This is to certify that the

dissertation entitled

DEVELOPMENT OF
EFFICIENT FAULT DIAGNOSABLE DESIGN
METHODOLOGIES FOR
ANALOG/MIXED-SIGNAL INTEGRATED CIRCUITS

presented by

Wei-hsing Huang

has been accepted towards fulfillment
of the requirements for

Ph.D. degree in Electrical Engineering

[illegible]
Major professor

Date 9/25/1998

DEVELOPMENT OF EFFICIENT FAULT DIAGNOSABLE DESIGN METHODOLOGIES FOR ANALOG/MIXED-SIGNAL INTEGRATED CIRCUITS

By

Wei-hsing Huang

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical Engineering

1998

ABSTRCACT

DEVELOPMENT OF EFFICIENT FAULT DIAGNOSABLE DESIGN METHODOLOGIES FOR ANALOG/MIXED-SIGNAL INTEGRATED CIRCUITS

By

Wei-hsing Huang

More and more mixed-signal devices are being designed recently for the applications of multimedia, wireless communication, and portable data systems. The analog circuit technology conventionally employed for such applications has been gradually switched to analog/digital mixed-signal circuit technology. Even though much more complicated digital circuits have been widely used in the DSP-based mixed-signal IC, analog circuits will remain for processing or interfacing analog signals. Integrating both digital and analog on a single chip has improved performance and reduced board size and cost. However, the increasing complexity of mixed-signal circuits drastically reduces the controllability and observability of the circuit on the chip. As a result, testing and fault diagnosis of such complex circuits becomes very difficult and expensive. Therefore, the goal of the thesis study is to develop efficient diagnosable design methodologies for analog/ mixed-signal integrated circuits and further integrate the methodologies for the development of an automated diagnostic test system.

This study develops a framework of the automated diagnostic test system for analog circuits, including two major analysis tools *Diagnosability Analysis* and *Redesignability*

Analysis, and *Diagnosable Design Methodologies*.

The diagnosability analysis estimates the maximum diagnosability a circuit can achieve and locates sections of a circuit having poor diagnosability for a given set of test points, while redesignability analysis derives the input/output (I/O) relations of the portions having poor diagnosability and reconstructs them for diagnosability enhancement. The developed diagnosable design methodologies include two major processes: Test Points Selection and Diagnostic Test Programs Generation. The former process identifies the number of test points required to achieve the desired diagnosability and their locations, while the latter process automatically generates test programs in analog version hardware description language, HDL-A.

A system, namely ADTS (Automated Diagnostic Test System), is also developed to realize the building blocks of the developed framework. A Graphic User Interface (GUI) environment is developed to allow user to input the circuit description and to conduct the desired analysis or analyses. The system is written in JAVA, an object-oriented language, with a data structure that implements spare matrix techniques for efficient memory usage.

To my parents and sister.

ACKNOWLEDGEMENTS

I sincerely thank Professor Chin-Long Wey, my thesis advisor, for his support and valuable suggestions throughout this research. It has been a pleasure to work under his supervision and I have certainly learned a lot from him. I would also like to express my gratitude to the members of my guidance committee, Dr. James A. Resh, Dr. Gregory M. Wierzba, Dr. Diane T. Rover and Dr. Chester Tsai, for taking time to serve in the committee. This research is sponsored in part by National Science Foundation under grant number MIP-9321255.

I have to thank my fellow graduate students, Dr. Cheng-Ping Wang and Jin-Sheng Wang. I got much inspiration from the discussion with them. I will also thank Mohammad Athat Khalil for his help of editing and suggestions for this dissertation.

A special thanks goes to my parents and my sister for the support and encouragement they have given me during the critical stage of my life.

A final note of appreciation must be sent to all the nice people who have ever enriched my campus life while I stayed at Michigan State University.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

More and more mixed-signal devices are being designed recently for the applications of multimedia, wireless communication, and portable data systems. The analog circuit technology conventionally employed for such applications has been gradually switched to analog/digital mixed-signal circuit technology. Integrating both digital and analog on a single chip has improved performance and reduced board size and cost. Even though much more complicated digital signal processing circuits have been widely used, analog circuits will remain for processing or interfacing analog signals [1]. For mixed-signal circuit testing today, the digital and analog components are tested separately. Procedure and equipment for testing stand-alone digital or analog chips have been well-established and implemented. However, manufacturers have found the costs associated with high-volume production of mixed-signal integrated circuits (ICs) are strongly affected by the cost of testing, where the analog circuit testing dominates [2,3]. Thus, developing efficient yet effective testing process for analog/mixed-signal circuits becomes a critical issue [4].

Why are analog/mixed-signal circuits so difficult to test? In general, a designer constructs the topological structure of a circuit and then selects a set of nominal parameters and the associated parameters tolerances to meet the design specifications. Due to the random fluctuations in fabrication process and variations in the circuit operating conditions, the parameters and their tolerances are selected in such a way that the designed circuit has enough margins to pass the acceptability criteria. Statistical design approaches [5-7] are usually employed to select the nominal parameters and their tolerances for enhancing manufacturability and maximizing chip yields. As a result, the above design process guarantees: a designed circuit meets the design specifications if all parameter values are selected within their tolerances. However, this process does not guarantee: a designed circuit fails to meet the design specifications when a parameter deviation is far beyond its tolerance [4,8]. Therefore, analog circuits must be tested by their specifications instead of their parameters. Since fault behaviors may not be reflected properly even with performance deviation, what the practical fault models for analog circuits are becomes a controversial issue. Without such practical fault models, fault simulations, test generation, and fault coverage evaluation become meaningless.

Testing and *fault diagnosis* are two important aspects in the design and maintenance of electronic circuits. Testing is to detect fault(s) in a circuit, while fault diagnosis is to detect and locate the fault(s) [9]. If a circuit has been found to be faulty during design characterization before it is in high volume production, it may be useful to diagnose the causes of the failure. Once faults are identified and located, a circuit can then be re-designed to be less sensitive to common failure mechanisms [2]. Fault diagnosis is an inverse problem of the sensitivity analysis problem. More specifically, the sensitivity analysis problem is to find the performance deviations from a set of given parameter deviations, while the fault diagnosis problem is to find the parameter deviations from a set of given performance deviations. If a parameter deviation is out of its tolerance, the component associated with this parameter is then said to be faulty.

Fault diagnosis of analog circuits and systems has been recognized as an extremely difficult problem because of the *lack of failure data*, *continuum nature of analog circuits*, *component tolerance*, and *high nonlinearity of diagnosis equation* [9-11]. A number of fault diagnosis algorithms have been developed to resolve the problem [9-29]. Among them, an efficient self-testing approach fault diagnosis algorithm and an analog automatic test program generator (AATPG) were developed for both linear and nonlinear analog circuits [16-18].

To develop efficient fault diagnosable design methodologies for mixed-signal ICs, the following important issues must be addressed: (1) given a circuit topology, how to measure the circuit's diangosability for a given set of test points; and (2) given a circuit topology, how to select a set of test points that meets the desired diagnosability.

It is important for a designer to know what the maximum diagnosability a circuit can achieve and which portion of the circuit having poor diagnosability. The information allows estimation of a circuit's diagnosability before the fault diagnosis is attempted. Hence any potential problems can be located early in the design phase, allowing modifications to be introduced to improve the final diagnosability of the circuit.

For analog circuit design, a designer usually takes a relatively short period time generating a cicuit topology and spends the remaining design time adjusting design parameter values to meet the design specification. Usually , there are some portions having poor testability/diagnosability which the designer cannot discover beforehand. Once those portions are found, inserting extra test points may be needed for improving the observability and controllability of the circuit. However, inserting test points can cause the circuit performance to deviate from the design specification due to the loading effect. Therefore, the designer may need additional design time and efforts to re-adjust the parameter values again. Evidently, an efficient test selection process can greatly reduce the design time and efforts.

1.1 Objectives and Research Tasks

The objective of the thesis study is to develop efficient diagnosable design methodologies and automated diagnostic test system for analog circuits. The developed methodologies and system will then be extended for mixed-signal circuits [30]. Fig. 1-1 illustrates the framework of the development in this study. Based on the self-testing approach fault diagnosis algorithm developed in [16-18], the framework is comprised of two major analysis tools: Diagnosability Analysis and Redesignability Analysis, and the developed Diagnosable Design Methodologies.

The diagnosability analysis estimates the maximum diagnosability a circuit can achieve and locates sections of a circuit having poor diagnosability for a given set of test points [31], while redesignability analysis derives the input/output (I/O) relations of the portions having poor diagnosability and reconstructs them for diagnosability enhancement [32]. In addition, at this momemnt, the developed diagnosable design methodologies include two major processes: Test Points Selection and Diagnostic Test Programs Generation. The former process identifies the number of test points required to achieve the desired diagnosability and their locations, while the latter process automatically generates test programs for fault diagnosis. For the mixed-signal circuit applications, hardware description languages (HDLs), such as HDL-A [33,34] for analog circuits and VHDL [35]

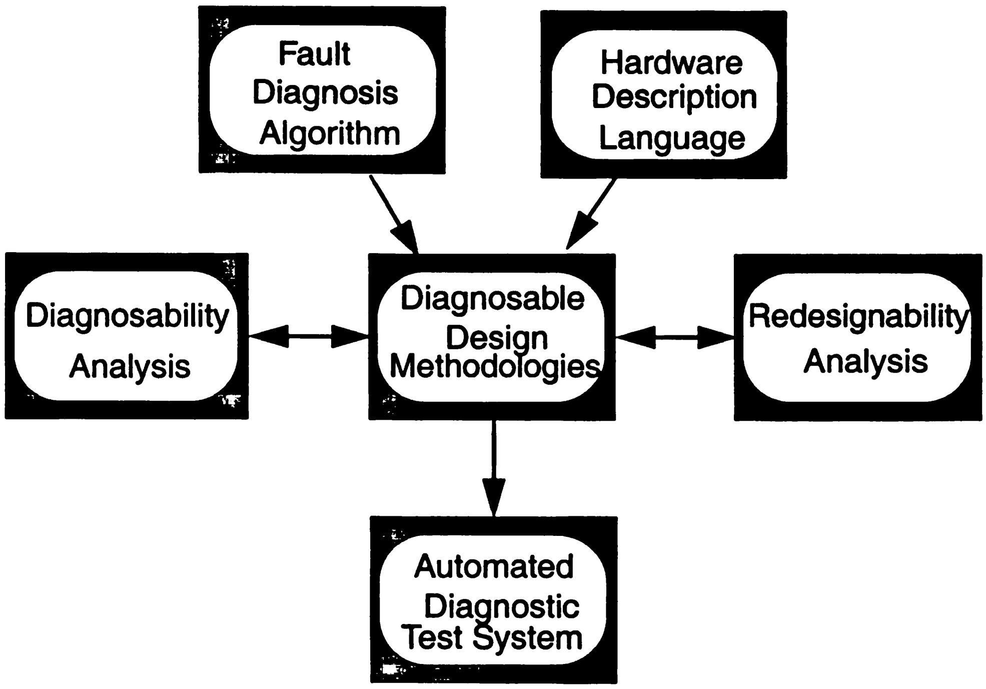

Fault Diagnosis Algorithm
Hardware Description Language
Diagnosability Analysis
Diagnosable Design Methodologies
Redesignability Analysis
Automated Diagnostic Test System

Fig. 1-1 Framework of an Automated Diagnostic Test System

for digital circuits, will be used to develop automatic test program generators.

With the successful development of hierarchical testability design system that define realistic fault models and generate test patterns for analog/mixed-signal circuits in [4], an automated diagnostic test system is also developed in this study. A software program, namely ADTS (Automated Diagnostic Test System) [36], as the system window shown in Fig. 1-2, is developed to realize the functional blocks in Fig. 1-1. A Graphic User Interface (GUI) environment, as shown in Fig. 1-3, allows user to input the circuit description and to select the analysis tools. The system contains two major processes: Circuit Description and Analysis Engine. The former process constructs the database for the analysis use, while the analysis engine performs the desired analysis/analyses. The system is written in JAVA, an object-oriented language, with a data structure that implements spare matrix techniques for efficient memory usage.

1.2 Thesis Organization

The dissertation is organized as follows: Chapter 2 reviews some of existing analog fault diagnosis techniques and the self-testing approach fault diagnosis algorithm.

Chapter 3 presents the diagnosability analysis process. Given a circuit topology and a set of test points, the upper-bound diagnosability of the circuit can be estimated. If the upper-bound diagnosability is less than the desired one, then redesign or adding extra testing points may be needed. Otherwise, the diagnosability analysis process will take place. Two major tasks are involved in this process: Component Assignment and Diagnosable Configuration Generation. The former generates the candidate diagnosable configurations based on the given circuit topology and test points information, while the latter selects the

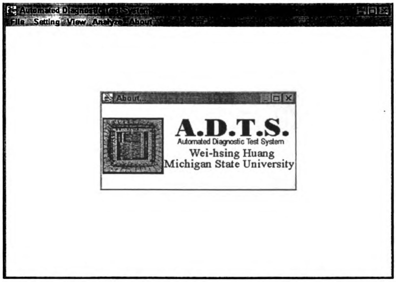
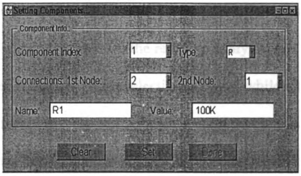

Automated Diagnostic Test System
File Setting View Analyze About
About...
A.D.T.S.
Automated Diagnostic Test System
Wei-hsing Huang
Michigan State University

Fig. 1-2 Automated Diagnostic Test System


Setting Components...

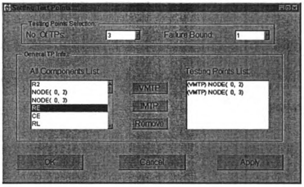
Component Info.:

Component Index: 1 Type: R

Connections: 1st Node: 2 2nd Node: 1

Name: R1 Value: 100K

(a)


Setting Test Points...

Testing Points Selection:

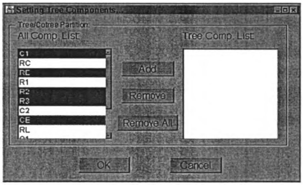
No. Of TPs: 3 Failure Bound: 1

General TP Info.:

All Components List:
R2
NODE(0, 2)
NODE(0, 3)
RE
CE
RL

Testing Points List:
(VMTP) NODE(0, 2)
(VMTP) NODE(0, 3)

(b)


Setting Tree Components...

Tree/Cotree Partition:

All Comp. List:
C1
RC
RE
R1
R2
R3
C2
CE
RL

Tree Comp. List:

(c)

Fig. 1-3. GUI Circuit Description.

one with the maximum diagnosability from the candidates. The process will also identify the portions having poor diagnosability.

Chapter 4 describes the redesignability analysis process which checks if the redesign is feasible. An efficient algorithm for the redesign analysis will be presented.

Chapter 5 presents the test points selection process, while Chapter 6 discusses the test program generation process. The test points selection problem is formulated as a minimum covering problem. Thus, efficient algorithms are developed to resolve the problem and to speed up the selection process. Chapter 6 describes the development of automatic test program generation process using HDL-A.

Finally, Chapter 7 summarizes the thesis, gives a concluding remark and some future research directions.

Chapter 2

BACKGROUND

With the ever-increasing complexity and compactness of analog/mixed-signal integrated circuits, analog testing and diagnosis has become more and more important. Conventionally, analog circuits are tested with *bed of nails*, i.e., every component on the unit under test (UUT) is probed and measured. However, with the progress of the modern packaging technology, as well as the increasing density of the number of components in a UUT, it is impractical to provide proportionally more I/O pins. As a result, the number of test points that can be externally accessible will be limited for modern circuit testing. Fault detection and location in electronic systems are performed with measurement done on these limited number of test points. Thus, an effective algorithm which utilizes these test points efficiently plays a key role in fault analysis.

Section 2.1 reviews some of existing analog fault diagnosis algorithms. Section 2.2 discusses salient features of the self-testing approach fault diagnosis algorithm developed in [16-18]. Section 2.3 presents the circuit modeling schemes using HDL-A. Finally, Section 2.4 addresses some issues related to the thesis research.

2.1 Existing Analog Fault Diagnosis Methodologies

Until recently, effort at producing algorithms for automatical diagnosis process has concerned mainly digital circuits for which more or less satisfactory solutions have been reached. Analog circuits, on the other hand, have received much less effort. The difficulties and problems are due to the following [37].

- Relations among input and output signals in analog circuits are somewhat intricate and not "exact". For instance, the truth tables of digital circuits will not work for analog circuits where only approximations allow modeling.
- Analog systems are frequently nonlinear. The noises and the values of parameters of the components exhibit large deviations. Consequently, the tolerance problem has to be considered even for a good circuit and deterministic approaches are most often inefficient.
- Fault categories as well as their statistical distributions and correlations are not known with precision. Thus, fault classification and modeling based on statistical result need more elaboration.
- Use of analog components is not as systematic as is the case for digital components.

To alleviate the above limitations, two approaches are often employed in analog fault analysis [11,37]: the simulation-before-test (SBT) approach and the simulation-after-test (SAT) approach. The SBT approach requires the simulation of different possible faults and storage of the results as a fault dictionary. The faulty subnetwork responses are compared with the dictionary entries and the closest entry to the responses determines the possible faults. The method is usually suitable for single catastrophic fault location. For multiple faults situation, the size of the dictionary becomes very large and the method is impractical. In the SAT approach, diagnosis algorithms are deployed so that the faulty network responses can be conducted to locate the faulty components. In both cases, there exists a trade-off between the computational effort and the number of accessible nodes.

Based on these two categories, a number of analog fault diagnosis techniques have been proposed [37]. They can be roughly categorized as the following three methods:

- *Estimation Methods*: These methods address detection, location, and identification. Two general classes of methods belong to this category: deterministic (or quasi-deterministic methods) and probabilistic methods. The former methods consist of determining from measurements as well as the actual values of the parameters of the system under test. In the latter methods, the distribution laws of measured responses are determined from the tolerances on the parameter values and their associated statistical distributions.
- *Topological Methods*: The basic data to be handled are the system's structure and, possibly, analytic relations between input variables and measured responses. Such methods apply to detection and location. These methods rely on some graphical analysis techniques.
- *Taxonomic Methods*: The system's reference responses corresponding to each potential fault condition are stored into fault dictionary. During actual testing, measurement results are compared to the responses recorded and the detected fault is the one for which the set of measurements differs the least. The accuracy of such methods is directly dependent on how comprehensive the fault dictionary is.

2.2 Self-Testing Approach Fault Diagnosis Algorithm

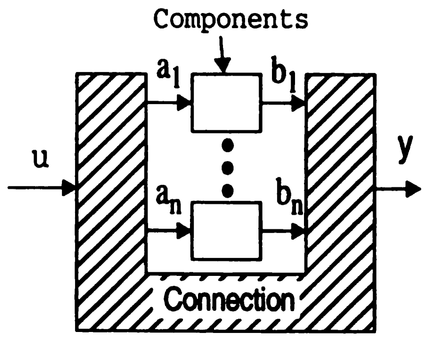
The self-testing approach fault diagnosis algorithm developed in [16-18] is a SAT approach with a single test vector. This sub-section describes the system model, the self-testing approach, and its implementation. In addition, the parallel processing capability [38] and diagnosability enhancement [39,40] are also discussed.

2.2.1 Component Connection Model

The fault diagnosis algorithm developed in [16,17] is based on an interconnection system model known as the *component connection model* (CCM), as shown in Fig. 2-1. Assume that the UUT is comprised of n components, k external test inputs, and m external test points. The UUT characterizes its components together with a connection equation as follows,

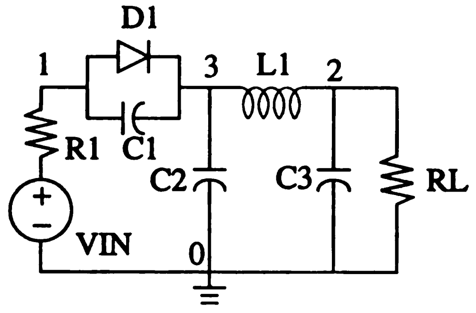
$$
\begin{aligned}
a &= L_{11}b + L_{12}u \\
y &= L_{21}b + L_{22}u
\end{aligned}
\qquad (2\text{-}1)
$$

where **a**=col(a_i) and **b**=col(b_i), i=1,2,..,n, are the column vectors of component input and output variables, respectively, and **u** and **y** are respectively the column vectors of external test inputs applied to the system and the system responses measured at the various test points. The connection matrix, L-matrix, is generally sparse and its entries are 1, 0, or -1. The entries of vectors a and b are either *node voltages* or *branch currents*. For analog linear circuits, the component equations is modeled in the frequency domain [16], while those in nonlinear circuits are modeled in the time domain [17]. Note that the components in the CCM model can be either *discrete components, individual chips*, or *subsystems*.



u: System Input
y: Output Response

Fig. 2-1 System Model



(a) Example Circuit

$$\begin{bmatrix} \text{Node 0} \\ \text{Node 1} \\ \text{Node 2} \\ \text{Node 3} \end{bmatrix} = \begin{array}{c} \begin{bmatrix} R1 & RL & C1 & C2 & C3 & L1 & D1 \end{bmatrix} \\ \begin{bmatrix} 1 & -1 & 0 & -1 & -1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & -1 \end{bmatrix} \end{array}$$

(b) Incidence Matrix

$$S_f = \begin{bmatrix} 1 & 0 & 0 & 0 & -1 & -1 & -1 \\ 0 & 1 & 0 & 1 & -1 & -1 & -1 \\ 0 & 0 & 1 & 0 & 0 & -1 & -1 \end{bmatrix} \qquad B_f = \begin{bmatrix} 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

(c) Fundamental Cut-set Matrix S_f and Fundamental Loop Matrix Bf

Fig. 2-2 Connection Matrixes

Thus, the system model can be applied for analog, digital, and mixed-signal integrated circuits and systems.

