{l} S^i\Big) \oplus S^{l+1}\Big] \\ &= 2pr\Big[\Big(\sum_{i=1}^{l} S^i\Big) \cup S^{l+1}\Big] - pr\Big(\sum_{i=1}^{l} S^i\Big) - pr(S^{l+1}). \end{aligned} \tag{B.19}$$

From Lemma B.2. eq. (B.19) can be rewritten as:

$$pr\Big(\sum_{i=1}^{l+1} S^i\Big) = \begin{cases} 2pr[\sum_{i=1}^{l}(S^i \cup S^{l+1})] - pr(\sum_{i=1}^{i} S^i) - pr(S^{l+1}) & \text{odd } l; \\ 2\Big(pr(S^{l+1}) + pr[\sum_{i=1}^{l}(S^i \cup S^{i-1})]\Big) - pr(\sum_{i=1}^{l} S^i) - pr(S^{l+1}) & \text{even } l. \end{cases} \tag{B.20}$$

Since only the sign of the term $pr(S^{l+1})$ differs for the two cases, eq. (B.20) can be rewritten as:

$$pr\Big(\sum_{i=1}^{l+1} S^i\Big) = 2pr\Big(\sum_{i=1}^{l}(S^i \cup S^{l+1})\Big) - pr\Big(\sum_{i=1}^{l} S^i\Big) + (-1)^l pr(S^{l+1}). \tag{B.21}$$

Substituting eq. (B.17) for $pr(\sum_{i=1}^{l} S^i)$ into eq. (B.21) yields

$$\begin{aligned} pr\Big(\sum_{i=1}^{l+1} S^i\Big) = {} & 2pr\Big[\sum_{i=1}^{l}(S^i \cup S^{l+1})\Big] \\ & - \Big[2^{l-1} pr(S^1 \cup S^2 \cup \cdots \cup S^l) \\ & \quad + \cdots + 4(-1)^{l-3} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{l} pr(S^i \cup S^j \cup S^k) \\ & \quad + 2(-1)^{l-2} \sum_{\substack{i,j=1 \\ i<j}}^{l} pr(S^i \cup S^j) \\ & \quad + (-1)^{l-1} \sum_{i=1}^{l} pr(S^i)\Big] \\ & + (-1)^l pr(S^{l+1}). \end{aligned} \tag{B.22}$$

But

$$
\begin{aligned}
pr\Big[\sum_{i=1}^{l}(S^i \cup S^{l+1})\Big] = 2^{l-1} pr\Big((S^1 \cup S^{l+1}) \cup (S^2 \cup S^{l+1}) \cup \cdots \cup (S^l \cup S^{l+1})\Big) \\
+ \cdots + \\
+ 4(-1)^{l-3} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{l} pr\Big((S^i \cup S^{l+1}) \cup (S^j \cup S^{l+1}) \cup (S^k \cup S^{l+1})\Big) \\
+ 2(-1)^{l-2} \sum_{\substack{i,j=1 \\ i<j}}^{l} pr\Big((S^i \cup S^{l+1}) \cup (S^j \cup S^{l+1})\Big) \\
+ (-1)^{l-1} \sum_{i=1}^{l} pr\Big((S^i \cup S^{l+1})\Big).
\end{aligned}
\tag{B.23}
$$

and eq. (B.22) becomes

$$
\begin{aligned}
pr\Big(\sum_{i=1}^{l+1} S^i\Big) = 2\Big[&2^{l-1} pr\Big((S^1 \cup S^{l+1}) \cup (S^2 \cup S^{l+1}) \cup \cdots \cup (S^l \cup S^{l+1})\Big) \\
&+ \cdots + \\
&+ 4(-1)^{l-3} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{l} pr\Big((S^i \cup S^{l+1}) \cup (S^j \cup S^{l+1}) \cup (S^k \cup S^{l+1})\Big) \\
&+ 2(-1)^{l-2} \sum_{\substack{i,j=1 \\ i<j}}^{l} pr\Big((S^i \cup S^{l+1}) \cup (S^j \cup S^{l+1})\Big) \\
&+ (-1)^{l-1} \sum_{i=1}^{l} pr\Big((S^i \cup S^{l+1})\Big)\Big] \\
- \Big[&2^{l-1} pr(S^1 \cup S^2 \cup \cdots \cup S^l) \\
&+ \cdots + 4(-1)^{l-3} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{l} pr(S^i \cup S^j \cup S^k) \\
&+ 2(-1)^{l-2} \sum_{\substack{i,j=1 \\ i<j}}^{l} pr(S^i \cup S^j) \\
&+ (-1)^{l-1} \sum_{i=1}^{l} pr(S^i)\Big] \\
+ (-1)^l &pr(S^{l+1}).
\end{aligned}
\tag{B.24}
$$

