Fault diagnosis and design for testability applied to analogue integrated circuits

Ho, Chung Kin

Award date:
1998

Awarding institution:
University of Bath

Link to publication

Fault Diagnosis and Design for Testability applied to Analogue Integrated Circuits

submitted by Chung Kin Ho

for the degree of PhD of the University of Bath, March, 1998

Chung Kin Ho [illegible] 13-3-98

Acknowledgement

I would like to express my sincere thanks and gratitude to my supervisor Dr Peter Shepherd for his guidance on this research project and encouragement during my time at the University of Bath. I am also in debt to Dr Wilfried Tenten, Dr Robert Kainer and Mr Friedemann Eberhardt of Robert Bosch GmbH for their fruitful discussions which resulted in the novel procedures developed in this research work, for their regular supply of literature surveys, and for their permission to use a Bosch IC as a test vehicle to aid the development of the diagnosis program. My time spent working with Friedemann when he came to Bath for his PhD work, especially on the evaluation of the diagnosis program, has proved to be beneficial to both of our PhD projects, and the intellectual discussions and sometimes argument between us were stimulating and constructive. I look forward to the opportunity to continue the development of the diagnosis program to a usable form with the Bosch colleagues in future.

Thanks also to the MEng Student, Graeme Anderson, who evaluated the primitive version of the diagnosis program in 1997, and to Dr Hartmut Schwab of Robert Bosch GmbH, who gave me guidance on documentation of the diagnosis program. Special thanks to Mr Martin Maclaren for his help in setting up the Sun Unix Workstation which is used for the development of the diagnosis program for this project, in installing the many software tools on the Sun which are needed for this research work and for his advice on Unix in general. I am also grateful to our technician Mr Dave Hatten for making up the serial printer cable for the Sun.

Contents

List of Figures in Chapters

List of Figures in Sections of Chapters

List of Tables

List of Acronyms and Symbols

CUT	Circuit Under Test
PCB	Printed Circuit Board
BIST	Build In Self Test
JTAG	Joint Test Action Group
PGA	Pin Grid Array
PLCC	Plastic Leaded Chip Carrier
ASIC	Application Specific Integrated Circuit
SBT	Simulation Before Test
SAT	Simulation After Test
KCL	Kirchhoff's Current Law
KVL	Kirchhoff's Voltage Law
ST	Self Test
DFT	Design For Testability
OTGP	Optimal Tree Generation Procedure
FDA	Fault Diagnostic Algorithm
TPS	Test Point Selection
CCM	Component Connection Model
NoIndS	Number of Independent Sources
HOTG	Hierarchical Optimal Tree Generation
CMOS	Complementary Metal Oxide Semiconductor
BiCMOS	Bipolar and CMOS
BILBO	Built-in logic block observers
PRBS	Pseudo Random Binary Sequence

$\mathbf{A}$	Node Incidence Matrix of Example 2
$\mathbf{Q_0}, \mathbf{A_0}$	Node Incidence Matrix of Example 3
$\mathbf{Q_1}$	Upper Triangular Step Matrix of Example 3
$\mathbf{Q_2}$	Upper Triangular Matrix of Example 3
$\mathbf{A_1}$	Tree/cotree Partitioned Node Incidence Matrix of Example 3
$\mathbf{a}$	Input Variable Vector
$\mathbf{b}$	Output Variable Vector
$\mathbf{L_{11}}$	Connection Matrix
$\mathbf{L_{12}}$	Connection Matrix
$\mathbf{u}$	Stimulus Vector
$\mathbf{i_t}$	Tree Partition Branch Current Vector (excluding independent sources)
$\mathbf{v_{ct}}$	Cotree Partition Branch Voltage Vector (excluding independent sources)
$\mathbf{v_t}$	Tree Partition Branch Voltage Vector (excluding independent sources)
$\mathbf{i_{ct}}$	Cotree Partition Branch Current Vector (excluding independent sources)
$\mathbf{J_t}$	Current Vector for Independent Current Sources
$\mathbf{E_{ct}}$	Voltage Vector for Independent Voltage Sources
$\mathbf{i_T}$	Tree Partition Branch Current Vector
$\mathbf{v_{CT}}$	Cotree Partition Branch Voltage Vector
$\mathbf{v_T}$	Tree Partition Branch Voltage Vector
$\mathbf{i_{CT}}$	Cotree Partition Branch Current Vector
$\mathbf{i_{vs}}$	Current Vector for Independent Voltage Sources
$\mathbf{v_{vs}}$	Voltage Vector for Independent Voltage Sources