To construct the connection matrix, given a circuit schematic diagram, as shown in Fig. 2-2(a), an incidence matrix A, as shown in Fig. 2-2(b), describes the interconnections between each component, where a directional notation is used where branch current flow out of node labeled as "-1" toward node labeled as "1". By performing Gaussian elimination process on matrix A, the fundamental cut-set matrix S_f can be obtained. S_f describes the relations between the current flow through each components and the nodes it is connected. It satisfies the Kirchoff's current law (KCL) and has the form of $S_f = [I_r \mid D]$, as shown in Fig. 2-2(c), where D is a r x (b-r) matrix and I_r is a r x r identity matrix. Similarly, the fundamental loop matrix which describes Kirchoff's voltage law (KVL) can be obtained by simply transposing D matrix as $B_f = [-D^t \mid I_{n-r}]$, where D^t is the transpose matrix of D and I_{n-r} is a (n-r) x (n-r) identity matrix.

Let $I=\begin{bmatrix} I_t \\ I_c \end{bmatrix}$ and $V=\begin{bmatrix} V_t \\ V_c \end{bmatrix}$, where I_t and I_c are the currents in the tree and co-tree edges, respectively, and V_t and V_c are the voltages in the tree and co-tree edges, respectively. According to KCL and KVL, we obtain

$$0 = S_f I = I_t + D I_c; \text{ and } 0 = B_f V = -D^t V_t + V_c \qquad (2\text{-}2)$$

Therefore,

$$\begin{bmatrix} I_t \\ V_c \end{bmatrix} = \begin{bmatrix} 0 & -D \\ D^T & 0 \end{bmatrix} \begin{bmatrix} V_t \\ I_c \end{bmatrix} \qquad (2\text{-}3)$$

If we choose the component input vector $a=\begin{bmatrix} I_t \\ V_c \end{bmatrix}$ and the component output vector

$b=\begin{bmatrix} V_t \\ I_c \end{bmatrix}$, then we have the matrix $L_{11}=\begin{bmatrix} 0 & -D \\ D^T & 0 \end{bmatrix}$. If all test points are chosen from either I_t or V_c, then the matrix L_{21} can be constructed from the L_{11} matrix. Fig. 2-3 illustrates the connection matrix L for the example circuit in Fig. 2-2(a).

For linear circuits, the component equation is modeled in the frequency domain as follows,

$$b = Z\,a \qquad (2\text{-}4)$$

where $Z=\mathrm{diag}(Z_i)$, $i=1,2,..,n$, is a frequency domain composite component transfer matrix. Each $Z_i=Z_i(s,r)$ describes the i-th component of the UUT, where $r=\mathrm{col}(r_i)$ is the column vector of unknown component parameters and s is the complex frequency variable [16]. Typically, the unknown component parameters take the form of resistance, capacitance, inductances, and amplifier gain, etc.

For nonlinear circuits, the component characteristics are expressed as follows,

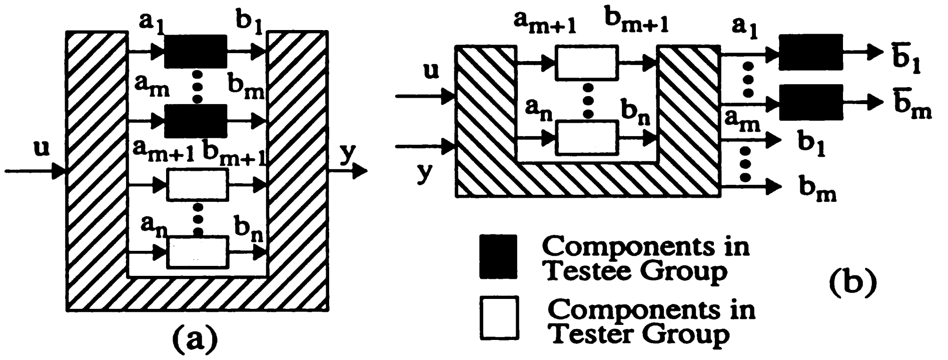
$$\dot{x}_i = F_i(x_i,a_i);\ b_i = G_i(x_i,a_i);\ x_i(0)=0;\ i=1,2,..,n \qquad (2\text{-}5)$$

where x_i's are the component state variables. The component equations in (2-5) are modeled in the time domain [17]. For notational purpose, we stack the individual component equations together to form the following composite component equations;

$$\dot{x} = F(x,a);\ b = G(x,a);\ x(0)=0; \qquad (2\text{-}6)$$

$$\begin{bmatrix} IR1 \\ VC1 \\ ID1 \\ VC2 \\ IL1 \\ VC3 \\ VRL \\ ID1 \\ IL1 \\ VRL \end{bmatrix} = \left[\begin{array}{ccccccc:c} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 1 & 1 & 0 \\ -1 & 0 & -1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ -1 & 0 & -1 & 0 & -1 & 0 & 0 & 1 \\ -1 & 0 & -1 & 0 & -1 & 0 & 0 & 1 \\ \hdashline 0 & -1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ -1 & 0 & -1 & 0 & -1 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} VR1 \\ IC1 \\ VD1 \\ IC2 \\ VL1 \\ IC3 \\ IRL \\ VIN \end{bmatrix}$$

Fig. 2-3 Connection Matrix of Example Circuit in Fig. 2-2(a)



$$\begin{bmatrix} ID1 \\ VC2 \\ IL1 \\ VRL \\ \hdashline IR1 \\ VC1 \\ VC3 \\ \hdashline VR1 \\ IC1 \\ IC3 \end{bmatrix} = \left[\begin{array}{cccc:cccc} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hdashline 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hdashline -1 & 0 & -1 & 0 & 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} VD1 \\ IC2 \\ VL1 \\ IRL \\ \hdashline u1 \\ y1 \\ y2 \\ y3 \end{bmatrix}$$

$$a^1 = \begin{bmatrix} ID1 \\ VC2 \\ IL1 \\ VRL \end{bmatrix} \quad b^1 = \begin{bmatrix} VD1 \\ IC2 \\ VL1 \\ IRL \end{bmatrix}$$

$$a^2 = \begin{bmatrix} IR1 \\ VC1 \\ VC3 \end{bmatrix} \quad b^2 = \begin{bmatrix} VR1 \\ IC1 \\ IC3 \end{bmatrix}$$

(c)

Fig. 2-4. Self-Testing Approach: (a) Subdivision; (b) Pseudo Circuit; and (c) Its Connection Matrix.

2.2.2 Self-Testing Approach

Conceptually, at each step of the diagnosis process, the components are subdivided into two groups, namely *Tester Group* and *Testee Group*. Assuming that the components in the *Testee Group*, i.e., components #(m+1) to #n, as shown in Fig. 2-4(a), are all good, (The assumption will be justified from the test results) and thus their component characteristics are known. From these known component characteristics and both known **u** and **y**, we estimate the function behaviors of the components in the *Tester Group*, i.e., components #1 to #m. For a given component subdivision, the circuit in Fig. 2-4(b), referred to as "*pseudo circuit*", is reconstructed from the original one in Fig. 2-4(a). Since we use the components in the *Tester Group* to test those in the *Testee Group*, hence it is "*self-testing*" approach. A number of component subdivisions may be needed and a decision algorithm is required to identify the faulty component(s).

Mathematically, at each step, the connection equations in (2-1) are partitioned as follows, where *Tester Group* and *Testee Group* are represented by superscripts "1" and "2", respectively.

$$a^1 = {L_{11}}^{11} b^1 + {L_{11}}^{12} b^2 + {L_{12}}^1 u \qquad (2\text{-}7a)$$
$$a^2 = {L_{11}}^{21} b^1 + {L_{11}}^{22} b^2 + {L_{12}}^2 u \qquad (2\text{-}7b)$$
$$y = {L_{21}}^1 b^1 + {L_{21}}^2 b^2 + L_{22} u \qquad (2\text{-}7c)$$

the component equations for linear case are also partitioned as follows,

$$b^1 = Z^1 a^1 \qquad (2\text{-}8a)$$
$$b^2 = Z^2 a^2 \qquad (2\text{-}8b)$$

and those for nonlinear case are

$$\dot{x}^1 = F^1(x^1,a^1);\ b^1 = G^1(x^1,a^1);\ x^1(0)=0 \qquad (2\text{-}9a)$$

$$\dot{x}^2 = F^2(x^2,a^2);\ b^2 = G^2(x^2,a^2);\ x^2(0)=0 \qquad (2\text{-}9b)$$

Therefore, a pseudo circuit can be derived and expressed as follows [16,17]

$$a^1 = K_{11}\, b^1 + K_{12}\, u^p \qquad (2\text{-}10a)$$

$$y^p = K_{21}\, b^1 + K_{22}\, u^p \qquad (2\text{-}10b)$$

Where u^p=col(u,y) and y^p=col(a^2,b^2) are the external input and output vectors of the pseudo circuit, respectively, and the K-matrix is the connection matrix.

$$\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} = \left[\begin{array}{c|cc} L^{11}_{11}-L^{12}_{11}(L^2_{21})^{-L}L^1_{21} & L^1_{12}-L^{12}_{11}(L^2_{21})^{-L}L_{22} & L^{12}_{11}(L^2_{21})^{-L} \\ \hline L^{21}_{11}-L^{22}_{11}(L^2_{21})^{-L}L^1_{21} & L^2_{12}-L^{22}_{11}(L^2_{21})^{-L}L_{22} & L^{22}_{11}(L^2_{21})^{-L} \\ -(L^2_{21})^{-L}L^1_{21} & -(L^2_{21})^{-L}L_{22} & (L^2_{21})^{-L} \end{array}\right] \qquad (2\text{-}11)$$

Consider a component subdivision of the circuit in Fig. 2-2(a), where the *Tester Group* contains D1, C2, L1, and RL, and the *Testee Group* includes R1, C1, and C3. The corresponding K-matrix is shown in Fig. 2-4(c). By (2-7), b^1 can be derived from the given u^p and the component characteristics of the *Tester Group*. Then, by (2-7b), both a^2 and b^2 are computed. Note that a_i^2 and b_i^2 are the input and output variables of the i-th component in the *Testee Group*. Therefore, the derived characteristic of each component in the *Testee Group* is compared with its nominal value to determine the component status.

Note that components in a circuit cannot be subdivided arbitrarily. In (2-11), it has been shown that the K-matrix exists if the matrix $L_{21}{}^2$ is left invertible. In other words, both a^2 and b^2 in (2-10) are solvable if $L_{21}{}^2$ is left invertible. For example, in the connection matrix of Fig. 2-3, the $L_{21}{}^2$ matrix with respect to R1, C1, and C3 is equal to

$$\begin{bmatrix} 0 & -1 & 1 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \end{bmatrix}$$

which is invertible, i.e., the K-matrix exists. Thus, the corresponding pseudo circuit can be constructed and simulated to derive the characteristics of components in the *Testee Group*.

Using the K matrix, the analog properties of y^p can be interpreted as linear combination of those of u^p. Diagnosis is performed by comparing responses in y^p with their nominal responses.

In the linear case [16], y^p can be represented as

$$y^p = M\,u^p \qquad (2\text{-}12)$$

where

$$M = K_{21}Z^1(I-K_{11}Z_1)^{-1}K_{12} + K_{22}$$

Specifically,

$$a^2 = M_{11}u + M_{12}y$$
$$b^2 = M_{21}u + M_{22}y$$

With

$$\bar{b}^2 = Z^2a^2 \qquad (2\text{-}13)$$

A component can be identified non-faulty if

$$(\bar{b}^2 - b^2) < \varepsilon, \qquad (2\text{-}14)$$

where ε is the predefined tolerance.

For the nonlinear circuits, component characteristics are represented by a set of decoupled state models, control sources are used to obtain the value of these state variables [17]. By (2-2), the i-th element of a^1 is the sum of the products of the i-th row of K_{11} and b^1, and the i-th row of K_{12} and u^p. Since the elements in a^1 and b^1 are either currents or voltages, they can be modeled by using "control sources." More specifically, assume that the vectors b^1 and u^p in the pseudo circuit are

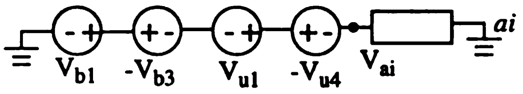
$$b^1 = \text{col}[Vb1, Ib2, Vb3, Ib4, Vb5];$$

$$u^p = \text{col}[Vu1, Vu2, Iu3, Vu4];$$

and the i-th element of a^1 is a voltage measurement, say V_{ai}. Let the i-th row of K_{11} and K_{12} be [1,0,-1,0,0] and [1,0,0,-1], respectively. Therefore, V_{ai} is expressed as

$$V_{ai} = Vb1 - Vb3 + Vu1 - Vu4 \qquad (2\text{-}15)$$

Here V_{ai} acts as a dependent voltage source controlled by the voltages across b1, b3, u1, and u4, i.e., V_{ai} is the voltage across the serial connection of these components shown below,

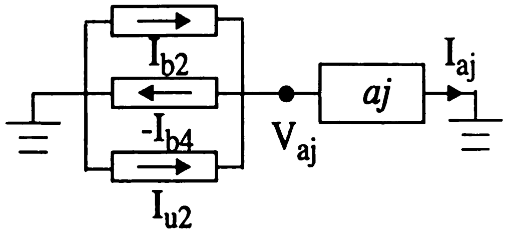


where four voltage controlled voltage sources (vcvs), Vb1, Vb3, Vu1, and Vu4 are used. Once the voltage V_{ai} is derived, the box is filled by the component "ai" to generate "I_{ai}" in b1.

Similarly, if the j-th element of a1 is a current measurement and

$$I_{aj} = Ib2 - Ib4 + Iu2 \qquad (2\text{-}16)$$

then I_{aj} is the sum of three current controlled current sources (cccs) which are connected in parallel as below,



2.2.3 Parallel Processing

In the self-testing approach, a number of component subdivisions are needed to identify the faulty components, where the corresponding pseudo circuits are simulated. Due to the independency of the component subdivisions, the corresponding pseudo circuits can be simulated simultaneously. Therefore, a number of parallel algorithms with various decision-making processes have been presented in [38] to speed-up the fault diagnosis process.

2.2.4 Diagnosability Enhancement

As mentioned, a pseudo circuit exists if the corresponding matrix $L_{21}{}^2$ has a full global column-rank [39]. For a full global column-rank, $L_{21}{}^2$ cannot have a zero column, nor two identical column. Since $L_{21}{}^2$ is a sub-matrix of L_{21} which is defined by the selected test points, the test points must be selected in such a way that the above cases can be avoided. A simple set of rules have been presented in [39, 40]. For parallel components, they have the same voltage across them and the total current that flows through them is known. Thus the measurement quantity has to be the current through all but one of the components to diagnose any fault, i.e., one of these parallel components has to be placed on the cotree edges and those whom are placed on tree edges must be chosen as test points. On the other hand, for series components, they have the same current through them and the total voltage across them is known. Thus the measurement quantity has to be the voltage across each but one of the series element to diagnose any fault, i.e., one of the series component has to be placed on the tree edges and those whom are placed on the cotree edges must be chosen as test points. Finally, to avoid L_{21}^2 with a zero column, the number of test points must be at least t+1 for a t-diagnosable circuit [39], where t is the maximum number of the allowable faults in the UUT.

Adding extra test points may enhance the circuit's diagnosability. However, adding extra test points also implies the need of extra pins in the IC which may be very costly in many cases. Therefore, Built-in Self-test (BIST) design methodologies may be practical alternatives for diagnosability enhancement [41-47].

2.2.5 Automatic Diagnosis System

Based on the self-testing approach fault diagnosis algorithms, an automatic diagnosis system was developed in [18], as illustrated in Fig. 2-5. The system can be divided into off-line and on-line components. The former, corresponding to the test system design stage, is used by test system designer to input nominal system specifications to generate a database which is used later by the on-line component. To implement the actual test, the field engineer invokes the on-line component input data describing the UUT; the assumed maximum number of simultaneous failures, the type of decision algorithm to be employed, and the source of the test data. The actual test can then be run in a fully automatic mode or interactively.

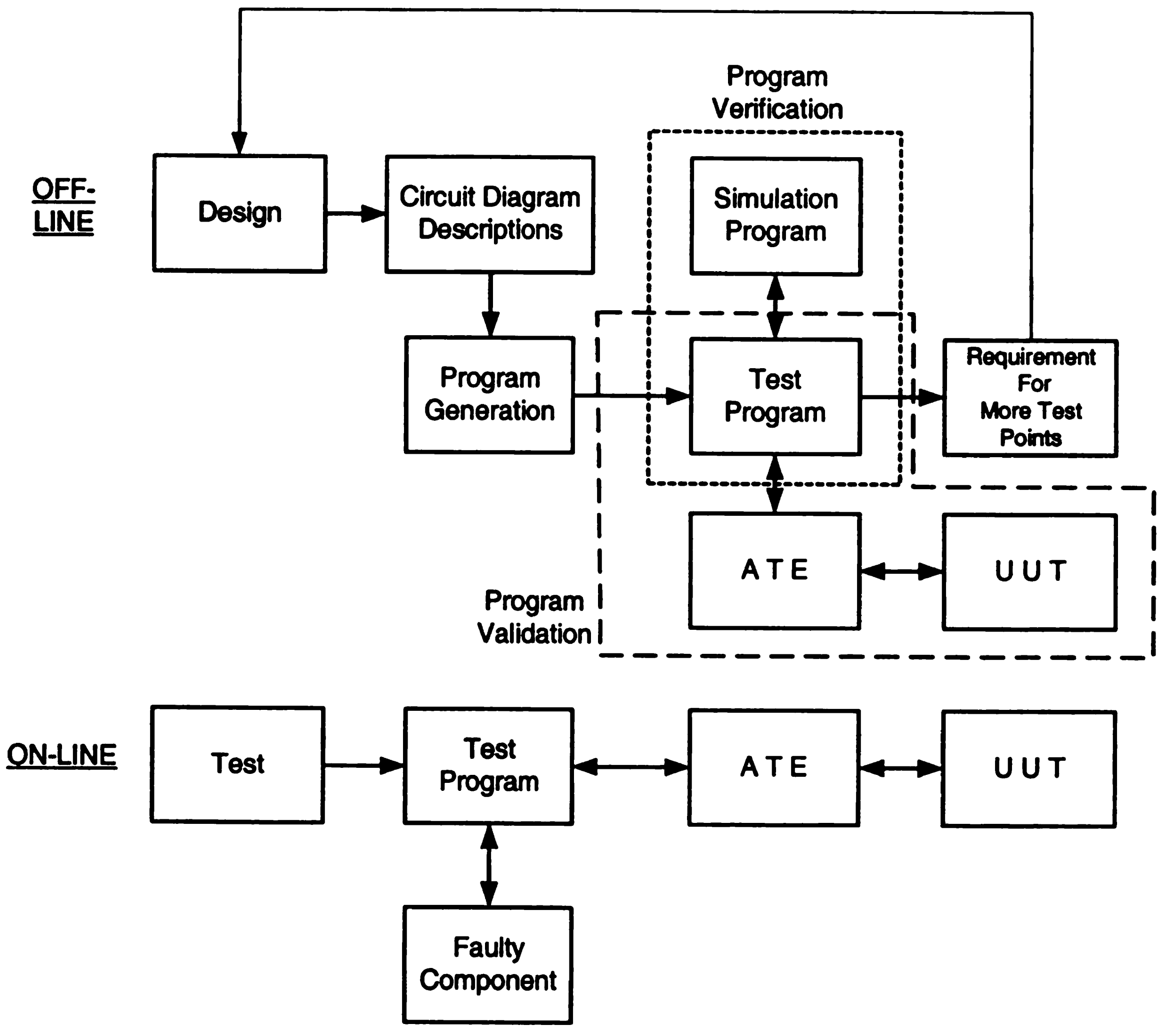


Fig. 2-5 Automatic Diagnosis System

A circuit description and test objectives, including circuit schematics, component values, accessible terminals and input frequency for linear circuits or time steps for nonlinear circuits, are given to the off-line component to generate the test programs. For test program verification, fault(s) are injected to detect the diagnosis ability of the test programs. At the validation stage, automatic test equipment (ATE) is used to generate test signals and store the measured test results. For linear circuits, these data are used to obtain test results via simple matrix multiplication of equations in (2-12). In the nonlinear case, the SPICE codes are used to evaluate via the on-line simulation of an appropriate pseudo circuit. Once the test programs are verified and validated, they can then be used for circuit fault diagnosis.

2.3 Analog Hardware Description Language (HDL-A)

In this section, the hardware description languages (HDLs), such as HDL-A and VHDL, are introduced and the software system, AATPG, are upgraded for analog circuits for analog/mixed-signal circuits. The HDL-A is primarily a language to be used to develop analog and mixed-signal models [48]. The language also allows the modeling of digital logic, either stand alone or embedded in analog models, using a subset of VHDL. HDL-A is primarily used for creating behavior models. Models written using the HDL-A are intended to interface to (called from) SPICE-like simulator, or alternatively VHDL simulator [49,50]. In electrical circuits, the fundamental principle is typically expressed as KCL and KVL. The connections of a model carry “across” and “through” quantities, i.e., voltage and current, rather than the single directional signal associated with a port in VHDL. The behavior of lumped analog devices is described by expressing the relations between voltages and currents at all connections of the device. Such relations can be described with linear, or non-linear, algebraic equations or ordinary differential equations. Each equation relates to the voltage across and the current through a component and imposes the constraint that the specified relation be satisfied by the analog solver at all times. Since HDL-A models are analog behavioral models, and thus, both time domain and frequency domain apply.