Eq. (B.24) can be rewritten as:

$$
\begin{aligned}
pr\Big(\sum_{i=1}^{l+1} S^i\Big) = 2^l pr(S^1 \cup S^2 \cup \cdots \cup S^l \cup S^{l+1}) \\
+ \cdots + 8(-1)^{l-3} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{l} pr(S^i \cup S^j \cup S^k \cup S^{l+1}) \\
+ 4(-1)^{l-2} \sum_{\substack{i,j=1 \\ i<j}}^{l} pr(S^i \cup S^j \cup S^{l+1}) \\
+ 2(-1)^{l-1} \sum_{i=1}^{l} pr(S^i \cup S^{l+1}) \\
+ (-1)^l pr(S^{l+1}) \\
- 2^{l-1} pr(S^1 \cup S^2 \cup \cdots \cup S^l) \\
+ \cdots + 4(-1)^{l-2} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{l} pr(S^i \cup S^j \cup S^k) \\
+ 2(-1)^{l-1} \sum_{\substack{i,j=1 \\ i<j}}^{l} pr(S^i \cup S^j) \\
+ (-1)^l \sum_{i=1}^{l} pr(S^i).
\end{aligned}
\tag{B.25}
$$

or

$$
\begin{aligned}
pr\Big(\sum_{i=1}^{l+1} S^i\Big) = 2^l pr(S^1 \cup S^2 \cup \cdots \cup S^l \cup S^{l+1}) \\
+ \cdots + 4(-1)^{l-2} \sum_{\substack{i,j,k=1 \\ i<j<k}}^{l+1} pr(S^i \cup S^j \cup S^k) \\
+ 2(-1)^{l-1} \sum_{\substack{i,j=1 \\ i<j}}^{l+1} pr(S^i \cup S^j) \\
+ (-1)^l \sum_{i=1}^{l+1} pr(S^i).
\end{aligned}
\tag{B.26}
$$

Clearly, eq. (B.26) is equal to eq. (B.18), which proves the lemma. ∎

Given Lemmas B.2. and B.3, the proof of Theorem 4.5 is rather straightforward.

Theorem 4.5:

$$
\begin{aligned}
pr(X_i^x \cup U_i^u = 1) = \Big((|x|+1) \ mod \ 2\Big) P_i^u \\
+ (-1)^{|x|-1} \sum_{s=1}^{|x|} (-2)^{s-1} \sum_t P_i^t,
\end{aligned}
$$

where $|x|$ denotes the number of ones in the binary representation of the index x, and where the indices t correspond to all the possible bitwise logical OR of the index u with the indices of s other components of X_i^x.

Proof: The desired $pr(X_i^x \cup U_i^u = 1)$ is a term which is essentially identical to the one in the statement of Lemma B.2. i.e., the probability of a union of a modulo-2 sum with an additional set or function. Here, the variable U in Lemma B.2 is replaced by U_i^u, and the more compact term X_i^x replaces the term $\sum_{i=1}^{n} S^i$ in Lemma B 2. Lemma B.2 states that the expression for this probability is the same as that of only the probability of the modulo-2 sum with a new set of variables. That is, here the same expression results when replacing all the original variables S_i^j by the variables $S_i^j \cup U_i^u$. The only difference in the expression arises when $|x|$ (n in Lemma B.2) is even. For $|x|$ even, the additional term $pr(U_i^u = 1) = P_i^u$ must be added to the original expression. In the

statement of Theorem 4.5, the factor $\Big((|x|+1) \ mod \ 2\Big)$ takes care of the appearance of this additional term when $|x|$ is even, and its disappearance when $|x|$ is odd. The other term, i.e.,

$$(-1)^{|x|-1} \sum_{s=1}^{|x|} (-2)^{s-1} \sum_{t} P_i^t,$$

in the statement of Theorem 4.5, is just another way of writing the statement of Lemma B.3. ∎

Appendix C. Proof of Theorem 4.9

Theorem 4.9: For an LFSR of size m with $g = |G|$ feedback taps, the time complexity of the iterative algorithm, for a sequence of length n, is upper bounded by $O(ng\ 2^m(3\ 2)^g) = O(ng2^{m+\ 585g})$.

Proof: The complexity of the algorithm is dominated by the computation of the term $pr(X_i^x \cup U_i^u = 1)$ where $x = G \wedge u'$, for $2^{m-1} \leq a \leq 2^m - 1$, $u = 2a - 2^m$, and $x = G \wedge u'$. If $|x| = k$, from Theorem 4.5, an upper bound on the computation of the term $pr(X_i^x \cup U_i^u = 1)$ is $O(k\ 2^k)$. This is because there are 2^k indices to be computed and the same number of terms to be summed, and an upper bound on the time complexity for computing each of the 2^k indices is $O(k)$. Let C_k denote the time to compute the term $pr(X_i^x \cup U_i^u = 1)$ when $|x| = k$. Therefore,

$$C_k \leq O(k\ 2^k). \tag{C.1}$$

Thus, the complexity of the computation of the term $pr(X_i^x \cup U_i^u = 1)$ is a function of $|x|$. Consequently, to establish the complexity of the algorithm, it is required to find the distribution of $|x|$ for a fixed G (i.e., number of cases where $|x| = 1, 2, 3,$ etc.), with a varying from 2^{m-1} to $2^m - 1$.