$\mathbf{v}_{is}$	Voltage Vector for Independent Current Sources
$\mathbf{i}_{is}$	Current Vector for Independent Current Sources
$\mathbf{D}$	Fundamental Matrix for KCL Equations
$\mathbf{A}_{T}$	Tree Partition Node Incidence Matrix for Example 2
$\mathbf{A}_{CT}$	Cotree Partition Node Incidence Matrix for Example 2
$\mathbf{A}_{1T}$	Tree Partition Node Incidence Matrix for Example 3
$\mathbf{A}_{1CT}$	Cotree Partition Node Incidence Matrix for Example 3
$\mathbf{v}_{node}$	Node Voltage Vector
m	Number of Test Points
$\mathbf{y}$	Test Point Vector
$\mathbf{L}_{21}$	Test Point Matrix
$\mathbf{L}_{22}$	Test Point Matrix
t	Diagnosis Depth
n	Number of components in the CUT
$n`$	*n-noIndS*
N	Set of indices for n circuit components (N={0, 1,...., n-1})
$\mathbf{A}_{c}$	Node Incidence Matrix for the Current Graph
$\mathbf{A}_{v}$	Node Incidence Matrix for the Voltage Graph
$\mathbf{i}$	Branch Current Vector
$\mathbf{v}$	Branch Voltage Vector
$\mathbf{B}$	Loop Matrix
$\mathbf{D}_{B}$	Fundamental Matrix for the KVL Equations

Abstract

This thesis reviews various fault diagnosis methods and proposes a novel Fault Diagnosis Algorithm (FDA) which builds on the Self Test Algorithm with the Component Connection Model formulation. With the adoption of the hierarchical approach, an improved Hierarchical FDA is proposed which enables efficient use of computing resources and hence its use in fault diagnosis of practical large scale analogue circuits is a possibility after further development. The evaluation of the effectiveness of the Hierarchical FDA with several test circuits has been reported and as a result of this evaluation, enhancement to the algorithm has been recommended and further areas of research to make it become a usable tool in testing have also been identified. The Hierarchical FDA consists of pre-test processing, which is divided into two phases, and sets of iterative test cycles. Each set of iterative test cycles is associated with one of the optimal testing trees the Hierarchical FDA provides.

Phase one of pre-test processing includes translating a netlist of the Circuit Under Test (CUT) to an abstract description represented by a directed linear graph and pre-processing of all the hierarchical components in the CUT to prepare a list of optimal trees for testing. Phase two of pre-test processing begins upon completion of phase one or re-starts due to an incomplete diagnosis with the previous optimal testing tree used. The processes in this phase are: deriving a matrix representation (connection equation) of the abstract description with the Hierarchical Optimal Tree Generation Procedure, and the application of the Test Point Selection Scheme to choose a set of test points (branch voltages and currents of circuit components) and test node voltages.

Each cycle of the iterative test procedure is comprised of partitioning the circuit components into testee and tester groups, rearrangement of the connection and test point matrices due to the component partition, a sequence of checks and computations to obtain a set of test results in the form of a normalized percentage error vector, using global or local decision thresholds obtained by appropriate fault models or from experience to convert the test results to a digital error vector, and applying decision algorithms on the digital error vector to choose fault-free testees to be included as testers in the next test cycle.