Since HDL-A is a behavioral language that has been targeted to satisfy the IEEE Design Objectives Document for VHDL 1076 B, their syntax and design methodologies are quite similar. A HDL-A model contains two sections: the ENTITY block and the ARCHITECTURE block. The ENTITY block contains the view of the model that the external world sees, such as parameters, pins, ports, and couplings. The ARCHITECTURE block contains the behavioral descriptions. In most of the cases, the internal calculations are known only within the model description, but the results are made available at the terminals listed in the PIN and PORT statements of the ENTITY block. However, internal quantities of a HDL-A model can still be accessed using the COUPLING statement. COUPLING is an OBJECT of TYPE STATE which is shared between more than one HDL-A model. Moreover, a COUPLING quantity can refer to another COUPLING quantity in a model outside the one in which the quantity is found.

Fig. 2-6. shows an example of HDL-A modeled resistor. The voltage across and the current flows through this resistor are coupled so that others can access these quantities with associated coupling variables used in their ENTITY statement. For more complicated modeling, many behavior or symbolic techniques have been proposed [51,52]. HDL-A based macromodel can be constructed accordingly for hierarchical fault diagnosis.

```
ENTITY resistor IS
  GENERIC (rval: analog);
  COUPLING (irl,vrl: analog);
  PIN (pos,neg: electrical);
END ENTITY resistor;

ARCHITECTURE res OF resistor IS
BEGIN
   RELATION
      PROCEDURAL FOR INIT =>
         rval:=1.0e3;
      PROCEDURAL FOR ac, dc, transient =>
         irl:=[pos,neg].v/rval;
         pos.i%=irl; neg.i%=-irl;
         vrl:=[pos,neg].v;
   END RELATION;
END ARCHITECTURE res;
```

Fig.2-6. A HDL-A Modeled Resistor

2.4 Discussion

As discussed in Section 2.2.1, the connection matrix L1 and the parameters in both **a** and **b** are derived from the transformation of the incident matrix of a given circuit. Since the transformation is not unique, a number of differnt connection matrices may be generated. In other words, for a circuit topology and a set of test points, a number of L_{21} matrices may be generated. Interestingly, some transformations may be t-dignosable, but others may be not [31]. For simplicity, a transformation which makes the circuit to be t-diagnosable is referred to as diangosable configuration. This implies that, for the same circuit toloplogy and test points, due to non-unique transformations, some configurations are diagnosable. Therefore, the question is *how to generate a diagnosable configuration from the given circuit topology and test points*. In addition, given a set of test points, it is important to know *what the maximum diagnosability a circuit can achieve? which configuration achieves the maximum diagnosability?* and *which configurations have poor diagnosability?* These issues will be addressed and investigated in the next chapter.

Chapter 3

DIAGNOSABILITY ANALYSIS

This chapter presents the diagnosability analysis of a circuit from its topology and test points. Section 3.1 discusses the upper bound of the diagnosability given a set of test points. Section 3.2 presents a component assignment algorithm that makes L_{21} matrix with the highest global column-rank. Example is provided to demonstrate the procedures of developed analysis process. Finally, summary and discussion for the diagnosability analysis are given in Section 3.3.

3.1 Upper Bound of Diagnosability

Consider a circuit that has n components and m test points. A test point can be either current measurement, referred to as *IM-test point,* or voltage measurement, referred to as *VM-test point*. Let m_I and m_V be the number of IM- and VM-test points, respectively, where $m_I+m_V=m$. For the L_{21} matrix, since it is extracted from L_{11} matrix given in (2-3), it will have the form of $L_{21}=\begin{bmatrix} 0 & E_1 \\ E_2 & 0 \end{bmatrix}$, where E_1 is an m_I-by-n_1 matrix, E_2 is an m_V-by-n_2 matrix, and $n_1+n_2=n$. In general, $n_1 \geq m$ and $n_2 \geq m$. Let r_1 and r_2 be the global column-ranks of E_1 and E_2, respectively. Apparently, $m_I \geq r_1$ and $m_V \geq r_2$. The following lemma and theorems result.

<u>Lemma 3-1</u>. The global column-rank of L_{21} is $r = min\{r_1,r_2\}$.

<u>Proof</u>: It is obvious that any columns in $\begin{bmatrix} 0 \\ E_2 \end{bmatrix}$ are linearly independent of those in $\begin{bmatrix} E_1 \\ 0 \end{bmatrix}$. The global column-rank of $\begin{bmatrix} 0 \\ E_2 \end{bmatrix}$ is the same as that of E_2, and both $\begin{bmatrix} E_1 \\ 0 \end{bmatrix}$ and E_1 have the same rank. This concludes $r = min\{r_1,r_2\}$.

<u>Theorem 3-1</u>. The upper bound of the diagnosability of a circuit is $min\{m_I,m_V\}$.

<u>Proof</u>: Since E_1 is an m_I-by-n_1 matrix, its global column-rank $r_1 \le m_I$. Similarly, $r_2 \le m_V$. Therefore, by Lemma 3-1, the global column-rank of L_{21} is $r=min\{r_1,r_2\} \le r_1 \le m_I$ and $r \le r_2 \le m_V$. This concludes $r \le min\{m_I,m_V\}$.

Theorem 3-1 shows that diagnosability is limited by $min\{m_I,m_V\}$, instead of the number of test points, m, in [39]. In fact, a tighter upper bound can be obtained from E_1 and E_2 [31].

<u>Lemma 3-2</u>. The global column-rank of E_1 is $r_1 \le m_I-1$, if $n_1 > C(m_I,\lceil m_I/2 \rceil)+\kappa$, where $\kappa=1$ for $m_I=2, 3$; and $\kappa=0$ for $m_I \ge 4$

Note that C(x,z) is the combinations of taking z out from x, and $\lceil\,\rceil$ is the ceiling function, i.e., $\lceil m_I/2 \rceil$ is the maximum integer $\ge m_I/2$. For $m_I=4$, each column vector in E_1 is one of the following 16 patterns,

Pattern →	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
Index	0	0	0	0	1	0	0	1	0	1	1	0	1	1	1	1
	0	0	0	1	0	0	1	0	1	0	1	1	0	1	1	1
	0	0	1	0	0	1	0	0	1	1	0	1	1	0	1	1
	0	1	0	0	0	1	1	1	0	0	0	1	1	1	0	1

groups, referred to as *p-Group*, $0 \le p \le 5$. For simplicity, 0-Group and 5-Group are called *trivial-p-Groups*, and the remaining ones are *non-trivial-p-Groups*. It can be easily shown that the global column-rank of the matrix formed by any non-trivial p-Group is $r_1=m_I=4$. However, if a matrix is formed by any non-trivial p-Group with a column vector from the other p-Groups, it will not have a full global column-rank, i.e., $r_1 < m_I$. Therefore, we conclude that the global column-rank of a matrix formed by two or more p-Groups is $r_1 < m_I$. An interesting question is: if a matrix is formed by the distinct column vectors selected from the 16 patterns, *what is the maximum number of column vectors the matrix has a full rank, i.e., $r_I=m_I=4$?* Note that the number of column vectors in a p-Group is $C(m_I,p)$, and the maximum $C(m_I,p)$ occurs at $p=\lceil m_I/2 \rceil$. Therefore, the solution to the above question is $C(m_I,\lceil m_I/2 \rceil) = (4,2) = 6$. In other words, E_1 does not have a full rank, $r_1 < 4$, if $n_1 > 6$. This has been verified by simulation, where all possible E_1 with more than 6 distinct column vectors selected from the 16 patterns are simulated and the resultant ranks are less than 4.

For $m_I=2$, each column vector will be one of the following four patterns,

$$\begin{array}{rcccc} \textit{Pattern index} \rightarrow & 0 & 1 & 2 & 3 \\ & 0 & 0 & 1 & 1 \\ & 0 & 1 & 0 & 1 \end{array}$$

Since the global column-rank of the matrix formed by Patterns #1-#3 is 2, the maximum number of column-rank is $C(m_I,\lceil m_I/2 \rceil)+1=C(2,1)+1=3$. Similarly, for $m_I=3$, the maximum number is also $C(m_I,\lceil m_I/2 \rceil)+1$, i.e., $C(3,2)+1=4$.

Similar to Lemma 3-2, the following lemma also concludes.

Lemma 3-3. The global column-rank of E_2 is $r_2 \le m_V-1$, if $n_2 > C(m_V,\lceil m_V/2 \rceil)+\kappa$, where $\kappa=1$ for $m_V=2, 3$; and $\kappa=0$ for $m_V \ge 4$

The maximum number of column vectors for various m_I are tabulated in Table 3-1.

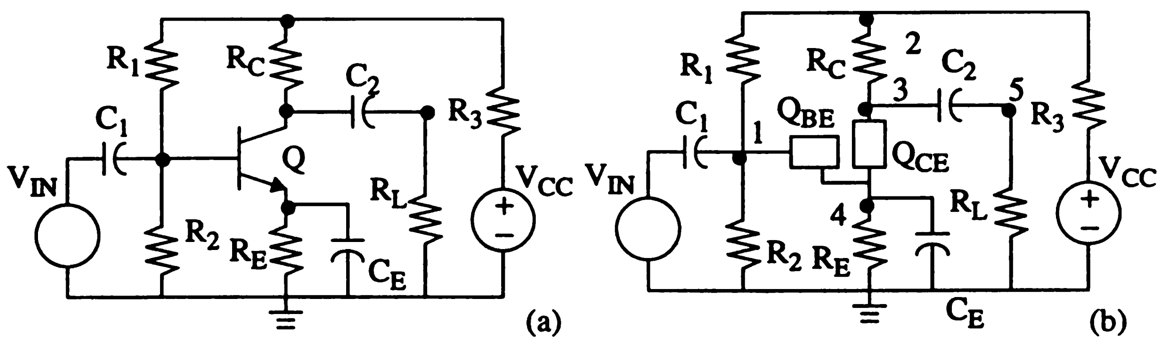
Table 3-1. **Maximum Number of Column Vectors**

m_I (or m_V)	2	3	4	5	6	7
# of col. vectors	3	4	6	10	20	35

Theorem 3-2. (a) The upper bound of the diagnosability of a circuit is (m_V-1) if $m_V \geq m_I$ and $n_2 > C(m_V, \lceil m_V/2 \rceil)+\kappa$; where $\kappa=1$ for $m_V=2, 3$; and $\kappa=0$ for $m_V \geq 4$; and

(b) The upper bound of the diagnosability of a circuit is (m_I-1) if $m_I \geq m_V$ and $n_1 > C(m_I, \lceil m_I/2 \rceil)+\kappa$, where $\kappa=1$ for $m_I=2, 3$; and $\kappa=0$ for $m_I \geq 4$.

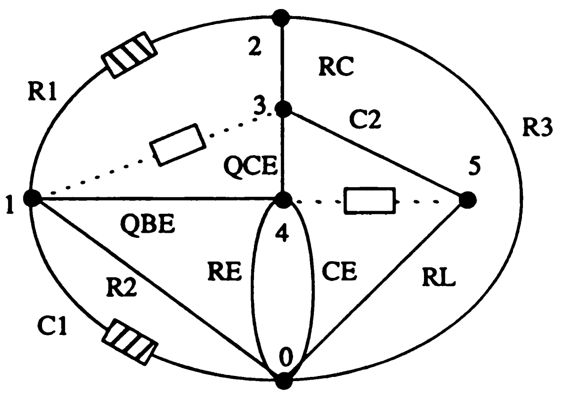
Consider the example circuit in Fig. 3-1 with the selected four test points. Fig. 3-2 illustrates a graph that describes the example circuit in Fig. 3-1(c) and test points. The edges in the graph represent the components in the circuit. The VMTP components for both VM-test points, V13 and V45, are not the real components in the circuit, and the edges representing these components are referred to as *virtual edges*. In this figure, the graph has 11 real edges and two virtual edges, where $n_1=5$, $n_2=6$, and $m_I=m_V=2$. By Theorem 3-2, the upper bound of diagnosability is 1, i.e., the set of test points is only good for at most 1-diagnosability. In fact, the circuit with the connection matrix in Fig. 3-1 is 1-diagnosable, while that with the connection matrix in Fig. 3-3(f) is 0-diagnosable.



$$\begin{array}{c}\mathbf{a}\\ \left[\begin{array}{l}IC1\\ IR1\\ VR2\\ IRC\\ IBQ\\ VCEQ\\ VRE\\ VCE\\ VC2\\ IRL\\ VR3\\ \hline IC1\\ IR1\\ V45\\ V13\end{array}\right]\\ \mathbf{y}\end{array} = \begin{array}{c}\mathbf{L_{11}}\\ \left[\begin{array}{ccccccccccc}0&0&1&0&0&0&1&1&1&0&1\\ 0&0&0&0&0&-1&0&0&-1&0&-1\\ -1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&1&0&0\\ 0&0&0&0&0&-1&1&1&0&0&0\\ 0&1&0&-1&1&0&0&0&0&0&0\\ -1&0&0&0&-1&0&0&0&0&0&0\\ -1&0&0&0&-1&0&0&0&0&0&0\\ -1&1&0&-1&0&0&0&0&0&-1&0\\ 0&0&0&0&0&0&0&0&1&0&0\\ -1&1&0&0&0&0&0&0&0&0&0\\ \hline 0&0&1&0&0&0&1&1&1&0&1\\ 0&0&0&0&0&-1&0&0&-1&0&-1\\ -1&0&0&0&-1&0&0&0&0&-1&0\\ 0&1&0&-1&0&0&0&0&0&0&0\end{array}\right]\\ \mathbf{L_{21}}\end{array} \begin{array}{c}\mathbf{b}\\ \left[\begin{array}{l}VC1\\ VR1\\ IR2\\ VRC\\ VBEQ\\ ICQ\\ IRE\\ ICE\\ IC2\\ VRL\\ IR3\end{array}\right]\end{array} + \begin{array}{c}\mathbf{L_{12}}\\ \left[\begin{array}{cc}0&0\\ 0&0\\ 1&0\\ 0&0\\ 0&0\\ 0&0\\ 1&0\\ 1&0\\ 1&0\\ 0&0\\ 1&-1\\ \hline 0&0\\ 0&0\\ 0&0\\ 0&0\end{array}\right]\\ \mathbf{L_{22}}\end{array} \begin{array}{c}\mathbf{u}\\ \left[\begin{array}{c}V_{IN}\\ V_{CC}\end{array}\right]\end{array}$$

(c)

Fig. 3-1: (a)&(b) Schematic diagrams of example circuit;
(c) Connection matrix X;



▨ IM-Test Points

□ VM-Test Points

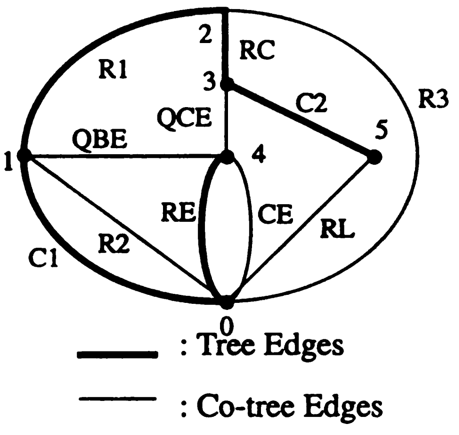
Fig. 3-2. A graph for the circuit in Fig. 3-1(b) with test points.

$$
\begin{bmatrix} IC1 \\ IR1 \\ IRC \\ IC2 \\ IRE \\ VCE \\ VR2 \\ VR3 \\ VRL \\ VBEQ \\ VCEQ \\[6pt] IC1 \\ IR1 \\ V45 \\ V13 \end{bmatrix}
=
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\[6pt]
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & -1 \\
-1 & 1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix} VC1 \\ VR1 \\ VRC \\ VC2 \\ VRE \\ ICE \\ IR2 \\ IR3 \\ IRL \\ IBQ \\ ICQ \end{bmatrix}
+
\begin{bmatrix}
0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 1 & -1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\[6pt]
0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0
\end{bmatrix}
\begin{bmatrix} V_{IN} \\ V_{CC} \end{bmatrix}
$$

(a)

$$
A = \begin{bmatrix}
1 & 0 & -1 & 0 & 0 & 0 & -1 & -1 & 0 & -1 & -1 \\
-1 & -1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & -1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0
\end{bmatrix}
$$

(b)



(c)

$$
D = \begin{bmatrix}
0 & -1 & -1 & -1 & -1 & -1 \\
0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & -1 & 0 & -1 \\
0 & 0 & 0 & -1 & 0 & 0 \\
1 & 0 & 0 & 0 & -1 & -1
\end{bmatrix}
$$

(d)

Fig. 3-3. Example circuit: (a) Connection Matrix Y; (b) incidence matrix; (c) a graph and (d) generated D-matrix.

3.2 Diagnosability Analysis: Component Assignment Algorithm

The connection matrix of a circuit is derived from the incidence matrix A of the circuit together with the component assignment. Our objective is to develop an efficient component assignment algorithm that provides a connection matrix with the maximum diagnosability [31]. These assignments are based on the circuit topology, which is applicable to not only the diagnosability analysis, but also the redesignability analysis presented in next chapter.

Consider a circuit consisting of n components and m test points which includes m_I *IMTPs* and m_V *VMTP*. According to (2-2), nodal analysis and loop analysis are performed for tree and co-tree elements, respectively. Therefore, the first assignment rule is obtained as follows,

Rule 1. (a) All IMTP components are assigned to the tree; and

(b) All VMTP components, except the virtual ones, are assigned to the co-tree.

Each component of an electrical network can be constructed so that it is included in at least one loop. Thus, two types of loops are then defined in this implementation: *T-loop* and *M-loop*. A T-loop is a loop associated with a VMTP component, while an M-loop is a loop associated with a non-TP component, i.e., neither an IMTP nor VMTP component. A T-loop, associated with a VMTP component, may contain IMTP components(s) and non-TP components, but not the VMTP components. On the other hand, an M-loop includes the non-TP component it is associated with and at least one IMTP component. An M-loop may include IMTP components and the other non-TP components, but not the VMTP components. This concludes that a VMTP component is included only in the T-loop it is associated with, but never in any other T-loops, nor M-loops.

By Rule 1(b), a VMTP component is assigned to the co-tree. Therefore, the remaining components in a T-loop associated with this VMTP component are assigned to the tree. On the other hand, the non-TP component associated with an M-loop is assigned to the co-tree, and the remaining components in the M-loop are assigned to the tree. The following rule results.

Rule 2. (a) In a T-loop, its associated VMTP component is assigned to the co-tree, and the remaining ones are assigned to the tree.

(b) In a M-loop, its associated non-IMTP component is assigned to the co-tree, and the remaining ones are assigned to the tree.

A graph has $(N_d\text{-}1)$ tree edges, where N_d is the number of nodes in the graph. Excluding the IMTP components which, by Rule 1(a), are assigned to the tree, the graph has only $(N_d\text{-}1\text{-}m_I)$ tree edges available. Suppose that a T-loop includes N_t non-TP components, by Rule 2(a), the N_t non-TP components are assigned to the tree. This implies that the number of non-TP components in a candidate T-loop cannot exceed $(N_d\text{-}1\text{-}m_I)$. Similarly, the number of non-TP components in a candidate M-loop, excluding the non-TP components which the M-loop is associated with, also cannot exceed $(N_d\text{-}1\text{-}m_I)$.

Rule 3. The number of non-TP components in a candidate T- or M-loop cannot exceed $(N_d\text{-}1\text{-}m_I)$.

The L_{21} matrix can be represented in terms of test points and components as follows,

L_{21} Matrix

$$\begin{array}{c|cccc|ccc}
 & PT_1 & PT_2 & \cdots & PT_{n2} & PC_1 & \cdots & PC_{n1} \\
\hline
TPI_1 & 0 & 0 & \cdots & 0 & & & \\
TPI_2 & 0 & 0 & \cdots & 0 & & E_1 & \\
\cdots & & & & & & & \\
TPI_{mI} & 0 & 0 & \cdots & 0 & & & \\
\hline
TPV_1 & & & & & 0 & \cdots & 0 \\
TPV_2 & & E_2 & & & 0 & \cdots & 0 \\
\cdots & & & & & & & \\
TPV_{mV} & & & & & 0 & \cdots & 0
\end{array} \qquad (3\text{-}1)$$

where TPV_1, TPV_2, ..., and TPV_{mV} are the VM-test points, TPI_1, TPI_2, ..., and TPI_{mI} are the IM-test points, PT_1, PT_2, .., and PT_{n2} are the tree-components, and PC_1, PC_2, .., and PC_{n1} are the co-tree components. Let T_{LS}, be a *T-loop set* which is defined as

$T_{LS} = \{(T_1, T_2, ..., T_{mV}) \mid T_i$ is a candidate T-loop associated with $TPV_i\}$ (3-2)

A candidate T-loop T_i constitutes the i-th row of the matrix E_2. A *candidate T-loop set* is the T_{LS} whose corresponding E_2 matrix has a non-zero global column-rank. In addition, by Rule 3, the total number of non-TP components included in a candidate T-loop set must not exceed $(N_d\text{-}1\text{-}m_I)$.

<u>Rule 4</u>. A candidate T-loop set must satisfy

(a) The total number of non-TP components is $(N_d\text{-}m_I\text{-}1)$, and

(b) The global column-rank of E_2 is $r_2 \geq 1$.

By definition, an IMTP component contributes to a column vector of E_2. If an IMTP component is not included in a T-loop set, a zero column results in E_2 and violates Rule 4(b). Since all the $(N_d\text{-}m_I\text{-}1)$ non-TP components and m_I IMTP components, i.e., PT_1, PT_2,

.., and PT_{n2}, included in a candidate T-loop set are assigned to the tree, the remaining components in the circuit, i.e., PC_1, PC_2, .., and PT_{n1}, are assigned to the co-tree. Therefore, an M-loop set, M_{LS}, is defined as

$$M_{LS} = \{(M_1,M_2, ...,M_{n1}) \mid M_j \text{ is a candidate M-loop associated with } PC_j\} \quad (3\text{-}3)$$

A candidate M-loop M_j constitutes the j-th column of the matrix E_1 corresponding to PC_j. A candidate M-loop set associated with a candidate T_{LS} is the M_{LS} whose corresponding E_1 matrix has a non-zero global column-rank. Similar to Rule 4, the following rule results

Rule 5. A candidate M-loop set must satisfy

(a) The total number of non-TP components in (T_{LS},M_{LS}) is $(N_d\text{-}m_I\text{-}1)$, and

(b) The global column-rank of E_1 is $r_1 \geq 1$.