Consider the binary representations of the indices u for $2^{m-1} \leq a \leq 2^m - 1$. These are all the following combinations:

000...0000010

000...0000100

⋮

111...1111110

Hence, the binary representations of u' are:

111...111101

111...111011

⋮

000...000001

Therefore, u' takes on all the odd values less than $2^m - 1$. Throughout this dissertation, it is assumed that the last stage of the LFSR always feeds back to the first stage. Hence, the binary representation of G always has a one in the position of the least significant bit, i.e., the binary representation of all the possible feedback configurations is $XXX\ldots XXX1$, where X can be either one or zero.

From the above observations, for a given G, the distribution of $|x|$ as a function of g (and hence G) can easily be determined. Assume $g = 1$, which implies that $G = 000\ldots01$. Then, for all a such that $2^{m-1} \leq a \leq 2^m - 1$, $x = G \wedge u' = 000 \quad 01 \quad G$ Therefore $|x| = 1$ in all cases. Consequently, for $g = 1$, the complexity of the algorithm is $O(2^{m-1}C_1)$ for which an upper bound is: $O(2^{m-1}) = O(2^m)$

Assume $g = 2$, which implies that G has a one as its least significant bit and a one in one other bit position. In turn, this implies that for the range of possible u''s, the value of $|x|$ can be either 1 or 2. Without loss of generality, assume that $G = 000\ldots011$. Then, from the possible values of u', $|x| = 2$ arises for all cases where $u' = XXX\ldots X11$. Since there are 2^{m-2} such cases, in the given range of a, $|x| \quad 2$ in 2^{m-2} occasions. The cases where $|x| = 1$ arise when $u' = XXX\ldots X01$ Since there are also 2^{m-2} such cases, the number of times that $|x| = 1$ is also 2^{m-2}. Hence, for $g = 2$, the complexity of the algorithm is $O(2^{m-2}(C_1 + C_2))$.

Assume $g = 3$. Without loss of generality, assume that $G = 000\ldots0111$. In this case, $|x| = 3$ arises in 2^{m-3} situations $|x| = 2$ arises when $u' = XXX\ldots X011$ or $u' = XXX\ldots X101$. There are therefore $\binom{2}{1} \times 2^{m-3}$ cases where $|x| = 2$ The cases where $|x| = 1$ arise when $u' = XXX\ldots X001$, for which there are 2^{m-3} possibilities. Hence, the complexity of the algorithm when $q = 3$ is $O(2^{m-3}C_3 + \binom{2}{1}2^{m-3}C_2 + 2^{m-3}C_1)$.

Let T_g denote the time complexity of the algorithm as a function of q By induction, the general expression for T_g can be shown to be:

$$T_g = O(\binom{g-1}{g-1}2^{m-g}C_g + \binom{g-1}{g-1}2^{m-g}C_{g-1} + \binom{g-2}{g-3}2^{m-q}C_{g-2} + \cdots + \binom{g-1}{0}2^{m-g}C_1 \quad (C.2)$$

which is can be rewritten as

$$T_g = O\Big(\sum_{i=0}^{g-1} \binom{g-1}{g-1-i} 2^{m-g} C_i\Big) \tag{C.3}$$

Substituting the upper bounds for C_i (eq. (C.1)) in eq. (C.3) yields

$$T_g \leq O\Big(\sum_{i=0}^{g-1} \binom{g-1}{g-1-i} 2^{m-g} i 2^i\Big). \tag{C.4}$$

In turn, an upper bound on the above expression is:

$$T_g \leq O(g 2^{m-g} \sum_{i=0}^{g-1} \binom{g-1}{g-1-i} 2^i. \tag{C.5}$$

Eq. (C.5) can be rewritten as:

$$T_g \leq O\Big(g 2^{m-g} 2^{g-1} \sum_{i=0}^{g-1} \binom{g-1}{g-1-i} (1/2)^{g-i-1}\Big), \tag{C.6}$$

which simplifies to the following by letting $j = g-1-i$:

$$T_g \leq O\Big(g 2^{m-g} 2^{g-1} \sum_{j=0}^{g-1} \binom{g-1}{j} (1/2)^j\Big). \tag{C.7}$$

By the binomial theorem, for $|x| < 1$,

$$(1+x)^q = \sum_{i=0}^{q} \binom{q}{i} x^q. \tag{C.8}$$

Substituting eq. (C.8) into eq. (C.7) yields

$$T_g \leq O\Big(g 2^{m-g} 2^{g-1} (1 + 1/2)^{g-1}\Big) \tag{C.9}$$

which can easily be simplified to

$$T_g \leq O(ng 2^m (3/2)^g). \tag{C.10}$$

Since $3 = 2^{1.585}$, eq. (C.10) can be rewritten as:

$$T_g \leq O(ng 2^{m+.585g}). \tag{C.11}$$

■