"The important thing in science is not so much to obtain new facts as to discover new ways of thinking about them", William Bragg.

1. Introduction

This chapter introduces the reader to the subject of fault diagnosis in general by explaining some test-related issues and the meaning of fault diagnosis, and reviewing its development from the past to the present state-of-the-art. Parallel to this review is a brief review on IC development leading to the test challenges of mixed signal ICs. Then, the industrial impetus behind the author's work is described before justifying the fault diagnosis approach taken in this research and outlining the layout of this thesis.

1.1. Fault Diagnosis, Test and Related Issues

Much in the same way as the common word "chips" is interchangeable in meaning with the formal phrase "Integrated Circuits (ICs)", the word "test", or "testing", is synonymous with "Fault Diagnosis". Despite the fact that ICs are either monolithic or hybrid[1], ICs usually refer to monolithic ICs as hybrid ICs are now a rare breed except for specialized applications such as microwave ICs. In the context of electronic circuits, fault diagnosis concerns the detection and identification of faults in a Circuit Under Test (CUT).

From the functional testing viewpoint, fault detection is a process which verifies that at least one of the functional modules of the CUT falls outside its design specifications, which have various parameters such as gain, bandwidth, signal to noise ratio, etc. What these design parameters are, depend on what the functional module is. Each of these design parameters has a nominal value and tolerance box which specifies the accepted deviation from the nominal value. A functional module is considered working or within its design specifications provided that all of its design parameters are within their tolerance box.

On the other hand, structural testing is defect-oriented. Fault detection is a process which looks for the presence of defects in at least one of the functional modules of the CUT. Defects in ICs are the result of variation and imperfection of their manufacturing processes and external factors such as radiation, electrical overstress (EOS) and electrostatic discharge (ESD). There are numerous mechanisms behind these defects and they are well documented in [2]. Examples of these mechanisms are gate oxide breakdown due to EOS or ESD, ionic contamination from the manufacturing process, dislocations caused by diffusion processes and electromigration. These defects manifest themselves as various degrees of degradation in the expected electrical, logical or functional characteristic of a CUT. Their manifestations are mapped onto faults which are represented as circuit conditions that can be modelled electrically, logically or functionally. The most frequently used fault models for digital circuits are the stuck-at faults[7], which model the defect manifestations at the logic gate level. Other fault models are the stuck-open and stuck-short faults[3] which model the defect manifestations at the transistor switch level and are used for CMOS digital circuits. Similar to the stuck-short fault is the bridging fault[4], which is a short circuit between two signal lines in the metalization layer. For analogue circuits, the fault models used are catastrophic (hard), parameter deviation (soft), and behaviour faults[17]. A functional module is considered faulty if the fault described by the fault model adopted, i.e. the manifestations of faults, is detectable by means of measured or simulated electrical quantities of the CUT. Depending on the realization of the CUT, a functional module can be a group of circuit components at the discrete component level, a group of chips on a PCB at the board level or a sub-circuit of the CUT at the monolithic level.

Fault identification determines which of the functional modules are faulty or not working, and if it is necessary, which of the components within a faulty module are faulty or not working. Parallel to the dual meanings of the acronym "ICs", the meaning of testing is sometimes confined to fault detection only. This is the case in production test (go/no-go testing), whereas in development and field tests, testing refers to both fault detection and identification[1].

The purpose of testing is to ensure the quality and reliability of whatever products which use ICs as their components. Test quality is usually measured by the term "fault coverage", which is the ratio of the number of detectable modelled faults to the number of all possible modelled faults, in structural testing. Central to the testability of a circuit are the concepts of controllability and observability[1]: controllability is the ability to set the CUT into a specific state in order to test for a particular fault at a particular node. On the other hand, observability is the capability to interpret the test results for a particular fault in the CUT at its outputs or special purpose test ports where currents and/or voltages can be measured. However, testing of ICs is a complex issue whose complexity is best appreciated with a brief review of the development of fault diagnosis and ICs.