Based on Rules 4 and 5, we derive pairs of (T_{LS},M_{LS}) with nonzero ranks r_1 and r_2. By Lemma 3-1, the global column-rank $r=min\{r_1,r_2\}$, the pair (T_{LS},M_{LS}) with the maximum rank r is then chosen for constructing the connection matrix which achieves the maximum diagnosability. The component assignment procedure is summarized in Algorithm 3-1. The following example illustrates the stepwise procedure of this algorithm and the detailed implementation.

Consider the circuit of Fig. 3-1(b), where n=11, m=4, $m_I=m_V=2$, $N_d=6$, $N_d\text{-}m_I\text{-}1=3$, $n_1=5$, and $n_2=6$. The VM-test points are V45 and V13, the IM-test points are IC1 and IR1, and the non-TP components include QBE, QCE, RL, RC, C2, R2, R3, RE, and CE. In Step 1, both IMTP components C1 and R1 are assigned to the tree. In Step 2, both T- and M-loops are generated using following symmetric matrix, referred to as the *adjacency matrix*, which describes the relationship between any two nodes, where "1" means connection and "0" means no connection.

$$\begin{array}{c|cccccc|}
\text{Node Index} & 0 & 1 & 2 & 3 & 4 & 5 \\
\hline
0 & 0 & 1 & 1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 1 & 0 \\
2 & 1 & 1 & 0 & 1 & 0 & 0 \\
3 & 0 & 0 & 1 & 0 & 1 & 1 \\
4 & 1 & 1 & 0 & 1 & 0 & 0 \\
5 & 1 & 0 & 0 & 1 & 0 & 0 \\
\hline
\end{array}$$

For V45, we generate a number of loops starting from node 4 to node 5. A path is invalid if a node is re-visited. This implementation chooses the next node to visit in an ascending order of the assigned node number, for simplicity. Thus, we start with node 4 which connects to nodes 0, 1, and 3. We first select node 0 which connects to nodes 1, 2, 4, and 5, where node 4 has been visited, and only nodes 1, 2, and 5 are valid. Consequently, we select the next valid nodes in the sequence of nodes 2, 3, and 5. Therefore, a path 4-0-1-2-3-5 connects node 4 to node 5, and the path with the VMTP component for V45 form a T-loop for V45. From Fig. 3-2, the path includes the following components: both edges for RE and CE in "4-0", R2 and C1 in "0-1", R1 in "1-2", RC in 2-3, and C2 in "3-5". Similarly, all T-loops for V45 can be generated as follows,

4 - 0 - 1 - 2 - 3 - 5	{RE,CE},{R2,C1},R1,RC,C2	4 - 1 - 2 - 0 - 5	QBE,R1,R3,RL
4 - 0 - 2 - 3 - 5	{RE,CE},R3,RC C2	4 - 1 - 2 - 3 - 5	QBE,R1,RC,C2
4 - 0 - 5	{RE,CE},RL	4 - 3 - 2 - 0 - 5	QCE,RC,R3,RL
4 - 1 - 0 - 2 - 3 - 5	QBE,{C1,R2},R3,RC,C2	4 - 3 - 2 - 1 - 0 - 5	QCE,RC,R1,{C1,R2},RL
4 - 1 - 0 - 5	QBE,{C1,R2},RL	4 - 3 - 5	QCE,C2

By Rule 2, the non-TP components must be less than or equal to 3. Thus, all candidate T-

Algorithm 3-1. Component Assignment for Diagnosability Analysis

```
Step 1. Assign the VMTP and IMTP components by Rule 1.
     2. Generate all candidate T-loops and M-loops by Rule 2, and
        assign the components by Rule 3.
     3. (Generate matrices $E_1$ and $E_2$)
        3.1 Select a candidate T-loop $T_i$ for each $TPV_i$ to construct a T-loop set
            $T_{LS}$ as in (3-2). IF end-of-selection, GOTO Step 4
        3.2 IF the $T_{LS}$ does not satisfy Rule 4, GOTO Step 3.1
        3.3 Select a candidate M-loop $M_j$ for each $PC_j$ to construct a M-loop set
            $M_{LS}$ as in (3-3). IF end-of-selection, GOTO Step 3.1
        3.4 IF the $M_{LS}$ does not satisfy Rule 5, GOTO Step 3.3
        3.5 The component assignment is recorded as well as $r_1$ and $r_2$.
     4. (Maximum Diagnosability)
        4.1 Find the maximum t, where $t=min\{r_1,r_2\}$, for each candidate com-
            ponent assignments.
            (The component assignment generates a connection matrix which
            achieves t-diagnosability.)
```

loops for V45 are listed below.

1.	RE,C1,R1,RC,C2	5	QBE,C1,RL	9	QCE, RC, R1, C1, RL
2.	CE,C1,R1,RC,C2	6	QBE,R2,RL	10	QCE, C2
3	RE,RL	7	QBE, R1, R3, RL		
4	CE,RL	8	QBE, R1, RC, C2		

A candidate T-loop set is selected from T_{LS}, in (3-1), and must satisfy Rule 4. In order to reduce the complexity of searching process, the T-loops, as shown in Table 3-2, are sorted in an ascending order for the numbers of non-TP components. Similarly, the sorted T-loops for V_{13} and the sorted candidate M-loops for all non-TP components are also listed in Table 3-2.

In Step 3.1, the first T-loop, (RE,RL), in V45, and the first one, (R1,RC), in V13, are selected. Since the IMTP-component C1 is not included in either T-loop, a zero column occurs in E_2 and violates Rule 4(b). Thus, the pair cannot be a candidate T-loop set. The procedure is repeated. The first candidate T-loop set is obtained by selecting the fourth T-loop, (QBE,C1,RL), of V45, and the first T-loop, (R1,RC), of V13. It generates the following E_2 matrix

$$\begin{array}{cccccc} & QBE & C1 & RL & R1 & RC \\ V45 & 1 & 1 & 1 & 0 & 0 \\ V13 & 0 & 0 & 0 & 1 & 1 \end{array}$$

which satisfies Rule 4 with $r_2=1$. Similarly, we also obtain two other candidate T-loop sets: the eighth T-loop, (QCE,RC,R1,C1,RL), of V45, and the first T-loop, (R1,RC), of V13; and the tenth T-loop, (CE,C1,R1,RC,C2), of V45, and the first T-loop, (R1,RC), of V13. Both also satisfy Rule 4 with $r_2=1$.

Table 3-2. T- & M-loop Candidates.

V45	1. RE,RL	5 QBE,R2,RL	9 RE,C1,R1,RC,C2
	2. CE,RL	6 QBE,R1,R3,RL	10 CE,C1,R1,RC,C2
	3 QCE,C2	7 QBE,R1,RC,C2	
	4 QBE,C1,RL	8 QCE,RC,R1,C1,RL	
V13	1. R1,RC	6 R2,RE.QCE	11 R1,R3,RE,QCE
	2. QBE,QCE	7 C1,CE,QCE	12 R1,R3,CE,QCE
	3 C1,R3,RC	8 R2,CE,QCE	13 R1,R3,RL,C2
	4 R2,R3,RC	9 C1,RL,C2	
	5 C1,RE.QCE	10 R2,RL,C2	
QBE	1. C1,RE	4 R1,R3,CE	7 C1,R3,RC,QCE
	2. C1,CE	5 R1,RC,QCE	
	3 R1,R3,RE	6 C1,RL,C2,QCE	
QCE	1. RC,R1,QBE	4. RC,R1,R2,CE	7. RC,R3,C1,QBE
	2. RC,R1,C1,CE	5. RC,R1,R2,RE	
	3. RC,R1,C1,RE	6. C2,RL,C1,QBE	
RL	1. C1,R1,RC,C2	2. R2,R1,RC,C2	3. C1,QBE,QCE,C2
RC	1. R1,QBE,QCE	5. R3,C1,RE,QCE	9. C1,R3,QBE,QCE
	2. R1,C1,RE,QCE	6. R1,C2,RE,QCE	10. R1,R2,RE,QCE
	3. R1,C1,CE,QCE	7. R1,R2,CE,QCE	
	4. R1,C1,RL,C2	8. R1,R2,RL,C2	
C2	1. RC,R1,C1,RL	2. RC,R1,R2,RL	3. QCE,QBE,C1,RL
R2	1. C1	3. CE,QCE,RC,R1	5. RL,C2,RC,R1
	2. R3,R1	4. RE,QCE,RC,R1	
R3	1. C1,R1	3. CE,QBE,R1	5. C1,QBE,QCE,RC
	2. R2,R1	4. RE,QBE,R1	
RE CE	1. C1,QBE	3. C1,R1,RC,QCE	4. R2,R1,RC,QCE
	2. R3,R1,QBE		

Consider the first candidate T-loop set, where the IMTP components, R1 and C1, and the non-TP components, QBE, RL, and RC, are assigned to the tree. Since the total number of tree components is 5, the remaining components, C2, RE, CE, R2, R3, and QCE, must be assigned to the co-tree. In Step 3.3, the M-loop sets associated with the candidate T-loop set are formed by the M-loops for the non-TP components C2, R2, R3, QCE, RE, and CE. Note that the non-TP components in an M-loop are assigned to the tree. Since the candidate T-loop set has already contained all 5 tree-components, a candidate M-loop must be the one that includes only the tree components. More specifically, consider the M-loops of C2 in Table 3-2, where it has 3 possible M-loops. The second M-loop, (RC,R1,R2,RL), cannot be chosen because it contains a co-tree component R2. Similarly, the 3rd M-loop is also rejected because it contains a co-tree component QCE. For the first M-loop, (RC, C1,R1,RL), it contains only the tree components and is then chosen. Similarly, the first M-loops of all other co-tree components, RE, CE, R2, R3, and QCE, are selected, and the corresponding matrix E_1 is,

$$\begin{array}{ccccccc} & C2 & RE & CE & R2 & R3 & QCE \\ IC1 & 1 & 1 & 1 & 1 & 1 & 0 \\ IR1 & 1 & 0 & 0 & 0 & 1 & 1 \end{array}$$

which satisfies Rule 5 with r_1=1. Thus, the component assignment, C1, R1, QBE, RL, and RC in the tree, and C2, RE, CE, R2, R3, and QCE in the co-tree, generates the partition in Fig. 3-4 and Connection matrix Y in Fig. 3-1(c) which has 1-diagnosability.

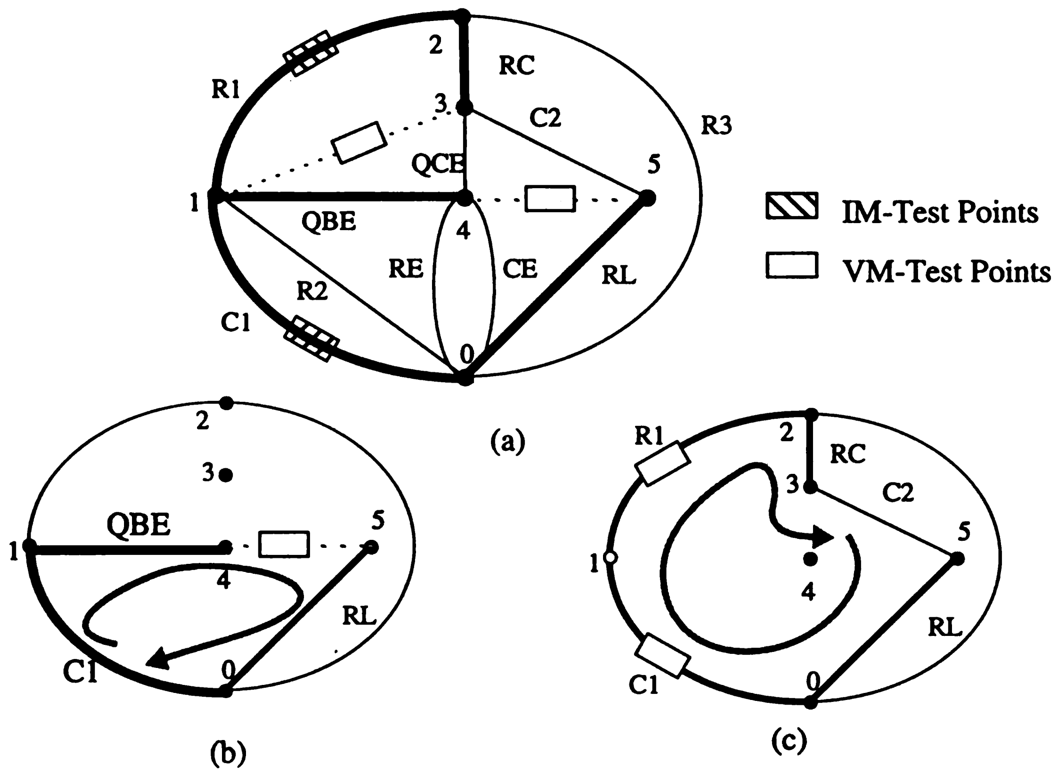


Fig. 3-4. Graph Representation: (a) Components and Test Points; (b) T-loop for V45; and (c) M-loop for C2.

For the second candidate T-loop set, it contains the tree-components, QCE, RC, C1, R1, and RL, and the corresponding E_2 matrix has $r_2=1$. Similarly, an associated candidate M-loop set is constructed by selecting the first M-loops of C2, R3, and R2, the third M-loops of RE and CE, and the fifth M-loop of QBE, and the corresponding E_1 matrix has $r_1=1$. Finally, for the third candidate T-loop set containing the tree-components, CE, C1, R1, RC, and C2, the corresponding E_2 matrix has $r_2=1$. However, since each M-loop in RE contains at least one co-tree component, a zero column vector occurs in E_2 at the column corresponding to RE, and thus no candidate M-loop sets exist. Therefore, only two different component assignments are concluded. In Step 4.1, both achieve the same 1-diagnosability.

3.3 Discussion

Given a circuit topology and test points, Algorithm 3-1 selects a connection matrix whcih achieves the maximum diagnosability that the circuit can achieve. The algorithm has been implemented to the system ADTS as a "Diagnosability Analysis" process. The algorithm also identifies the components having poor diagnosability from the connection matrix. To enhance the circuit diagnosability, the portions having poor diagnosability may be redesigned using the redesignability analysis discussed in the next chapter.

Chapter 4

REDESIGNABILITY ANALYSIS

The redesign problem can be formulated as follows: assume that the schematic circuit diagram is given, but some parts of the circuit are unknown or missing, the circuit is *re-designable* if the input/output (I/O) relation of each missing part can be traced, and the derived I/O relations can then be implemented. Note that we do not intend to discover the exact circuit schematic and components that were present in the circuit originally implemented [32,53]. Rather, the functions originally intended to be present will be identical. Thus, many possible implementations exist to recover their I/O relations.

Based on our system model, CCM, with the known test input signals ***u*** and system responses ***y*** and the characteristics of the components in the *Tester Group*, as in (2-13), the I/O relations of the components in the *Testee Group* can be derived and evaluated. That is, the I/O relation Z^2 can be derived as long as $\overline{b^2}$ and a^2 exist. The redesign problem can be stated in a similar way, as illustrated in Fig. 4-1, where the I/O relation of each *Missing Part* is derived using the characteristics of the *Known Parts* together with the applied input signals and the measured responses. Thus, both fault diagnosis problem and redesign problem share some similarities. Based on the system model and fault diagnosis algorithm, the *Missing Parts* of the redesign problem in Fig. 4-1 are equivalent to the components in the *Testee Group* as in Fig. 2-4, while the *Known Parts* are equivalent to those in the *Tester Group*.

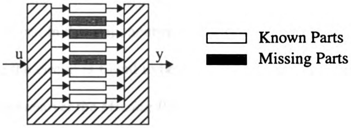


Fig. 4-1 Redesign problem.

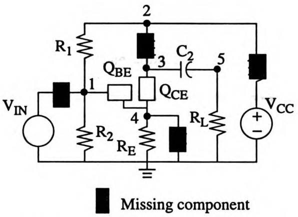


Fig. 4-2 Example circuit with four missing components.

Therefore, the process of deriving the I/O relations for the components in the *Testee Group* can be applied to derive the I/O relations for the *Missing Parts* in the redesign problem.

Our proposed redesign process is comprised of two steps: (1) redesignability check; and (2) redesign solution. Based on a set of rules derived from circuit topology, the former step checks whether a given circuit is redesignable with respect to a set of *Missing Parts*, given some test points. In this step, neither circuit simulations nor test vector applications are needed. Once the circuit is found to be redesignable, the I/O relations of the *Missing Parts* are derived to recover the *Missing Parts*. This aim is to develop an algorithm that efficiently determines whether the circuit is redesignable.

In the next section, rules for redesignability check are discussed. Section 4.2 presents the development of the proposed redesign process. Finally, discussion is given in Section 4.3.

4.1 Unredesignability/Redesignability

Given a target circuit with the selected test points, one can derive the connection matrices L_{11} and L_{12}, in (2-1), from the circuit topology, and L_{21} and L_{22} from the test points. Suppose that the circuit consists of n components and m test points, the L_{21} matrix is an m-by-n matrix. Suppose also that there exist q missing components in that circuit, a matrix, referred to as *MP-matrix*, is formed by those columns in L_{21} which are corresponding to these q missing components. Therefore, the following lemma results.

Lemma 4-1. A target circuit is redesignable if the corresponding MP-matrix has a full global column-rank.

Proof: A full global column-rank MP-matrix is left invertible. Thus, the I/O relation of each missing component can be derived as shown in (2-10a). This concludes that the circuit is redesignable.

Example 1.

Consider the example circuit and test points in Fig. 3-1(b). Suppose that C1, RC, CE, and R3 are four missing components, as the shaded blocks indicated in Fig. 4-2. The following MP-matrix is extracted from the columns 1,4,8 and 11 of the L_{21} matrix in Fig. 3-3(a),

$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}$$

and it has a full global column-rank. By Lemma 4-1, the circuit is redesignable.

By definition, a circuit is *t-diagnosable* if, given the results of all allowable tests, one can uniquely identify all faulty components provided that the number of faulty components does not exceed t [54,55]. The following theorem results.

Theorem 4-1. A t-diagnosable circuit is definitely redesignable if the number of missing components in the target circuit is less than t.

Proof: If the circuit contains less than t missing components, then the corresponding MP-matrix is contained in a $L_{21}{}^2$ matrix which has a full global column-rank because the circuit is t-diagnosable. Thus, the MP-matrix also has a full global column-rank. By Lemma 4-1, the circuit is redesignable.

The circuit in Fig. 4-2 has been shown as 1-diagnosable [39] if partitioned properly. By Theorem 4-1, the circuit with any single missing component is definitely redesignable. The t-diagnosability in Theorem 4-1 ensures the circuit redesignability if the circuit has at most t missing components. However, a t-diagnosable circuit may still be redesignable even when it has more than t missing components.

Example 2.

Consider the same circuit with the same connection matrix in Example 1, except changing the missing components as RC, QBE, C2 and RL, as shown in Fig. 4-3(a). The MP-matrix,

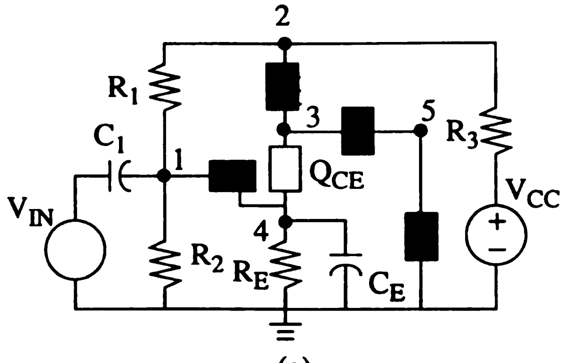
$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & -1 & -1 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix}$$

does not have a full global column-rank. However, with an alternative connection matrix shown in Fig. 4-3(b), the MP-matrix corresponding to the missing components, RC, C2, RL, and QBE is

$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & -1 & 0 \\ -1 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix}$$

and it has a full global column-rank. Thus, the circuit is redesignable.

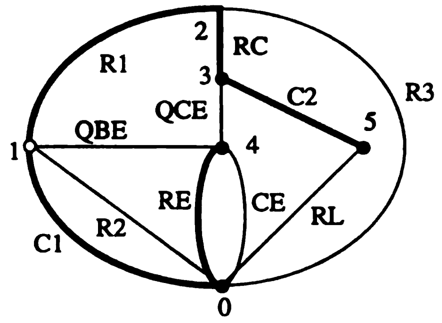
Example 2 shows that the corresponding MP-matrices for different connection matrices may be different. The MP-matrix for one connection matrix may not be full global column-rank, but it may be full global column-rank for other connection matrices. There-



(a)

$$
\begin{bmatrix} IC1 \\ IR1 \\ IRC \\ IC2 \\ IRE \\ VCE \\ VR2 \\ VR3 \\ VRL \\ VBEQ \\ VCEQ \\ IC1 \\ IR1 \\ V45 \\ V13 \end{bmatrix}
=
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & -1 \\
-1 & 1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix} VC1 \\ VR1 \\ VRC \\ VC2 \\ VRE \\ ICE \\ IR2 \\ IR3 \\ IRL \\ IBQ \\ ICQ \end{bmatrix}
+
\begin{bmatrix}
0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 1 & -1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0
\end{bmatrix}
\begin{bmatrix} V_{IN} \\ V_{CC} \end{bmatrix}
$$

(b)



(c)

Fig4-3. Example circuit: (a) with missing components; (b) a connection matrix; and (c) a graph.

fore, a circuit is redesignable if there exists at least one connection matrix in which the corresponding MP-matrix has a full global column-rank. Otherwise, the circuit is unredesignable.

Theorem 4-2. A circuit is unredesignable if there exist no full global column-rank MP matrices for all possible connection matrices.

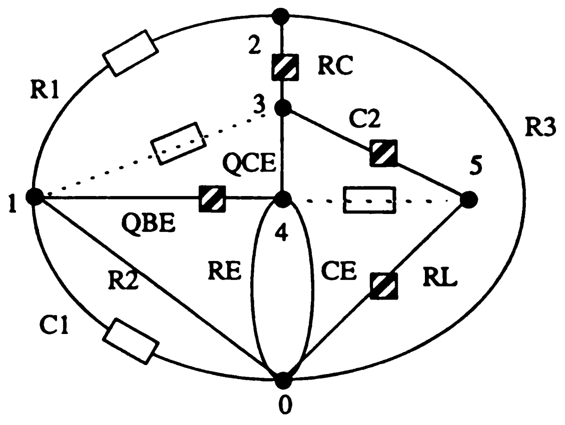
If a circuit has more than m missing components, where m is the number of test points, then we will never be able to find a full global column-rank MP-matrix. Thus, the following theorem results.

Theorem 4-3. A circuit is unredesignable if the number of missing components exceeds m.

It is impractical to check the rank of the corresponding MP-matrices of all possible connection matrices of a target circuit. An efficient algorithm is developed to find a connection matrix whose corresponding MP-matrix has a full global column-rank from circuit topology, which will be presented in the next section.

4.2 Redesignability Analysis: Component Assignment Algorithm

Consider a target circuit that has n components, m test points, and q missing components. Let m_I and m_V be the numbers of IM- and VM-test points, respectively, where $m_I+m_V=m$. Fig. 4-4 illustrates a graph that describes the example circuit in Fig. 4-3(a), where both test points and missing components are indicated. Similar to the partitioned L_{21}



▨ Missing Components ▭ Test Points

Fig. 4-4 A graph for the circuit in Fig. 4-3(a) with test points and missing components.

matrix in (3-1), the MP-matrix can be partitioned as follows,

MP-Matrix

$$\begin{array}{|c|cccc|ccc|}
\hline
 & MC_1 & MC_2 & \cdots & MC_{qV} & MC_{qV+1} & \cdots & MC_q \\
\hline
TPI_1 & 0 & 0 & \cdots & 0 & & & \\
TPI_2 & 0 & 0 & \cdots & 0 & & \boxed{\text{MY}} & \\
\cdots & & & & & & & \\
TPI_{mI} & 0 & 0 & \cdots & 0 & & & \\
\hline
TPV_1 & & & & & 0 & \cdots & 0 \\
TPV_2 & & \boxed{\text{MX}} & & & 0 & \cdots & 0 \\
\cdots & & & & & & & \\
TPV_{mV} & & & & & 0 & \cdots & 0 \\
\hline
\end{array}$$

where TPV_1, TPV_2, ..., and TPV_{mV} are the VM-test points, TPI_1, TPI_2, ..., and TPI_{mI} are the IM-test points, and MC_1,..,MC_{qV}, MC_{qV+1},..,MC_q are the missing components. Constructing linearly independent columns in a MP-matrix is equivalent to constructing linearly independent columns in both matrices MX and MY. Without loss of generality, we assume that the matrix MX includes q_V's missing components, and MY includes the remaining ones. Let a *T-loop set, T_{LS}*, be defined as (3-2), we define a *candidate T-loop set* as the T_{LS} which constitutes a MX matrix with a global column-rank of q_V. The necessary condition for MX to have a global column-rank of q_V is $m_V \geq q_V$. Also, the necessary condition for MY to have a global column-rank of $(q-q_V)$ is $m-m_V \geq q-q_V$. Thus, the number of missing components in a candidate T-loop set is ranged between m_V and $q-m+m_V$.

<u>Rule 4-1</u>. A candidate T-loop set with q_V's missing components must satisfy

(a) $m_V \geq q_V \geq q-m+m_V$;

(b) All candidate T-loops constitute MX with a global column-rank of q_V.

(c) Rule 4(a).

Given a candidate T_{LS} which includes the missing components MC_i, $i=1,2,..,q_V$, let a *M-loop set*, M_{LS}, be defined as (3-2) and a candidate M-loop M_{qV+j} constitutes the j-th column of the matrix MY. Therefore, we define a candidate M-loop set associated with a candidate T_{LS} as the M_{LS} which constitutes a MY matrix with a global column-rank of ($q-q_V$). Similar to Rule 4-1, the following rule results

Rule 4-2 A candidate M-loop set with ($q-q_V$)'s missing components must satisfy

(a) All candidate M-loops constitute MY with a global column-rank of ($q-q_V$).

(b) Rule 5(a).

Consequently, a MP-matrix formed by a pair of a candidate T_{LS} and its associated candidate M_{LS}, (T_{LS},M_{LS}), has a full global column-rank q.

The component assignment procedure is summarized in Algorithm 4-1. The following example illustrates the stepwise procedure of Algorithm 4-1 and the detailed implementation.

Example 3.

Consider the circuit of Fig. 4-3(a) which has 11 components. The VM-test points are V45 and V13, and the corresponding VMTP components are virtual components. The IM-test points are IC1 and IR1, and the IMTP components are C1 and R1. The missing components are RC, C2, RL, and QBE. In Step 1, q=m=4, and, by Step 2, both components C1 and R1 are assigned to the tree. In Step 3, both T- and M-loops are generated using symmetric adjacent matrix described in section 3.

For V45, all T-loops can be generated as follows,

4 - 0 - 1 - 2 - 3 - 5	{RE,CE},{R2,C1},R1,RC,C2	4 - 1 - 2 - 0 - 5	QBE,R1,R3,RL
4 - 0 - 2 - 3 - 5	{RE,CE},R3,RC C2	4 - 1 - 2 - 3 - 5	QBE,R1,RC,C2
4 - 0 - 5	{RE,CE},RL	4 - 3 - 2 - 0 - 5	QCE,RC,R3,RL
4 - 1 - 0 - 2 - 3 - 5	QBE,{C1,R2},R3,RC,C2	4 - 3 - 2 - 1 - 0 - 5	QCE,RC,R1,{C1,R2},RL
4 - 1 - 0 - 5	QBE,{C1,R2},RL	4 - 3 - 5	QCE,C2

In Fig. 4-4, the number of non-IMTP components is N_d-m_V=6-2=4, where the IMTP components are C1 and R1. By Rule 2, all candidate T-loops for V45 are listed below.

1.	RE,C1,R1,RC,C2	5	QBE,C1,RL	9	QCE, RC, R1, C1, RL
2.	CE,C1,R1,RC,C2	6	QBE,R2,RL	10	QCE, C2
3	RE,RL	7	QBE, R1, R3, RL		
4	CE,RL	8	QBE, R1, RC, C2		

According to Step 4.1, a candidate T-loop set is selected from T_{LS} in (4-1) which satisfies Rule 4-1, where the total non-IMTP components in the candidate T-loop set cannot exceed (N_d-m_V-1). Therefore, we sort the above table so that the number of non-IMTP components is in an ascending order. In this arrangement, we may have a better chance to derive a candidate T-loop set which meets Rule 4-1(b).

V45	1.	RE,RL	4	QBE,C1,RL	7	QBE, R1, RC, C2
	2.	CE,RL	5	QBE,R2,RL	8	RE,C1,R1,RC,C2
	3	QCE, C2	6	QBE, R1, R3, RL	9	CE,C1,R1,RC,C2

Similarly, the sorted T-loops for V13 and M-loops for QBE, RL, RC, and C2 are listed as follows.

V13	1. R1,RC 2. QBE,QCE 3 C1,R3,Rc	4 C1,RE.QCE 5 C1,CE,QCE 6 C1,RL,C2	7 R1,R3,RE,QCE 8 R1,R3,CE,QCE 9 R1,R3,R:,C2
QBE	1. C1,RE 2. C1,CE 3 R1,R3,RE	4 R1,R3,CE 5 R1,RC,QCE 6 C1,RL,C2,QCE	7 C1,R3,RC,QCE
RL	1. C1,R1,RC,C2	2. R2,R1,RC,C2	3. C1,QBE,QCE,C2
RC	1. R1,QBE,QCE 2. R1,C1,RE,QCE 3. R1,C1,CE,QCE	4. R1,C1,RL,C2 5. R3,C1,RE,QCE 6. R1,C2,RE,QCE	7. R1,R2,CE,QCE 8. R1,R2,RL,C2
C2	1. RC,R1,C1,RL	2. RC,R1,R2,RL	3. QCE,QBE,C1,RL

In Step 4.1, we choose the first T-loop, {RE,RL}, for V45, as shown in Fig. 4-5(a), and the first T-loop, {R1,RC}, for V13, as shown in Fig. 4-5(b), to form a candidate T-loop set which satisfies Rule 4-1. The matrix MX has a global column-rank of 2, where the matrix is

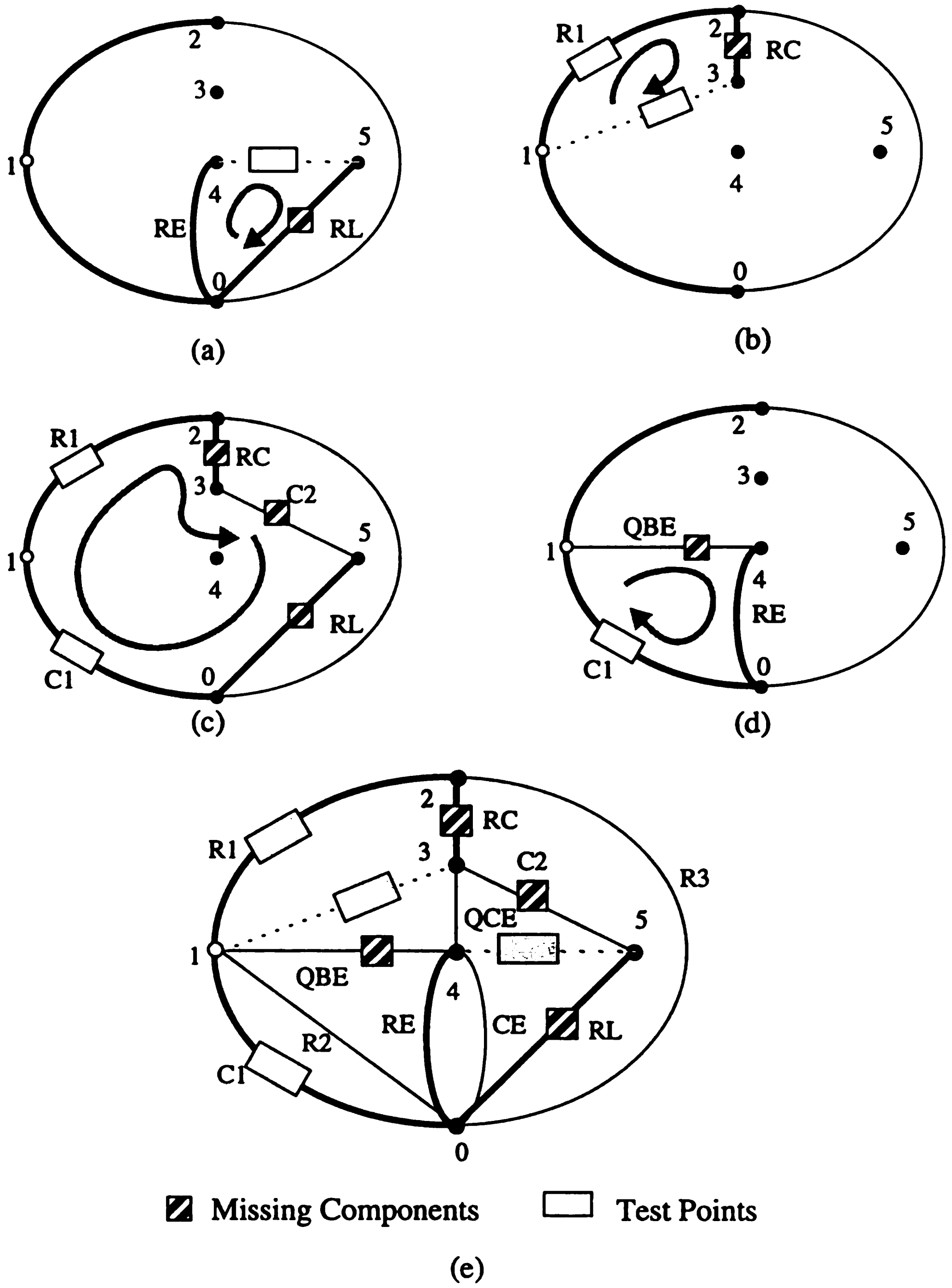
$$\begin{array}{ccc} & RC & RL \\ V45 & 0 & 1 \\ V13 & 1 & 0 \end{array}$$

The uncovered missing components are C2 and QBE. In Step 4.3, the first M-loops for both C2 and QBE, as shown in Fig. 4-5(c) and Fig. 4-5(d), are chosen to form a candidate M-loop set which satisfies Rule 4-2. The matrix MY is

$$\begin{array}{ccc} & C2 & QBE \\ IR1 & 1 & 0 \\ IC1 & 1 & 1 \end{array}$$

This concludes that the components {R1,C1,RE,RL,RC} are assigned to the tree and the remaining ones to the co-tree, as shown in Fig. 4-5(e).

In fact, the component assignment for the graph in Fig. 4-5(e) is derived by taking the 8-th T-loop for V45, the first T-loop for V13, the first M-loop for QBE, and the first M-loop of RL. Thus, RE, C1, R1, RC, and C2 are assigned to the tree and the remaining ones are to the co-tree. Algorithm 4-1 has been implemented to the test system, ADTS, as a



Figu. 4-5. Component Assignment Procedure: (a) A T-loop for V45; (b) A T-loop for V13; (c) A M-loop for C2; (d) A M-loop for QBE; and (e) Component Assignment.

Algorithm 4-1. Component Assignment for Redesignability Analysis

```
Step 0. Let n, m, and q be the number of components, test points, and
        missing components.
     1. IF $q > m$, GOTO Step 5
     2. Assign the VMTP and IMTP components by Rule 1.
     3. Generate all candidate T-loops and M-loops by Rule 2, and
        assign the components by Rule 3.
     4. (Generate matrices MX and MY)
        4.1 Select a candidate T-loop $T_i$ for each $TPV_i$ to construct a candi-
            date T-loop set $T_{LS}$. IF end-of-selection, GOTO Step 4-5
        4.2 IF the $T_{LS}$ does not satisfy Rule 4-1, GOTO Step 4.1
        4.3 Select a candidate M-loop $M_j$ for each $MC_j$ to construct a candi-
            date M-loop set $M_{LS}$. IF end-of-selection, GOTO Step 4.1
        4.4 IF the $M_{LS}$ does not satisfy Rule 4-2, GOTO Step 4.3
        4.5 The component assignment is done, and the circuit is redesign-
            able.
        EXIT
     5. The circuit is unredesignable.
```

"*Redesignability Analysis*" process. It should be mentioned that, with exhaustive search of the possible component assignments, we have identified 5 different assignments for the example circuit in Fig. 4-4.

4.3 Discussion

This chapter presents the redesignability analysis process for analog circuits. One of the most difficult aspects of redesign is the recognition of the functionality of existing implementation. The developed redesign process is comprised of two steps: (a) redesignability check; and (b) redesign solution. Based on circuit topology, a set of rules is developed to assign the edges in a graph to either tree or co-tree so that the associated MP-matrix with generated L-matrix is left invertible. Consequently, the I/O relations of the missing components can be reconstructed from the corresponding pseudo circuits. If a valid component assignment exists by Algorithm 4-1, the target circuit is redesignable. Otherwise, it is unredesignable with respect to the missing components. Note that the circuit may be possibly modified to become redesignable by decreasing the number of missing components. (we assume that the number of test points is fixed during the redesign process.) This can be achieved by combining some missing components as a bigger component, as shown in Fig. 4-6, referred to as missing circuit augmentation [56]. A combined missing component may include a number of missing components and/or known components. As the number of missing components is decreased, the number of test points may also decrease if IMTP- or VMTP component(s) are included by combined missing components.

Once circuit is found redesignable, the I/O relations of missing components can be derived by (2-10): the redesignable configuration is used to construct the L-matrix in (2-7)

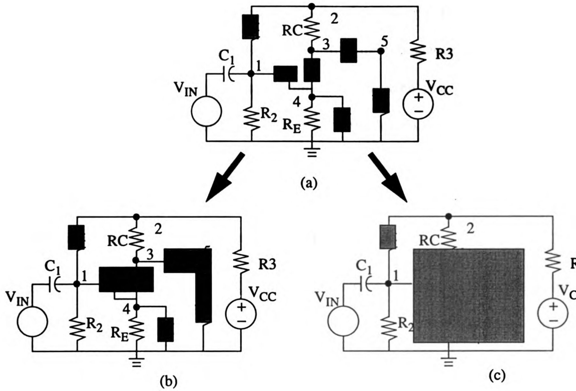


Fig. 4-6. Missing Component Compaction examples.

first. Since the corresponding MP-matrix is left-invertible, by the K-matrix for the pseudo circuit in (2-11), we can find a_i^2 and b_i^2 of the i-th missing component to construct its I/O relation, where both a_i^2 and b_i^2 are functions of the system input vector u, the system response vector y and characteristic functions of the *known parts*. In fact, the I/O relation of the i-th component can also be derived using *Sspice*, a symbolic version of *Spice* [57,58]. With appropriate selection of input signal vectors u and the generated system response vector y, the missing components can be constructed from the I/O relations.

As mentioned, the component assignment for a given graph (or circuit) is not unique. Which component assignment may provide the lowest cost for redesign is very important for further exploration. In addition, the complexity of the symbolically represented I/O relations [58] is an important issue. Selecting appropriate input vector u is also essential for generating redesign solution(s) [56].

Chapter 5

TEST POINTS SELECTION

This chapter presents an efficient test points selection (TPS) process for easily testable analog circuit design. The selections of both VMTPs and IMTPs are discussed. The selection of VMTPs can be formulated as to find a minimum set of fundamental loops that cover all nodes in the circuit graph, while the selection of IMTPs is to find a minimum number of cut-sets that cover all co-tree edges in the graph. Section 5.1 describes the problem statement of the test points selection. Section 5.2 presents the development of TPS algorithm which finds the minimum number of test points and identifies their locations from circuit topology. Section 5.3 introduces a heuristic algorithm to speed up the selection process for larger circuits. Finally, summary and concluding remarks are given in Section 5.4.

5.1 TPS Problem Statement

According to Rule 1 ~ Rule 3 in Chapter 3, the edges included in the fundamental loops associated with the selected VMTPs are assigned to tree edges, and the remaining co-tree edges must be included in the cut-sets associated with the selected IMTPs [31]. In other words, given a tree, the test points must be selected in such a way that the fundamental loops associated with the given VMTPs cover all tree edges, and the cut-sets associated with the selected IMTPs cover all co-tree edges. Therefore, the TPS problem for VMTPs, referred to as ***TPS_V problem***, can be stated as follows,

"to find a minimum number of fundamental loops that cover all tree edges"

Similarly, the TPS problem for IMTPs, referred to as TPS_I problem, is stated below,

TPS_I Problem:

"to find a minimum number of cut-sets that cover all co-tree edges"

Unfortunately, the number of possible trees in a graph is N_d^{Nd-2} for circuit graph with the number of nodes $N_d \geq 3$ [59]. Thus, exhaustively searching for all possible trees for test points selection problem is impractical particularly for large N_d.

For each VMTP, there exists a unique fundamental loop associated with it. The fundamental loop includes at least two nodes, i.e, the terminal nodes of the VMTP component. Since a fundamental loop visits the nodes it includes only once, the loop across these two terminal nodes is a valid path. Therefore, the TPS_V problem can be re-stated as follows,

TPS_V Problem:

"to find a minimum number of valid paths that construct a tree and cover all nodes in a graph"

It should be mentioned that (1) any node in a valid path cannot be visited twice or more. However, one node can be visited by two different valid paths as long as no close loops are formed; (2) the valid paths define a tree in the given circuit graph. (3) the selected VMTPs construct the E_2 matrix in Table 3-2 and must satisfy $r_2 \geq 1$. Therefore, the TPS_I problem then selects a minimum number of cut-sets that covers all co-tree edges. The tree edges associated with the selected cut-sets are selected as the IMTPs which construct the

E_l matrix satisfying $r_l \geq 1$. For an example circuit and given tree/co-tree partition shown in Fig. 5-1(a), Fig. 5-1(b) illustrates the two fundamental loops associated with V45 and V13. while Fig. 5-1(c) shows the cut-sets associated with IC1 and IR1. Since all circuit elements can be covered by either fundamental loops associated with VMTPs or cut-sets associated with IMTPs, this configuration is at least 1-diagnosable.

5.2 Algorithm Development

A *Hamilton path* is a valid path in a graph such that it contains all the nodes [60]. If there exists a Hamilton path in a graph, by the TPS_V problem, the circuit needs only one VMTP, which is across two terminals of the Hamilton path, to diagnose. A simple sufficient, but not necessary, condition for checking the existence of a Hamilton path in a graph can be found in [59]. In general, not all circuits contain the Hamilton paths.Thus, efficient algorithm are developed to resolve the TPS_V problem.

To reduce the complexity of searching valid paths in a graph, the original graph G is simplified as G', a simple graph, by merging the parallel edges and/or series edges as follows,

<u>Rule 5-1</u>. (a) if two or more edges are connected in parallel, they are merged as one in G'; and

(b) if two or more edges are connected in series, they are merged as one in G'.