1.2. Development of Fault Diagnosis

Historically speaking, research on fault diagnosis had its focus on digital rather than analogue circuits. This partly contributes to the fact that the development of analogue fault diagnosis and test equipment lags so far behind its digital counterpart. Research and development in digital fault diagnosis began in the mid 1960s due to the availability of large scale computers and made available the first available test program a decade[5] later. After more than three decades of development and research, there are now proven digital automatic test pattern generation (ATPG) and design for testability/test (DFT) techniques. For small combinational circuits with small number of primary inputs, application of all possible test vectors guarantees near-maximum fault coverage yet their test cost is the lowest among other ATPG techniques as its test generation (computations required to generate the test vectors) and application (time required to apply the test vectors) costs are extremely low. This kind of exhaustive testing can be applied to more complex but not large circuits so long as the partitioning of the CUT into small sub-circuits is possible. Another inexpensive way to test combinational circuits with a low level of logic and gate fan-ins is the Random Testing technique[6]. This technique employs a random number generator to derive test vectors in succession. Each test vector generated can be found to detect an additional fault or not with the use of a simulator and the fault models of the CUT.

For the testing of large combinational circuits, ATPG algorithms are available and based on the oldest but universally accepted stuck-at fault models proposed by Eldred[7]. A pioneering example in algorithmic ATPG is the Roth's D-algorithm[8][9]. It is based on the concepts of controllability and observability (Chapter 1.1): the test vector for a stuck-at fault at a particular node is obtained by establishing a path to activate the fault and to propagate the fault effect to an observable output node by manipulating input values along the path sensitized. An alternative to the D-algorithm is the Boolean Difference technique[10]. It derives all the possible test vectors for both types of stuck-at faults at a specific node by performing the exclusive-or Boolean operation on the logic function corresponding to an observable output node, and a function derived from this output function. This Boolean operation nature results in slow test vector generation and hence its use in fault detection for large combinational circuits with complex output functions is precluded, whereas the D-algorithm is more effective in test vector generation in terms of computation speed and the fact that it does not generate redundant test vectors. However, the single stuck-at fault test set produced by the D-algorithm and other algorithms alike, will only achieve high fault coverage (over 99%) for multiple stuck-at faults if the number of observable circuit output nodes is above two[11], whereas the Boolean

Difference technique will be able to detect multiple faults after some extension[1]. Therefore, the Boolean Difference method can be used to test small partitions of a large combinational circuit, where multiple faults are more likely to occur, and the rest of the circuit is tested with a single stuck-at fault algorithm such as the D-algorithm.

The D-algorithm will not always work for a CUT if any signal branches out as inputs to more than one logic gate, and as the outputs from these logic gates propagate down the logic levels, they re-join together as inputs to the same logic gate. This reconvergent fanout of a signal will sometimes lead to multiple paths being sensitized in the CUT, which prevent the propagation of a fault to an observable output node. Goel[12] proposed the path-oriented decision making (PODEM) algorithm to solve this problem of reconvergence fanout. PODEM is similar to the D-algorithm but it will retry a step to reverse an incorrect decision and by doing so, it effectively disables other sensitized paths that interfere with the propagation of the fault to the observable output node. There are also other ATPG algorithms[13][14] which improved on PODEM. The D-algorithm, PODEM and their successor algorithms alike, are now the basis of many ATPG systems incorporated into industry standard CAD tools. From the D-algorithm[8] to its successor[14], digital ATPG procedures have advanced to a very mature stage in two decades and will continue to evolve as new challenges arise. A review of this subject and an abundant source of literature references can be found in [15].