Consider the graph G in Fig. 5-2(a), the parallel edges R2 and C1 in G are merged as the edge g_1 in G', as shown in Fig. 5-2(b). Similarly, the parallel edges RE and CE are merged as g_6, while the series edges C2 and RL are merged as g_7, where node "5" is elim-

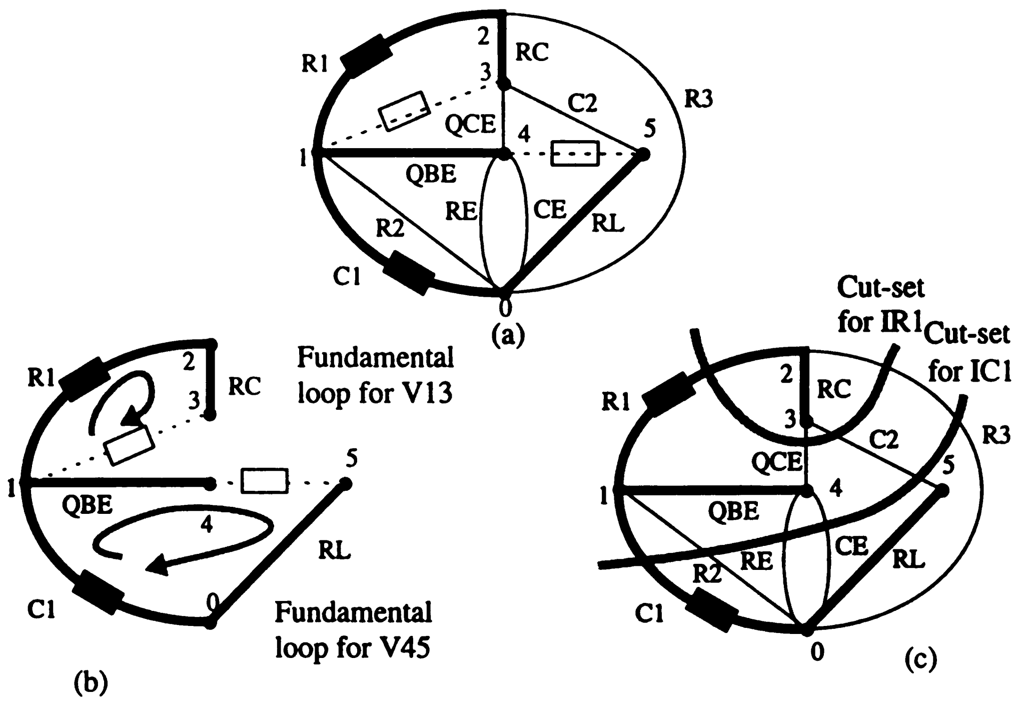


Fig. 5-1. Example Circuit: (a) Tree in A circuit Graph;
(b) Fundamental Loops; and (c) Cut-sets.

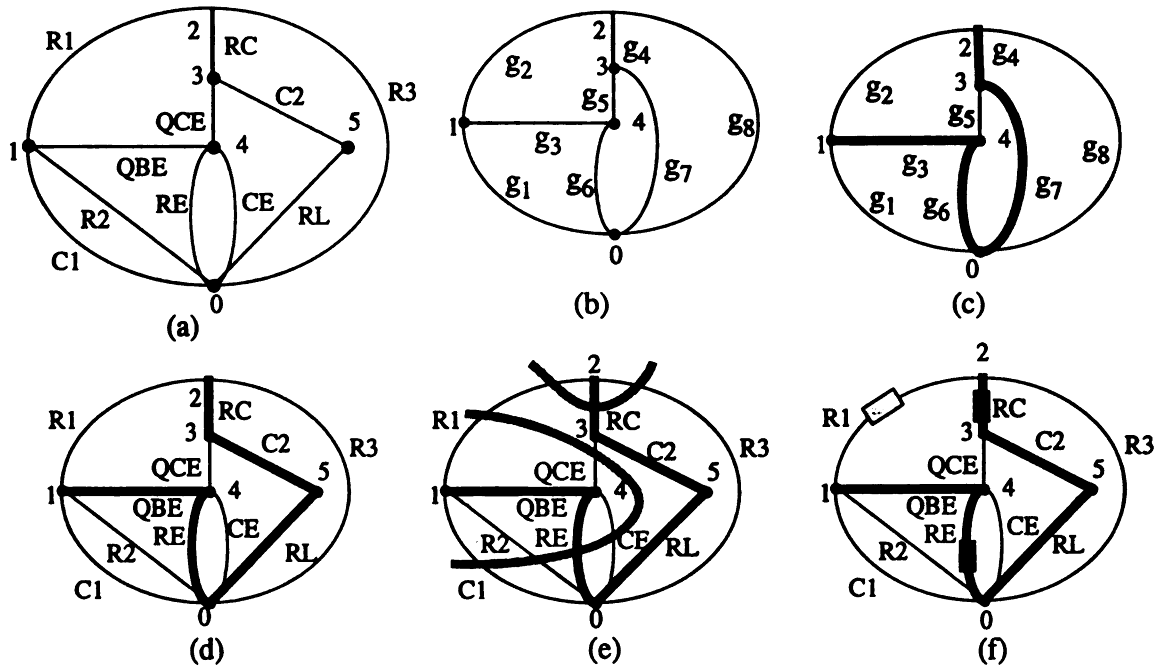


Fig. 5-2. Example for Test Point Selection Process: (a) Circuit Graph;
(b) Simplified Version; (c) Path Selection; (d) Selected F-loop(s);
(e) Cut-set Selection; and (f) Test Point Locations.

inated. As a result, the number of edges and nodes in G are 11 and 6, and they are reduced to 8 and 5 in G', respectively.

The next step is to search for the possible valid paths from the adjancecy matrix of the graph, where the adjacency matrix of G' is as follows:

$$\begin{array}{c|ccccc|}
\text{Node Index} & 0 & 1 & 2 & 3 & 4 \\
\hline
0 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 1 \\
2 & 1 & 1 & 0 & 1 & 0 \\
3 & 1 & 0 & 1 & 0 & 1 \\
4 & 1 & 1 & 0 & 1 & 0 \\
\hline
\end{array}$$

Consider the construction of all valid paths between node 1 to node 2. all valid paths between nodes 1 and 2 can be generated as follows, where the paths with "*" indicate the Hamilton paths:

1-0-2	1-4-0-2
1-0-3-2	* 1-4-0-3-2
* 1-0-4-3-2	* 1-4-3-0-2
1-2	1-4-3-2

Once the valid paths are constructed, a minimum set of valid paths that cover all nodes is selected, where the valid paths construct a tree in G'. The parallel and series edges in G are assigned based on the assignment in G' and the following rule.

<u>Rule 5-2</u>. (a) For parallel edges in G, if the corresponding edge in G' is assigned to tree, we assign any one of the parallel edges to tree and the remaining ones to co-tree; if the corresponding edge in G' is assigned to co-tree, we assign all parallel edges to co-tree.

(b) For series edges in G, let the terminal nodes of the series edges be α and β, respectively, if the corresponding edge in G' is assigned to tree, we assign all series edges to tree, if the corresponding edge in G' is assigned to co-tree, we assign the edge with α to co-tree, and the remaining ones to tree, where α is the terminal node of the valid path.

Consider a simple G' with a Hamilton path 1-4-0-3-2, as illustrated in Fig. 5-2(c), where the tree edges are g_3, g_6, g_7, and g_4, and the co-tree edges include g_1, g_2, g_5, and g_8. By Rule 5-2, the parallel edges C1 and R2 are assigned to co-tree, as shown in Fig. 5-2(d). Similarly, for tree edge g_6, the parallel edges RE and CE are assigned to tree and co-tree edges, respectively. Finally, for the tree edge g_7, the series edges RL and C2 are assigned to tree. This concludes that the circuit needs only one VMTP at the voltage across nodes 1 and 2, i.e., V12. In general, if there exist a minimum set of m_V's valid paths that include all nodes in G, then the circuit requires m_V's VMTPs.

For the TPS_I problem, it is to find a minimum number of cut-sets that cover all co-tree edges. The problem can be formulated as a minimum covering problem. More specifically, consider a matrix $D=(D_{ij})_{exk}$, where $e=N_d-1$ and $k=n-e$. The selected VMTPs define the tree edges $\{PT_1, PT_2, .., PT_e\}$ and the remaining edges in the graph are co-tree edges, say, $\{PC_1, PC_2, .., PC_k\}$. The j-th column of D is associated with PC_i and describes the fundamental loop associated with PC_i. Similarly, the j-th row means the cut-set associated with PT_j. Therefore, if the co-tree edge PT_j is included in the fundamental loop associated with PC_i, then D_{ji}="x". Otherwise, D_{ji}=" " (blank). For example, consider the tree in Fig. 5-2(d), where the tree edges are RC, C2, RL, RE, and QBE, and the co-tree edges include R1, R2, R3, C1, CE, QCE. Thus, the corresponding matrix D can be constructed as follows

$$\begin{array}{lcccccc} & R1 & R2 & R3 & C1 & CE & QCE \\ RC & x & & x & & & \\ C2 & x & & x & & & x \\ RL & x & & x & & & x \\ RE & x & x & & x & x & x \\ QBE & x & x & & x & & \end{array}$$

For example, there exists a fundamental loop associated with co-tree edge R1 including the tree edges RC, C2, RL, RE, and QBE. Thus, the x's in the first column indicate the tree edges that are included in the fundamental loop associated with R1. Therefore, the TPS_I problem is to find the minimum number of rows that cover all co-tree edges R1, R2, R3, C1, CE, and QCE. In other words, the TPS_I problem is to find the minimum number of rows that cover all co-tree edges, i.e., all PC_i's.

The minimum covering problem can be resolved using the prime implicants chart used in the Quine-McClusky method [60] for two-level logic minimization. In that chart, we first find the essential prime implicants, equivalent to *essential cut-sets* here. These are immediately apparent whenever there is a single "x" in any column. This means that there is a co-tree edge which is covered by one and only one cut-set. For example, the column with respect to CE has a single "x", where the cut-set associated with the tree edge RE is the one and only one which covers CE. The essential cut-sets must participate in the final cover. A reduced prime implicant chart can be obtained by deleting the column and row associated with the essential cut-set. Since the essential cut-sets usually cover additional co-tree edges. Any columns that have an "x" in a row associated with an essential cut-set are also deleted. In case that there exist cyclic prime implicant tables, the Patrick's method can be applied to find a set of minimum covers [60].

An essential cut-set associated with RE was found in the above chart and it covers the co-tree edges R1, R2, C1, and QCE. Only one column R3 is remained and can be covered by any one of the cut-sets associated with RC, C2, or RL. Fig. 5-2(e) shows the two cut-sets associated with RE and RC which cover all co-tree edges. Fig. 5-2(f) illustrates that the circuit can be diagnosable with a VMTP, V12, and two IMTPs, IRE and IRC. Based on this edge assignment, we obtain the following matrices E_1 and E_2:

	QBE	RE	RL	C2	RC	R1	R3	QCE	CE	R2	C1
IRC	0	0	0	0	0	1	1	0	0	0	0
IRE	0	0	0	0	0	0	0	1	1	1	1
VR1	1	1	1	1	1	0	0	0	0	0	0

It can be easily verified that both r_1 and r_2 are equal to 1. Algorithm 5-1 summarizes the above test point selection process.

Consider the simple graph G' with a Hamilton path 1-4-3-2-0, as shown in Fig. 5-3 (a). Since the edge g7 in G' is assigned to co-tree and node 0 is a terminal node of the valid path, by Rule 5-2, RL is assigned to tree and C2 to co-tree, as shown in Fig. 5-3(b). As a result, the edge V15 is chosen as the only VMTP, and the tree edges include QBE, QCE, RC, R3, and RL, while R1, R2, RE, C1, C2, and CE are in co-tree. The corresponding D matrix can be generated as follows.

	R1	R2	RE	C1	C2	CE
QBE	x	x		x		
QCE	x	x	x	x		x
RC	x	x	x	x	x	x
R3		x	x	x	x	x
RL					x	

Obviously, the cut-set associated with RC covers all co-tree edges and can be illustrated in Fig. 5-3(c). Thus, the circuit is diagnosable with two test points: the VMTP V15 and the

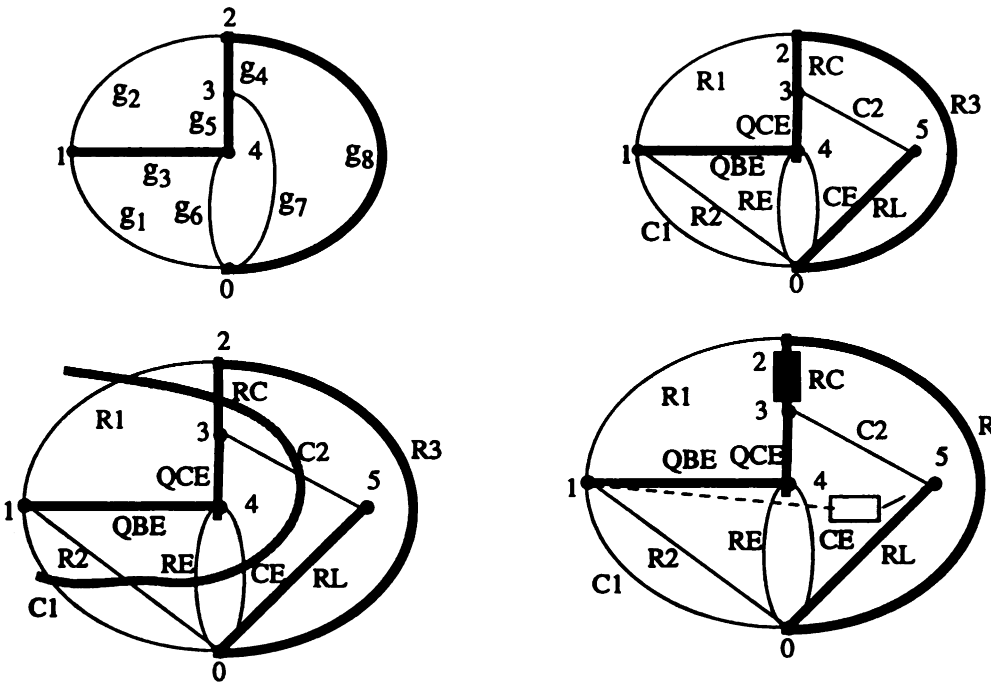


Fig. 5-3. Example for TPS Process: (a) Circuit Graph G' with Tree; (b) G with Tree and Cut-set; and (c) Test Point Locations.

IMTP IRC. Based on the edge assignment, we obtain the following matrices E_1 and E_2

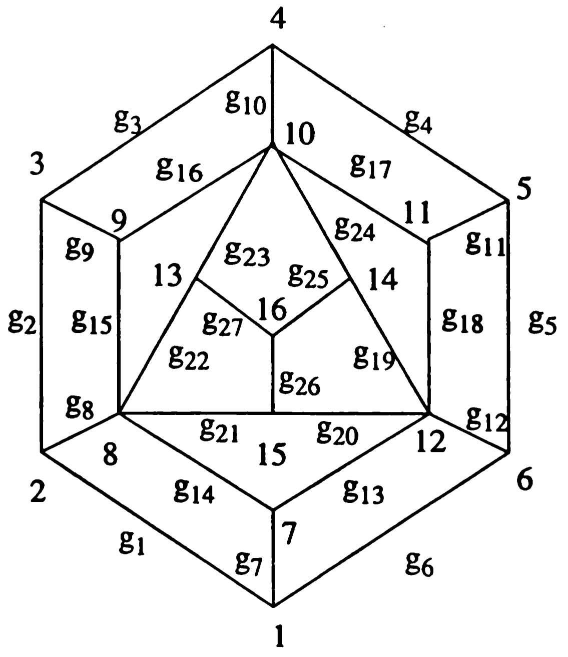
$$\begin{array}{c|ccccc|cccccc}
 & QBE & QCE & RC & R3 & RL & R1 & R2 & RE & C1 & C2 & CE \\
\hline
IRC & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
VR1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}$$

where both E_1 and E_2 have global column-rank of 1.

5.3 Improvement

As presented in the previous section, a simple adjacency matrix was used to find the possible valid paths. Among the possible valid paths, a minimum set is chosen to cover all nodes in the graph and to determine the VMTPs. However, finding all possible valid paths from the adjacency matrix of a graph for a reasonably large circuit may have very high searching complexity. Therefore, this chapter presents a simple heuristic algorithm, as summarized in Algorithm 5-2, for finding a near minimum set of valid paths that covers all nodes in the graph. For simplicity of this discussion, a challenge graph given in Fig. 5-4(a) is used to illustrate the stepwise procedure, and its adjacency matrix is given in Fig. 5-4(b), where the degree of a node is the number of its adjacent nodes. The node degree can be computed by summing up the number of 1's in the associated row.

The developed algorithm finds a valid path at a time. For the first valid path, the node degrees are first computed and the node with the highest degree is selected as the initial node. If more than one nodes with the highest degree exist, the node with the lower index has the higher priority to be selected. For simplicity, the node with highest priority is the one to be selected, NTBS node, for short. For example, the highest node degree in the adjacency matrix is 5, where nodes 8, 10, and 12 have the same highest node degree. Thus,



(a)

Node Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Node Degree
1	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	3
2	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	3
3	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	3
4	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	3
5	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	3
6	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	3
7	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	3
8	0	1	0	0	0	0	1	0	1	0	0	0	1	0	1	0	5
9	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	3
10	0	0	0	1	0	0	0	0	1	0	1	0	1	1	0	0	5
11	0	0	0	0	1	0	0	0	0	1	0	1	0	0	0	0	3
12	0	0	0	0	0	1	1	0	0	0	1	0	0	1	1	0	5
13	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	3
14	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	3
15	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	3
16	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	3

(b)

Fig. 5-4. Example: (a) Graph; and (b) Adjacency Matrix.

the NTBS node is the node 8. Let the initial node of the first valid path be denoted as node α, i.e., the valid path start with the node α. The next step is to find the NTBS node, β, of α. For example, node 8 has 5 adjacent nodes 2, 7, 9, 13, and 15 that have the same degree of 3. Thus, the NTBS node of α is node 2, i.e., β=2. It should be noted that the degree of a node must be updated if its adjacent nodes have been visited. Once β is selected, the next step is to find its NTBS node. The same process is repeated until no more NTBS node exists. Let γ be the node which does not have NTBS node, i.e., γ is the terminal node of the valid path. The process generates the following valid path, as shown in Fig. 5-5(a),

$$8\rightarrow2\rightarrow1\rightarrow6\rightarrow5\rightarrow4\rightarrow3\rightarrow9\rightarrow10\rightarrow11\rightarrow12\rightarrow7$$

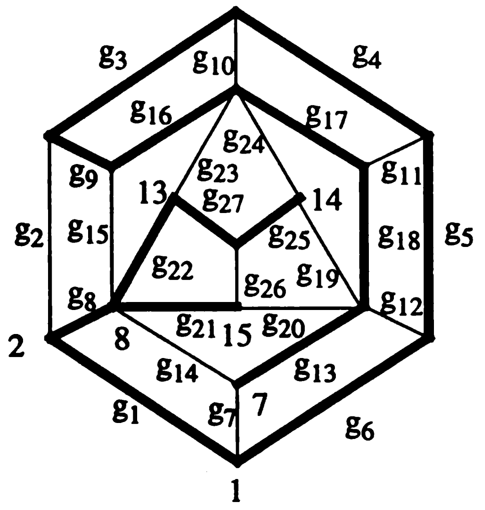
where γ=7. Once the terminal node is found, the valid path may be extended from the initial node α if its NTBS node exists. Let η be the NTBS node of α in the extended valid path. For example, at this moment, node 8 still have two adjacent nodes 13 and 15 which are not yet visited. Thus, node 13 is chosen as its NTBS node, i.e., η=13. The next step is to find the NTBS node of η, and the same process is repeated until no more NTBS node exists. Let ρ be the terminal node of the extended valid path. The extended path for the example is

$$8\rightarrow13\rightarrow16\rightarrow14$$

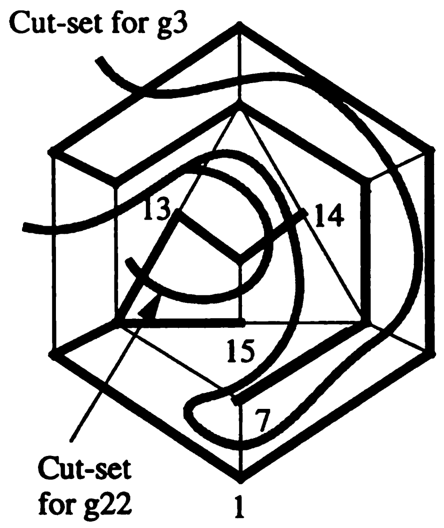
where ρ=14.

In summary, the first valid path is constructed from α to γ if no extension exists, and thus the VMTP is selected at V(α,γ). On the other hand, if extended path exists, then the valid path starts from ρ to γ and the VMTP is selected at V(ρ,γ). In the above example, the path is from node 14 to 7, i.e., the VMTP is either V(7,14) or V(14,7).

Once the first valid path is constructed, the remaining valid paths are generated from the unvisited nodes. Note that loops are not allowed in a tree. In order to avoid forming



(a) VMTPS: V(7,14), V(8,15)



(c) IMTP: g3 and g22

	2	7	10	11	12	14	15	19	20	23	24	26
1	x					x	x	x	x	x	x	
→ 3	x	x	x	x	x	x	x	x	x	x	x	
4	x	x		x	x	x	x	x	x	x	x	
5	x	x			x	x	x	x	x	x	x	
6	x	x				x	x	x	x	x	x	
8						x	x	x	x	x	x	
9		x	x	x	x	x	x	x	x	x	x	
13		x				x						
16		x	x	x	x	x		x	x	x	x	
17		x		x	x	x		x	x			
18		x			x	x		x	x			
21									x			x
→ 22								x		x	x	x
25								x			x	
27								x			x	x

(b)

Fig. 5-5. Example Graph: (a) Tree and Co-tree Edges; (b) Prime Implicant Chart; and (c) Cut-sets for g3 and g22.

Algorithm 5-1: Algorithm for Test Points Selections

```
Step 1. Generate simple graph G' using Rule 5-1.

     2. (Select VMTPs)
        2.1 Generate all valid paths
        2.2 Select a candidate set of valid paths that cover all nodes in G'.
        2.3 Assign edges in G using Rule 5-2.

     3. (Select IMTPs)
        3.1 Generate the prime implicant table or reduced table
        3.2 Find essential cut-sets
        3.3 If exist, GOTO Step 3.1;
        3.4 Apply Patrick's method if necessary.
        3.5 $m=m_I+m_V$;
            IF m is not minimum, GOTO Step 2.2.
        3.6 DONE.
```

loops, any nodes in a valid path cannot be revisited. However, a node may be visited by two different valid paths as long as no loops are formed. Therefore, the nodes included in the first valid path may be re-visited by the other valid paths.

In this algorithm, a flag, FLAG(i), is used to define the status of the i-th node. More specifically, FLAG(i)=0 if the i-th node has not yet been visited, FLAG(i)=1 if the i-th node is visited by the current path, and FLAG(i)=2 if the i-th node is visited by the previous paths. The flags are initialized as 0's before the first valid path is constructed. Once the first path is generated, FLAG(i) is set to a 2 if FLAG(i)=1. When the second path is constructed, the candidate nodes for the NTBS node are those adjacent nodes with FLAG(i)=0 or 2. However, the ones with FLAG(i)=0 has higher priority to be chosen than those with FLAG(i)=2 so that more unvisited nodes can be covered. Once the i-th node is chosen as the NTBS node, the flag, FLAG(i)=0 or 2, will be set to a 1. The node with FLAG(i)=1 is always disqualified as the candidate NTBS node. Thus, no loops will be formed in this selection process.

For the above example, after constructing the first valid path, there exists only one node which is unvisited, i.e., node 15. By Algorithm 5-2, its NTBS node is node 8 and no more unvisited nodes. The second valid path is from node 15 to node 8 and thus the second VMTP is V(8,15) or V(15,8). Since no unvisited node exists, the process terminated and a tree is constructed and shown in Fig. 5-5(a). Therefore, the tree and co-tree edges are listed as the indices shown in Fig. 5-5(b). Based on the row covering scheme in Algorithm 5-1, both rows 3 and 22 cover all co-tree edges. Therefore, the IMTPs are selected at g_3 and g_{22}. Fig. 5-5(c) illustrates the cut-sets for g_3 and g_{22}. This concludes that the circuit needs only four test points to make it easily testable.

Algorithm 5-2: Improved Algorithm for Test Points Selections

```
/*   Node(i): i-th node in the graph, i=1,2,..,n;
     Adj(x): the adjacent nodes of the node x;
     FLAG(i): the status of the i-th node;
     0- unvisited; 1- visited in current path; 2- visited in previous paths.
     V_path(v)[]: an array of nodes in the v-th valid path.
     NTBS() is a function for finding NTSB node.
     NTSB(0,v,j) - from all nodes with FLAG=0 for v-th path;
     NTSB(1,v,j) - from the adjacent nodes of V_Path(v)[j] with FLAG=0;
     It returns the NTSB nodes, or a 0 if no NTSB exist.
*/
Step 1. v = 1; FLAG=0; p=#{i | FLAG(i)=0}
     2. /* Construct the v-th valid path, first find the initial node */
        2.1  j=1; V_path(v)[1]=NTBS(0,v,j); FLAG(j)=1; p=p-1;
        2.2  k=NTBS(1,v,j);
        2.3  If (k≠0)
               {j=j+1; V_path(v)[j]=k; FLAG(k)=1; p=p-1;
              if (p≥0) GOTO Step 2.2;}
        2.4  r=j; /* terminal node */
     3. /* Extended path */
        3.1  k=NTBS(1,v,1);
        3.2  If (k≠0 & p≥0)
               { j=j+1; V_path(v)[j]=k; FLAG(k)=1; p=p-1;
               if (p≥0) GOTO Step 3.1;}
        3.3  q=j; /* terminal node */
        3.4  If (q=r) q=1;  /* No extended path */
        /* v-th valid-path from V_Path(v)[q] to V_path(v)[r] */
     4. FLAG(i)=2 if FLAG(i)=1 for all i; /* reset flags */
     5. If (p≥0) {v=v+1; GOTO Step 2;}
     6. Valid Paths, V_path(j), j=1,2,..,v-1, are generated.
```

It should be mentioned that the minimum covering problem is an NP-complete problem. This chapter developed an efficient heuristic algorithm to find a near optimum set of valid paths that cover all the nodes in a graph. Algorithm 5-2 is, in fact, a greedy algorithm. The first valid path is constructed in such a way that it is extended as longer as possible. Then, the remaining valid paths are generated from the unvisited nodes. In general, with the longest paths in the first attempt, the number of valid paths may not be the minimum, but it can be near minimum. Therefore, developing efficient algorithms which are better than the greedy algorithm can improve the quality of the TPS solutions.

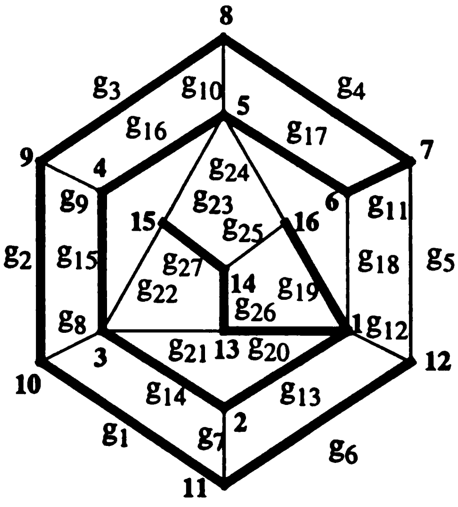
5.4 Discussion

This chapter formulates the problem of selecting test points for easily testable analog circuits as a minimum covering problem on a circuit graph. More specifically, the selection of VMTPs is the problem of finding a minimum set of valid paths that cover all nodes in the graph, while the selection of IMTPs is to find a minimum set of cut-sets that cover all co-tree edges. Two efficient algorithms, namely, Algorithms 5-1 and 5-2 have been developed to select the test points from circuit graph during the design phase[61]. The valid paths in Algorithm 5-1 are constructed in an exhaustive-like fashion, while those in Algorithm 5-2 are generated in a greedy approach. The emphasis of this chapter is placed on formulating the TPS problem and presenting some efficient and effective algorithms. However, the quality of test points selection can be improved in many different ways which lead to a very interesting research topic for further study.

More specifically, in Algorithm 5-2, the next node to visit is the adjacent node with the highest degree. If there are more than one nodes with the highest node degree, the one with the lower index has the higher priority to be selected. Consider the same circuit graph in Fig. 5-4(a). With the same edge index assignment, but different node index assignment, Algorithm 5-2 generates an alternative tree in Fig. 5-6(a). Thus, the circuit needs two VMTPs at $V_{12,15}$ and $V_{1,16}$, and three IMTPs at g_4, g_{14}, and g_{20}, as shown in Fig. 5-6(b). In other words, Algorithm 5-2 may generate different results for different index assignments. Therefore, how to generate an index set for Algorithm 5-2 such that the results is optimal is a very interesting problem to be further developed.

In addition, in Algorithm 5-2, the valid paths are selected from the simple graph G'. All edges in G' are weighted equally when the node degrees are calculated. It should be mentioned that, in the simple graph G', if an edge is obtained by merging all parallel edges, it should have less weight. Because we assign the edge in G' to tree, by Rule 5-2(a), we assign any one of the parallel edges to tree and the remaining ones to co-tree. Therefore, we need the edge as an IMTP, as illustrated in Fig. 5-1. With the above different weight, one may obtain much less number of test points, as illustrated in Fig. 5-3. This TPS process has been integrated into yjr test system, ADTS, as a "*Test Points Suggestions*" process.

The goal of the TPS problem is to find a minimum set of test points. In other words, the problem is to minimize the total number of VMTPs and IMTPs. To simplify the TPS process, we first minimize the number of VMTPs to construct a tree and then minimize the number of IMTPs. In some cases, the tree constructed from the minimum set of VMTPs may not be the tree which has the minimum set of IMTPs in the circuit graph. Thus, one may re-formulate the TPS problem so that both VMTPs and IMTPs can be minimized simultaneously. This is also an important problem to improve the quality of the developed TPS solutions.



(a)

VMTPS: V(12,15), V(1,16)

		5	7	8	9	10	12	18	21	22	23	24	25
	1	x	x				x						
	2	x	x	x			x						
	3	x	x	x	x		x						
→	4	x	x	x	x	x	x						
	6	x					x						
	11		x	x	x	x							
	13						x	x		x	x	x	x
→	14		x				x	x		x	x	x	x
	15		x	x			x	x			x	x	
	16		x	x	x		x	x			x	x	
	17		x	x	x	x	x	x					
	19								x			x	
→	20								x	x	x	x	x
	26								x	x	x		
	27									x	x		

(b)

Fig. 5-6. Example Graph: (a) Tree and Co-tree Edges; and (b) Row Coverings.

Chapter 6

TEST PROGRAM GENERATION

HDL-A is an analog version of hardware description language which is suitable for structural and behavioral descriptions and simulations of digital, analog, and mixed-signal circuits and systems. This chapter presents the test program generation process for fault diagnosis of analog/mixed-signal circuits. The generation core, Automatic Test Program Generator (ATPRG) [33], works under the Mentor Graphics (MG) design system environment, where the UUTs are modeled in HDL-A language and simulated in *accusim* [48]. The ATPRG takes the nominal system data and test specifications of a UUT, provided by the test system designer, to generate test programs building up a database for fault diagnosis. To automate the generation process, the AMPLE (Advanced Multi-Purpose LanguagE) in MG design system is used to define and execute the generation process automatically.

In the next section, the building blocks of the test program generation process and their corresponding HDL-A descriptions are presented. Section 6.2 describes the HDL-A modeled pseudo circuits. Section 6.3 presents the automatic test program generation process. Finally, discussion is given in Section 6.4.

6.1 HDL-A Modeled Control Sources

The HDL-A is primarily a language to be used to develop analog and mixed-signal models [34]. The language allows the modeling of digital logic, either stand alone or embedded in analog models, using a subset of VHDL. The connections of a model carry two non-directional quantities, i.e., voltage and current, rather than the single directional ***signal*** associated with a ***port*** in VHDL. The behavior of lumped analog devices is described by expressing the relations between voltages and currents at all connections of the device. Such relations can be described with linear, or non-linear, algebraic equations or even differential equations. Each equation relates to the voltage across and the current through a component and imposes the constraint that the specified relation be satisfied by the analog solver at all times. Since HDL-A models are analog behavioral models, both time domain and frequency domain apply.

Consider the example in Eqn. (2-15), assume that the component b3 is a resistor R_X, i.e., $V_{b3}=V_{RX}$. A voltage controlled voltage source can be used to describe V_{RX}. In HDL-A, V_{RX} can be modeled, as shown in Fig. 6-1(a), by adding ***COUPLING*** to the ENTITY declaration so that we can take the voltage, V_{RX}, without affecting the resistor. In other words, the ***COUPLING*** statement is used to pass analog behaviors of elements, i.e. voltage or current, into and between pseudo circuits. On the other hand, V_{RX} also can be modeled using control source shown in Fig. 6-1(b). For fault diagnosis applications, the latter implementation is suggested because the former requires adding ***COUPLING*** to the ENTITY declaration of the component model in the library. Fig. 6-2(a) and Fig. 6-2(b) show the HDL-A descriptions of controlled current and voltage sources which are used for modeling

```
ENTITY resistor IS
  GENERIC (rval: analog);
  COUPLING (irl,vrl: analog);
  PIN (pos,neg: electrical);
END ENTITY resistor;

ARCHITECTURE res OF resistor IS
BEGIN
   RELATION
     PROCEDURAL FOR INIT =>
        rval:=1.0e3;
     PROCEDURAL FOR ac, dc, transient =>
        irl:=[pos,neg].v/rval;
        pos.i%=irl; neg.i%=-irl;
        vrl:=[pos,neg].v;
   END RELATION;
END ARCHITECTURE res;
```

(a)

```
ENTITY vcvs IS
  COUPLING(vx:analog);
  PIN (pos,neg:electrical);
END ENTITY vcvs;

ARCHITECTURE vx OF vcvs IS
BEGIN
   RELATION
     PROCEDURAL FOR ac, dc, transient =>
       [pos,neg].v%:=vx;
   END RELATION;
END ARCHITECTURE vx;
```

(b)

Fig. 6-1. Controlled Source Models: (a) Direct Control; and (b) Indirect Control.

```
-- CCCS HDL-A MODEL
ENTITY cccs IS
  COUPLING(vx,ix:analog);
  PIN (pos,neg:electrical);
END ENTITY cccs;

ARCHITECTURE icon OF cccs IS
BEGIN
   RELATION
     PROCEDURAL FOR ac, dc, transient =>
        pos.i%=ix;
        neg.i%=-ix;
        vx:=[pos,neg].v;
   END RELATION;
END ARCHITECTURE icon;
```

(a)

```
-- VCVS HDL-A MODEL
ENTITY vcvs IS
  COUPLING(vx,ix:analog);
  PIN (pos,neg:electrical);
END ENTITY vcvs;

ARCHITECTURE vcon OF vcvs IS
BEGIN
   RELATION
     PROCEDURAL FOR ac, dc,  transient =>
        [pos,neg].v%:=vx;
        ix:=pos.i;
   END RELATION;
END ARCHITECTURE vcon;
```

(b)

Fig. 6-2. Controlled Source Models: (a) Current Source; and (b) Voltage Source.

the pseudo circuits. By using controlled source, vendor provided component library can be applied directly without any modifications.

6.2 HDL-A Modeled Pseudo Circuits

The connection matrix K in Eqn. (2-10) of a pseudo circuit is used to describe a test program for the component subdivision. A test program is generated by the following three steps:

Step 1. Compute a^1 and b^1 from (2-10) and (2-8a), or (2-9a);

Step 2. Calculate both a_i^2 and $\bar{b}_i^2$ from (2-10b) and from (2-8b) or (2-9b).

Step 3. Compute b_i^2 from (2-10b) and find the difference $(\bar{b}_i^2 - b_i^2)$.

Consider the connection matrix of the pseudo circuit in Fig. 3-1, with

$$a^1=(a11,a12,a13,a14,a15,a16,a17)=(IR1,VC2,IQ1,VCE,VR3,VR2,VRE)$$

and $$b^1=(b11,b12,b13,b14,b15,b16,b17)=(VR1,IC2,VQ1,ICE,IR3,IR2,IRE).$$

Therefore, we have

$$a11=y1;\ a12=b4-b5+b6-b7+y2;\ a13=b6-y1+y2;\ a14=b2-y3-y4;$$

$$a15=-b1-b2+b3-u2+y3+y4;\ a16=b2-b3-y3-y4;\ a17=b2-y3-y4.$$

The corresponding HDL-A description is given in Fig. 6-3(a), and its ENTITY is referred to as *B1-block.* Note that ***COUPLING*** allows the B1-block to use their values of u1, y1, y2, y3 and y4, the input and test signals, obtained from the source description. The parameter in the ***COUPLING*** declaration, is either voltage source controlled by the voltage across, or current source controlled by the current flow through associated components.

```
-- HDLA MODELED PSEUDO CIRCUIT B1-BLOCK
-- FOR TESTEE GROUP COMPONENTS: C1, RC, Q2, RL

ENTITY B1_BLOCK IS
   COUPLING (a11,a12,a13,a14,a15,a16,
             a17,y1,y2,y3,y4,u1,u2:analog);
   PIN (gnd:electrical);
END ENTITY B1_BLOCK;

ARCHITECTURE P1 OF B1_BLOCK IS
BEGIN
   RELATION
      PROCEDURE FOR DC, AC, TRANSIENT=>
      -- GET a1 FROM b1, u, y AND THEN COUPLING OUT
         a11:=+y1;                   --R1
         a12:=+b4-b5+b6+b7+y2;       --C2
         a13:=+b6-y1+y2;             --Q1
         a14:=+b2-y3-y4;             --CE
         a15:=-b1-b2+b3-u2+y3+y4;    --R3
         a16:=+b2-b3-y3-y4;          --R2
         a17:=+b2-y3-y4;             --RE
   END RELATION;
END ARCHITECTURE P1;
```

(a)

```
-- HDLA MODELED PSEUDO CIRCUIT A2-BLOCK
-- FOR TESTEE GROUP COMPONENTS: C1, RC, Q2, RL

ENTITY A2_BLOCK IS
   COUPLING (b11,b12,b13,b14,b15,b16,b17,a21,
             a22,a23,a24,y1,y2,y3,y4,u1,u2:analog);
   PIN (gnd:electrical);
END ENTITY A2_BLOCK;

ARCHITECTURE P2 OF A2_BLOCK IS
BEGIN
   RELATION
      PROCEDURE FOR DC, AC, TRANSIENT=>
      -- GET a2 FROM b1, u, y AND THEN COUPLING OUT
         a21:=+y2;                   --C1
         a22:=-b5+y1;                --RC
         a23:=+b2+u2-y4;             --Q2
         a24:=+b2-u2-y3;             --RL
   END RELATION;
END ARCHITECTURE P2;
```

(b)

```
-- HDLA MODELED PSEUDO CIRCUIT DIFF-BLOCK
-- FOR TESTEE GROUP COMPONENTS: C1, RC, Q2, RL

ENTITY DIFF_BLOCK IS
  COUPLING (b11,b12,b13,b14,b15,b16,b17,bb21
           ,bb22,bb23,bb24,y1,y2,y3,y4,u1,u2:analog);
  PIN (gnd:electrical);
END ENTITY DIFF_BLOCK;

ARCHITECTURE P3 OF DIFF_BLOCK IS
BEGIN
  RELATION
    PROCEDURE FOR DC, AC, TRANSIENT=>
    -- OUTPUT DIFFERENCE OF b2 & b^2
      b21.diff%:=bb21-(+b2-b3-u1-y3-y4);    --C1
      b22.diff%:=bb22-(-b1-b2+b3-u2+y4);    --RC
      b23.diff%:=bb23-(-b4-b6-b7+y1-y2);    --Q2
      b24.diff%:=bb24-(-b4+b5-b6-b7-y2);    --RL
  END RELATION;
END ARCHITECTURE P3;
```

(c)

Fig. 6-3. HDL-A Modeled Pseudo Circuit: (a) B1_block; (b) A2_block; and (c) DIFF_block.

Once the values of a^1=(a11 ~ a17) in B1-block are obtained, the HDL-A description for the components in the *Tester Group* will derive the values of b^1=(b11 ~ b17) which will be used in *Step 2*. Again, from the K-matrix, a^2=(a21,a22,a23,a24)=(VC1,VRC,IQ2,VRL). where

a21=y2; a22=b5-y1; a23=b2+u2-y4; and a24=b2-u2-y3.

The corresponding HDL-A description is illustrated in Fig. 6-3(b), and its ENTITY is referred to as *A2-block*. Once the values of a^2 are obtained, the HDL-A description of the components in the *Testee Group* will derive the values of $\overline{b}^2$=(bb1,bb2,bb3,bb4) which will be used in *Step 3*. The values of b^2=(b21,b22,b23,b24) are first computed and then compared with $\overline{b}^2$. The differences can be expressed as follows,

b21diff=bb21-b21=bb21-(b2-b3-u1-y3-y4);

b22diff=bb22-b22=bb22-(-b1-b2+b3-u2+y4);

b23diff=bb23-b23=bb23-(-b4-b6-b7+y1-y2);

b24diff=bb24-b24=bb24-(-b4+b5-b6-b71-y2).

Fig. 6-3(c) shows the HDL-A description of *DIFF-block* for *Step 3*.

In summary, the HDL-A description of a pseudo circuit includes three ENTITY blocks, B1-block, A2-block, and DIFF-block. As illustrated in Fig. 6-3, the detailed structures of the three ENTITY blocks are described by the architectures p1, p2, and p3.

6.3 Test Program Generation Process

In the test program generation process, the generated test programs should be verified and validated. A simulation-based verification process using HDL-A simulator, *AccuSim*, has been developed to check if the test programs are properly generated and to verify whether the test programs are capable of identifying fault(s) effectively. After the generated test programs are completely verified, they will be validated by emulating the UUT with injected faults. With the measured test responses obtained from the faulty UUT through a virtual instrument, the test programs will be validated if they can properly identify fault(s).

Fig. 6-4 illustrates the simulation-based verification and validation process. A test program includes the compiled HDL-A codes for B1-block, A2-block, and DIFF-block. The test program for the example circuit takes four input pins, u1, y1, y2,y3 and y4, and produces three output pins, b21diff, b22diff, b23diff, and b24diff. In this development, the test programs will receive the simulated test responses from the UUT or faulty UUT during the verification process, and they will takes measured test responses during the validation process. Thus, each test program require an interface, as shown in Fig. 6-4, and its HDL-A code is described in Fig. 6-5(a).