Design for testability/test (DFT), as its name implies, is the use of design-oriented measures to enhance testability of a circuit from the outset of the design stage so that on the one hand IC quality and reliability are ensured, but on the other hand test cost is kept to a minimum to make a product economically viable. The need to test sequential circuits, which consist of fed-back state variables (memory elements) that are not easily controllable and observable, is the major drive for the development of structured DFT approaches such as the scan path design and Built-In-Self-Test (BIST). Opposite to the structured approach is the ad hoc approach which uses good engineering practice, such as additional design rules at the schematic or layout level, to improve testability. This is a tailor-made approach as the exact measure taken depends on the individual CUT. Scan path design enhances the controllability and observability of a sequential circuit by incorporating an on-chip 2-to-1 multiplexer (MUX) with every flip-flop element in the sequential circuit. In scan mode, the MUXs configure the flip-flops into a serial shift register to form a scan path, and break the connections between the flip-flops and the input logic block of the sequential circuit. This latter action of the MUXs turns the sequential circuit into a combinational circuit which can be tested with the combinational ATPG techniques. During the scan mode, the outputs and inputs of the flip-flops become the respective pseudoprimary inputs and outputs which are controllable and observable via the scan path. However, test vector generation, control of the MUXs, test result observation and evaluation are still not done on-chip for the scan path design. BIST enhances the testability of a sequential circuit even further by incorporating any or all of the other elements of the test process on-chip at a price. Chip area is inevitably increased by implementing BIST and often a balance between the complexity of the BIST and its cost has to be struck. The elements of the test process that are most likely incorporated into an IC directly are PRBS test vector generation, signature analysis and built-in logic block observers (BILBO).

Scan path design had been extended to the so called boundary scan design for the testing of board level products to circumvent the worsening problem of test probing ICs on a PCB with increasingly dense packing density due to device minimization and advances in process, packages and PCB manufacturing technologies. The essence of the boundary scan design is to associate a SRL (shift register latch), which performs the dual role of a multiplexer and flip-flop with a master-slave flip-flop configuration and a two-phase clocking scheme, with every functional pin of an IC which is going to be mounted on a PCB. The overhead of an IC with boundary scan design is four additional pins for the scan path, the control and clock lines of the scan SRL. These SRLs can then be configured into a shift register around the IC boundary to form a boundary scan path. A PCB with boundary scan design has on board scan path connections so that the scan paths within all the ICs on the board can be connected to form an overall serial data path with a board scan input and output.

As a PCB may consist of ICs from more than one manufacturer, a common standard for boundary scan implementation on the board and system levels was needed. The Joint Test Action Group (JTAG) was formed in 1985 by leading semiconductor manufacturers to serve the purpose of devising future test concepts and to work on this common standard. There is now a testability standard JTAG/IEEE 1149.1 for boundary scan implementation, which sets out the requirements for testing complex digital systems in a hierarchical manner that covers component, board and system levels. This standard specifies a 4-wire, serial interface test bus which is used to interconnect components, boards and systems that are conformed to its test structure so that various modes of testing can be activated by software control. One semiconductor manufacturer[16] has even developed a dedicated software tool, a controller card and a range of IEEE 1149.1 chips to support the hierarchical testing of digital circuits.

Another useful DFT technique is the Iddq test, which uses the measurement of quiescent supply current of an IC to detect faults based on short circuit effects where higher than usual supply current will result. Its main attraction is its relative low test cost and it is usually used to detect stuck-short[3] and bridging[4] faults, which are difficult to detect with the stuck-at fault based ATPG, in CMOS digital ICs.

The advance in digital fault diagnosis has been driven by the demand to test reliably digital circuits with integration density and clock speed increasing annually. This advance is made possible by the use of the simple model of stuck-at-one and stuck-at-zero faults. However, there is no analogue fault model parallel to the digital one because an analogue signal is continuous in nature. Furthermore, analogue functions are diverse and circuit performance can be measured in many ways. Example of analogue variables are circuit response time (rise and fall time), bandwidth, gain, temperature variations, offset voltage and leakage current. Like component values, some of these variables have tolerance bands which complicate testing. In addition, the accuracy of analogue measurements needs to be considered in testing. This lack of a universal analogue fault model is another reason for the development of digital testing techniques and equipment to be so far ahead of its analogue counterpart.