In the Mentor Graphics design system, where the HDL-A codes are compiled and executed, a circuit description can be either provided by the test system designer with a netlist, or imported from the schematic diagram generated by *Design Architect*. This implementation takes the latter approach. During the verification process, the responses measured at various test points in a UUT are modeled as controlled sources. Fig. 6-5(b) illustrates the HDL-A description of the test points. Their measured voltages/currents are obtained by *"Coupling"* the corresponding component measurements in the schematic diagram. For simplicity, all current measurements are converted to voltage measurements. For example, "y1pin.v%= iC1*1.0;" in Fig. 6-5(b), means that the voltage across the pin "y1pin" is defined as that across the 1Ω-resistor connected in series with the current "iC1".

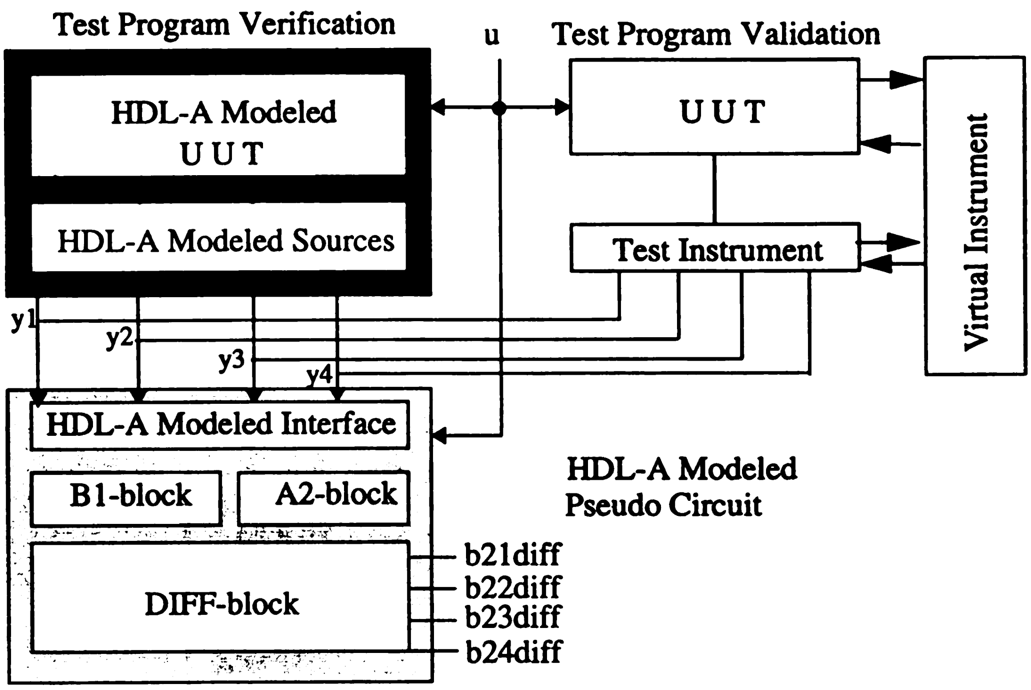


Fig. 6-4. HDL-A Modeled Test Program Process.

```
-- Interface Description
ENTITY interface IS
  COUPLING(u1,y1,y2.y3,y4:analog);
  PIN (u1pin,y1pin,y2pin,
        y3pin,y4pin,gnd:electrical);
END ENTITY interface;

ARCHITECTURE itrface OF interface IS
BEGIN
   RELATION
    PROCEDURAL FOR ac, dc, transient =>
        u1pin:=[u1,gnd].v;
        y1pin:=[y1,gnd].v;
        y2pin:=[y2,gnd].v;
        y3pin:=[y3,gnd].v;
        y4pin:=[y4,gnd].v;
   END RELATION;
END ARCHITECTURE itrface;
```

(a)

```
-- Source Description
ENTITY sources IS
  COUPLING(iC1,iR1,vX13,vX45:analog);
  PIN (y1pin,y2pin,y3pin,y4pin:electrical);
END ENTITY sources;

ARCHITECTURE uy OF sources IS
BEGIN
   RELATION
    PROCEDURAL FOR ac, dc, transient =>
    -- output y values to interface pins
        y1pin.v%=iC1*1.0;
        y2pin.v%=iR1*1.0;
        y3pin.v%=vX13;
        y3pin.v%=vX45;
   END RELATION;
END ARCHITECTURE uy;
```

(b)

Fig. 6-5. HDL-A Descriptions: (a) for Interface; and (b) for Sources.

6.4 Discussion

HDL-A is an analog version of hardware description language which is suitable for the structural, behavioral descriptions and simulations of digital, analog, and mixed-signal electronic circuits and systems. It is able to support any design methodology and is technology independent. This chapter presents the development of the ATPRG system, an automatic test program generator using HDL-A model, for fault diagnosis of analog circuits. The components in the system model, CCM, can be analog discrete components, digital logic gates, integrated circuits, or subsystem. With the hardware description languages, VHDL for digital and HDL-A for analog, the developed ATPRG system can be applied for mixed-signal integrated circuits [30,33]. The generation core of ATPRG has been integrated into the test system, ADTS, as a "*HDL-A Test Program Generation*" process. To keep pace with the ever increasing demands on higher productivity and shorter development cycles, Labview, a virtual test environment, as shown in Fig. 6-4, is used to derive quality-assured test program from high-level specifications [30]. The reliability and quality of the test program can be increased using simulation-based test program verification tools.

Chapter 7

CONCLUSION

Mixed-signal ICs have become the main-stream solutions for the applications such as portable data systems, wireless communication, and multimedia. Due to the increasing complexity and circuit density, manufacturers confront the problem of long testing cycle by using the conventional external functional test performed on ATE (Automatic Test Equipment). As a result, this not only reduces the testing confidence, but also increases the cost directly. Therefore, it is necessary to develop efficient test/diagnosis strategies that will not only help finding the causes of common failure mechanism, but may also help to resolve testing complexity problem by providing informations of proper locations of test points. This thesis study presents the development of an automated diagnostic test system that includes several features such as diagnosability analysis, redesignability analysis, test points selections, and HDL-A test program generations. All these processes have been integrated as a complete test system, the *Automated Diagnostic Test System*.

7.1 Summary and Contributions

The framework of the test system, as illustrated in Fig. 1-1, includes diagnosability analysis, redesignability analysis, and diagnosability design methodologies. Chapter 3 presents the diagnosability analysis process. Given a circuit topology and test points, Algorithm 3-1 assigns the components to tree/co-tree edges of the graph representing the circuit topology. Based on the all possible component assignments, candidate diagnosable configurations are generated and the one with the maximum diagnosability is chosen. The configurations having poor diagnosability are also identified.

The diagnosability of the portion having poor diagnosability can be enhanced by redesigning the portion. An efficient redesign analysis process presented in Chapter 4 checks if the redesign is feasible. Two major steps were developed: (a) redesignability check; and (b) redesign solution. A set of component assignment rules, similar to those in diagnosability analysis, is applied to check if the associated MP-matrix is left invertible for deriving the I/O relationships of those components. If no such MP-matrices exist, the circuit is not redesignable. Otherwise, the developed I/O relations are used to reconstruct the components. Several redesign methodologies are also discussed to speed up the process.

In Chapter 5, the test points selection problem is formulated as a minimum covering problem on a circuit graph. The selection of VMTPs is the problem of finding a minimum set of valid paths that cover all nodes in the graph, while the selection of IMTPs is to find a minimum set of cut-sets that cover all co-tree edges. Two efficient algorithms were developed to define the number of test points required and identify their locations. In Chapter 6, an automatic test program generator, ATPRG, were developed. The circuit model process using HDL-A was presented, including the development of the HDL-A modeled control sources and HDL-A modeled pseudo circuits.

The major contributions in this thesis study can be categorized as follows:

(a) Developing an efficient diagnosability analysis process to evaluate the diagnosability of a circuit from the given circuit topology and selected test points. The process can also identify the portion with poor diagnosability;

(b) Developing an effective redesignability analysis process to check if a portion with poor diagnosability can be redesigned;

(c) Developing an efficient test points section process to select the number of test points required and their locations to meet the desired diagnosability;

(d) Developing the first automatic test program generator with HDL-A so that the developed fault diagnosis process can be used for large analog/mixed-signal integrated circuits; and

(e) Integrating the developed process to an automated diagnostic test system.

7.2 Future Work

It is believed that the complexity of analog/mixed-signal ICs will continuously increase. Further enhancing circuit testability and diagnosability becomes very important tasks and developing efficient and effective diagnosable design methodology becomes a great demand for IC industrials.

This thesis study has effectively demonstrate the feasibility of the developed analysis processes and design methodologies. However, for practical large analog/mixed-signal integrated circuits, further quality improvements are necessary and can be achieved in the following different ways. As the circuit complexity increases, searching the valid diagnosable configurations in the diagnosability analysis process becomes very complex. Thus, developing efficient searching algorithms will promote the developed diagnosability analysis process to be applied for practically large analog/mixed-signal circuits. Similarly, the searching complexity for redesignability analysis, test points section, and test program generation increases as the circuit complexity increases.

For the redesignability analysis process, the emphasis of this study was placed on the development of redesignability check. Once a circuit is determined to be redesignable, the I/O relations of the missing components are derived. Efficiently deriving the I/O relations becomes an important tasks for the redesignability analysis and should be investigated further. How to select the appropriate test signals to derive the I/O relations is also an important tasks to work.

The automatic test program generation process can effectively generate the test programs for fault diagnosis of analog/mixed-signal circuits. It is efficient and effective because the use of hardware description languages. Once the generated test programs are used for fault diagnosis, the test programs are simulated. The developed system ADTS is capable of simulating any reasonably large linear analog circuits. For nonlinear circuits, however, the fault simulations become the bottleneck of the fault diagnosis process. Developing an efficient simulator for simulating such test programs efficiently is also an important research topic for further investigation.

REFERENCES

1. A. Mastsuzawa, "Low-Voltage and Low-Power Circuit Design for Mixed Analog/ Digital Systems in Portable Equipment", IEEE Journal of Solid-State Circuits, Vol. 29, No.4, pp. 470-480, April 1994.
2. C.L. Wey, "Mixed-Signal Testing -- a Review", (invited), Proc. of the IEEE International Conference on Electronics, Circuits, and Systems, Rodos, Greece, pp. 1064-1067, 1996.
3. C.L. Wey, "Mixed-Signal Fault Models and Design For Test", (Tutorial Course), IEEE Asian Test Symposium, Hsinchu, Taiwan, 1996.
4. C.-P. Wang, "Efficient Testability Design Methodologies for Analog/Mixed-Signal Integrated Circuits", Ph.D. Dissertation, Department of Electrical Engineering, Michigan State University, 1997.
5. W.-H. Huang, J.A. Resh, and C.L. Wey, "On Synthesis of Manufacturable and Testable Analog Integrated Circuits," Proc. of the 41th Midwest Symposium on Circuits and Systems, Notre Dame, IN, 1998.
6. R. Spence, *Tolerance Design of Electronic Circuits*, Addison-Wesley Publishing Company, 1988.
7. M. Li and L. Milor, "Computing Parametric Yield Adaptively Using Linear Modelis," Proc. Design Automation Conference, pp. 831-836, 1996.
8. C.-P. Wang and C.L. Wey, "Development of Hierarchical Testability Design Methodologies for Mixed-Signal/Analog Integrated Circuits", Proc. of the International Conference on Computer Design (ICCD 97), pp. 468-474, 1997.
9. R.-W. Liu, *"Testing and Diagnosis of Analog Circuits and Systems"*, North-Holland, 1991.
10. J.W. Bandler and A.E. Salama, "Fault Diagnosis of Analog Circuits", IEEE Proceedings, pp. 1279-1325, August 1985.
11. R. Saeks, "Criteria For Analog Fault Diagnosis", Proc. of the European Conference on Circuit Theory and Design, pp. 75-78, 1981.
12 L. Rapisarda and R. DeCarlo, "Analog Multi-frequency Fault Diagnosis", IEEE Trans. on Circuits and Systems, Vol.CAS-30, No.4, pp. 223-234, April 1983.
13. Z.F. Huang, C.-S. Lin, and R.-W. Liu, "Node-fault Diagnosis and a Design of Testability", IEEE Trans. on Circuits and Systems, Vol. CAS-30, pp. 257-265, May 1983.

14. M. Slamani and B. Kaminska, "Analog Circuit Fault Diagnosis Based on Sensitivity Computation and Functional Test", IEEE Design & Test of Computers, pp. 30-39, 1992.

15. A. Charoenrook and M. Soma, "Fault Diagnosis of Flash ADC Using DNL Test", Proc. of the International Test Conference, pp. 680-689, 1993.

16. C.L. Wey and R. Saeks, "On the Implementation of An Analog ATPG: The Linear Case", IEEE Trans. on Instrumentation and Measurement, Vol. IM-34, No.3, pp. 277-284, Spetember, 1985.

17. C.L. Wey and R. Saeks, "On the Implementation of An Analog ATPG: The Nonlinear Case", IEEE Trans. on Instrumentation and Measurement, Vol. IM-37, No.2, pp. 252-258, June 1988.

18. C.L. Wey, "A Searching Approach Self-testing Algorithm For Analog Fault Diagnosis", in *Testing and Diagnosis of Analog Circuits and Systems*, edited by R.-W. Liu, Chapter 6, pp. 147-186, North-Holland, 1991.

19. H. Dai and T.M. Souders, "Time-Domain Testing Strategies and Fault Diagnosis For Analog Systems", IEEE Trans. on Instrumentation and Measurement, Vol. IM-39, No.1, pp. 157-162, February 1990.

20. F. Lin, J. Markee and B. Rado, "Design and Test of Mixed Signal Circuits: A discrete-Event Approach", Proc. of the 32nd Conference on Decision Control, pp. 217-222, 1993.

21. T.H. Morrin, "Mixed-Mode Simulation for Time-Domain Fault Analysis", Proc. of the International Test Conference, pp. 231-241, 1989.

22. A. Balivada, J. Chen and J.A. Abraham, "Analog Testing with Time Response Parameters", IEEE Design & Test of Computers, pp. 18-25, 1996.

23. Z. You and E. Sanchez-S, "Analog System-level Fault Diagnosis Based on a Symbolic Method in the Frequency Domain", IEEE Tran. on Instrumentation and Measurement, Vol. IM-44, No.1, pp. 28-35, February 1995.

24. M. Slamani and B. Kaminska, "Multi-frequency Testability Analysis for Analog Circuits", Proc. of the 12th IEEE VLSI Test Symposium, pp. 54-59, 1994.

25. M. Slamani and B. Kaminska, "Fault Obervability Analysis of Analog Circuits in Frequency Domain", Proc. of the IEEE Trans. on Circuits and Systems., Part.II, Vol. 43, No.2, pp. 134-139, February 1996.

26. S. S. Somayajula, E. Sanchez-S and J. Pineda de Gyvez, "Analog Fault Diagnosis Based on Ramping Power Supply Current Signature Clusters", IEEE Trans. on Circuits and Systems. Part. II, Vol.43, No. 10, pp. 703-712, October 1996.

27. Z. Wang, G. Gielen and W. Sansen, "Testing of Analog Integrated Circuits Based on Power-Supply Current Monitoring and Discrimination Analysis", Proc. of the 3rd Asian Test Symposium, pp. 126-131, 1996.

28. A. Walker, P.K. Lala and W.E. Alexander, "Analogue Fault Diagnosis of Active

Networks Via Bias Modulation", Electronics Letters, Vol. 27, No. 24, pp. 2279-2281, 1991.

29. A. Walker, W.E. Alexander and P.K. Lala, "Fault Diagnosis in Analog Circuits Using Element Modulation", IEEE Design & Test of Computers Vol. 9, No. 1, pp. 19-29, 1992.

30. W.-H. Huang and C.L. Wey, "Development of Automated Diagnostic Test System for Mixed-Signal/Analog Integrated Circuits", Proc. of the 40th Midwest Symposium on Circuits and Systems, Davis, pp. 1434-1437, CA, 1997.

31. W.-H. Huang and C.L. Wey, "Diagnosability Analysis For Mixed-signal Integrated Circuits", International Journal of Circuit Theory and Applications, Wiley, NY, Vol. 26, December 1998.

32. C.L. Wey and W.-H. Huang, "Redesignability Check of Analog Circuits with Incomplete Implementation Information," Accepted to appear in IEEE Trans. on Circuit and System, Part I, Fundamental Theory and Applications.

33. W.-H. Huang and C.L. Wey, "Development of HDL-A Modeled Test Programs for Fault Diagnosis of Analog/Mixed-Signal Circuits", Proc. of the 3rd IEEE International Mixed Signal Testing Workshop, Seattle, WA, pp. 3-14, 1997.

34. R.C.-J. Shi, "*AHDL Tutorial*", presented in 3rd IEEE International Mixed Signal Testing Workshop, Seattle, WA, 1997.

35. J. Bhasker, "*A VHDL Premier*", Prentice Hall, 1992.

36. W.-H. Huang, "ADTS: An Automatic Diagnostic Test System for Analog Circuits - User's Manual and Programmer's Guide", Department of Electrical Engineering, Michigan State University, 1998.

37. P. Duhamel and J.-C. Rault, "Automatic Test Generation Techniques for Analog Circuits and Systems: A Review", IEEE Trans. on Circuits and Systems. Vol. CAS-26, No.7, pp. 411-439, February 1979.

38. C.L. Wey, "Parallel Processing for Analog Fault Diagnosis", International Journal of Circuit Theory and Applications, Wiley, NY, Vol.16, No.3. pp. 303-316, July 1988.

39. C.L. Wey, "Design of Testability for Analog Fault Diagnosis", International Journal of Circuit Theory and Application, Vol.15, No.2, pp. 123-142, April 1987.

40. C.-K. Ho, P.R. Shepherd, W. Tenten and R. Kainer, "Improvements in Analogue Fault Diagnosis Techniques", Proc. of the 2nd IEEE International Mixed Signal Testing Workshop, Quebec, Canada, pp. 81-97, 1996.

41. C.L. Wey, "Built-In Self-Test (BIST) Structure for Analog Circuits Fault Diagnosis," IEEE Trans. on Instrumentation and Measurement. Vol. IM-39, No.2, pp. 517-521, June 1990.

42. C.L. Wey, "Alternative Built-In Self-Test Structure (BIST) for Analog Circuit Fault Diagnosis," Electronics Letters. Vol.27, No.18, pp. 1627-1628, August 1991.

43. C.L. Wey and S. Krishnan, "Built-In Self-Test (BIST) Structures for Analog Circuit Fault Diagnosis with Current Test Data," IEEE Trans. on Instrumentation and Measure-

ment, Vol. IM-41, No.4, pp. 535-539, August 1992.

44. M. Soma, "Structure and Concepts for Current-based Analog Scan," Proc. of the Custom Integrated-Circuits Conference, pp. 517-520, 1995.

45. C.L. Wey, "Built-in Self-Test (BIST) Design of Current-mode Algorithmic A/D Converter" IEEE Trans. on Instrumentation and Measurement, Vol.IM-46, No.3, pp. 667-671, June 1997.

46. M.S. Nejad, L. Sebaa, A. Ladick and F. Kuo, "Analog Built-in Self-test", Proc. of the 7th Annual IEEE International ASIC Conference and Exhibit, pp. 407-411, 1994.

47. A. Chatterjee, B.C. Kim and N. Nagi, "DC Built-in Self-test for Linear Analog Circuits", IEEE Design & Test of Computers, Vol. 13, No. 2, pp. 26-33, 1996.

48. "AccuSim II HDL-A/DEV User and Reference Manual", Software Version 8.4_1, Mentor Graphics Corporation, 1994.

49. R.A. Saleh, D.L. Rhodes, E. Christen, and B.A.A. Antao, "Analog Hardware Description Languages", Proc. of the IEEE Custom Integrated Circuits Conference, pp. 349-356, 1994.

50. R.C.-J. Shi, "HDL-A Design Objectives and Rationale", Technical Report, Analogy, Inc., 1994.

51. M. Pierzchala and B. Rodanski, "Algebraic Method for Calculating Symbolic Network Functions of Large Electronic Networks", Proc. of the 37th Midwest Symposium on Circuits and Systems, 1994.

52. T.M. Souder and G.N. Stenbakken, "A Comprehensive Approach For Modeling and Testing Analog and Mixed-Signal Devices", Proc. of the International Test Conference, pp. 169-176, 1990.

53. C.L. Wey, "Development of Redesign Process for Digital VLSI Systems," Proc. 40th Midwest Symposium on Circuits and Systems, Davis, CA, pp. 177-180, 1997.

54. F.P. Preparata, G. Metze, and R.T. Chien, "On the Connection Assignment Problem of Diagnosable Systems," IEEE Trans. on Electronic Computers, Vol. EC-16, pp. 448-454, 1967.

55. C.S. Lin, Z.F. Huang, and R.-W. Liu, "Fault Diagnosis of Linear Analog Networks: A Theory and Its Application," Proc. of the IEEE International Symposiums on Circuits and Systems, pp. 1090-1093, 1983.

56. M.A. Khalil, "Efficient Redesign Process for Digital VLSI Circuits", Master Thesis, Department of Electrical Engineering, Michigan State University, 1998.

57. G.M. Wierzba, *Sspice User Manual*, East Lansing, MI: Michigan State University Instructional Media Center, 1991.

58. S-M. Chang and G.M. Wierzba, "Circuit Level Decomposition of Networks with Nullor for Symbolic Analysis," IEEE Transactions on Circuits and Systems, Vol. CAS-41, pp. 699-711, November 1994.

59. M.N.S. Swamy and K. Thulasiraman, *Graph, Networks, and Algorithms*, Wiley, 1981.

60. E.J. McCluskey, *Logic Design Principles with empahsis on Testable Semicustom Circuits*, Prentice-Hall, Englewood Cliffs, New Jersey, 1986.

61. W.-H. Huang and C.L. Wey, "Test Point Selection Process and Diagnosability Analysis for Analog Integrated Circuits," IEEE International Concerence on Computer Design: VLSI in Computers & Processors (ICCD 98), pp. 582-587, 1998.