The perennial problem in analogue testing is the accessability of circuit nodes for measurements. At chip level testing the designer has to consider testing in the design stage and to make available some test points on the pins of the chip for test-probing. This inevitably adds some overhead in the cost of the chip. Test-probing nodes on a circuit board is made difficult or sometimes impossible by the use of multilayer board, modern chip package (PLCC and PGA) and surface mount technology. This problem is exacerbated by increased device minimisation, in particular the advent of the Mixed-signal Application Specific Integrated Circuits (ASICs) which are brought about by the recent improvements in process technology that has made possible reliable CMOS and BiCMOS realisations of both digital and analogue circuit elements on the same chip. As the process technology improves and the minimum feature size of an IC decreases, the technological disparity between test equipment and new analogue and mixed-signal ICs becomes significant to a point which prompts semiconductor manufacturers to update their test equipment and hence the cost of testing increases while process technology advances. The demand to keep the cost of testing down and to make analogue testing a manageable task has forced researchers to focus their efforts on analogue fault diagnosis since the mid 1970's[5] and has produced a variety of fault diagnosis methods for analogue circuits.

Since the advent of the mixed analogue and digital ICs, market demand on them has been on the increase, so that their test issue is gaining importance and attracting renewed interest from researchers on mixed signal testing. This renewed interest has been so abundant that the IEEE Computer Society Test Technology Technical Committee has sponsored an annual workshop focusing on issues related to the testing of mixed signal ICs since 1995. With such workshops, the testing of mixed signal ICs have truly established itself as a discipline which has been driven by three fundamental market and technology forces identified by Soma[17]: testing for the simple fault model of digital circuits is no longer sufficient to guarantee satisfactory test results as digital VLSI ICs start to show analogue behaviour, especially during signal transitions, due to a combination of ever higher operating frequencies and lower operating power. The second driving force originates from the fast expanding telecommunication market, in which every system is almost mixed signal in nature. These mixed signal circuits, such as converters and transceivers, are also characterized by high operating frequencies and low operating power (e.g. mobile phone). Their high and increasing test cost has resulted in intensive industrial efforts to find new cost effective test techniques. The last driving force comes from the increasing presence of systems integrating sensors and ICs. These systems are used in automotive electronics, medical equipment and consumer electronics, etc. They employ converters as the interface between the sensors and the rest of the systems. Both the sensors and converters are very difficult to test and their test cost are high. This again leads to intensive industrial efforts to find alternative test techniques to reduce test cost. In response to these driving forces, researchers from academia and industry have proposed a wide range of techniques to meet the mixed signal test challenges (Chapter 1.3). For example, some techniques borrow ideas from established digital testing methods, whereas others concentrate on the bottleneck of mixed signal test, i.e. the testing of analogue circuits for which there are no commercially available test tools[18].

To address the problem of testing mixed signal ICs at the board level, work on a new mixed signal test bus standard, the IEEE 1149.4, an extension for the 1149.1 boundary scan standard, has been on-going since 1991 and would have passed its first ballot phase in

1997. Its salient features and cost benefits are reported by Sunter[19]. Mixed signal ICs which are 1149.4 compliant will have two pins for supplying the analogue stimulus and observing the associated response (AT1 and AT2) and four pins (TCK, TMS, TDI and TDO) for the 1149.1 test access port interface and associated support circuitry.

The accepted standard test practice leading to very good fault isolation and high product reliability for mixed signal circuits is the divide and conquer approach[20]. This approach partitions a mixed signal circuit into analogue, memory and logic blocks so that each block can be tested with its own specific test methods. Central to this approach is that the CUT, whether at the monolithic or board level, has a test mode to allow direct access to the inputs and outputs of each block via boundary scan and additional analogue signal buses[21]. After each block has been tested with its own specific tests, the complete test of the CUT requires two additional steps which are the testing of the interconnects between blocks [22]and the testing of the combined function of all blocks and interconnects (system test). Even though the interconnect tests only apply to mixed signal circuits at the board level, high test cost and the fact that a complete system test is usually not possible are the disadvantages of this partitioned methodology. Due to this reason, other methods have been proposed to either supplement this partitioned approach or replace it with a unified test for mixed signal circuits.

Examples of the supplementary tests are the transient response technique (TRT) and Iddq testing. The essence of TRT[23] is the use of PRBS stimulus to excite a CUT and the capturing of the resulting transient response for off-line auto-correlation (on fault free response) and cross-correlation (on fault free and faulty response) on which their comparison is used to detect faults. Iddq testing is well established for digital CMOS ICs[24]. Its use in the testing of mixed signal ICs began with DC supply current monitoring for CMOS macros, such as opamps and comparators, with hard faults. However, Iddq testing for mixed signal circuits has developed to include a number of variants[25] which are based on the monitoring of AC and

DC supply currents. Some of these techniques have combined supply current monitoring with the use of Fast Fourier Transform (FFT) or an artificial neutral network (ANN) while others use the cross-correlation of the supply current and output test responses for fault detection.

There have been three attempts to propose a unified approach for mixed signal test. Reisig and DeCarlo[26] proposed a time domain method based on the combined formulation of Component Connection Model (CCM) and Self Test (ST) Algorithm a decade ago. This was followed by Chang and Sheu[27][28] who proposed a frequency domain method based on the same formulation nearly a decade after Reisig's publication. Both methods are characterized by extremely complicated mathematics and Chang's approach is more promising as it results in a lower order matrix equation and the required number of test points are reduced. In addition, Chang's approach has avoided some of the problems in Reisig's approach. However, neither approach tackles the problem of tolerance masking and investigate the effect of changes in the precision of test points on the diagnosing power of their algorithms. Recently, Lin[29] proposed a method based on a newly developed theory of discrete event systems but the three examples used in his paper are too simple and are either pure digital (a 2-gate circuit with 3 inputs and 1 output) or analogue (a voltage divider and a 5-resistor network).

At the moment, there is no analogue ATPG advanced enough to parallel the highly developed digital ATPG such as PODEM[12] and its successor[13]. However, there are a wide diversity of developing analogue ATPG techniques whose descriptions are not necessary to set the scene for reading the author's work. Soma[30] has reviewed this issue recently and grouped 14 different types of ATPG techniques into four main classes. These are functional (3) and structural (4) test generation algorithms, ATPG based on automatic test selection algorithms (1) and DFT-based analogue ATPG algorithms (6). It is interesting to note that two out of six different DFT strategies in Soma's classification do not have associated ATPG algorithms.

Analogue Fault diagnosis methods can be grouped into either two[31][32][33] or four[34] categories. The former grouping divides the methods into Simulation Before Test (SBT) and Simulation After Test (SAT) techniques. SBT techniques are best used to detect and locate hard faults (open and short circuit) by simulating all possible fault conditions of the CUT and storing the simulated data in a fault dictionary so that measured data from the test can be compared with the fault signatures stored in the dictionary. A fault is diagnosed when its corresponding signature in the dictionary matches the measured data to within a predefined tolerance. The name SBT is self-explanatory as all the simulations of faults are done before the test (comparison of the measured data and fault signatures) is carried out. The variations of SBT techniques[32][35] differ mostly in how the fault dictionary is established and handled. Its disadvantages are the large volume of data to be processed when forming the fault dictionary and the risk of overlooking faults which are not included in the fault dictionary. These disadvantages become more significant when SBT techniques are used to diagnose soft faults (deviation in component values) because the size of the dictionary is much larger than that in the case of hard faults, and the possibility of not including all possible fault conditions in the dictionary is also greater.

With SAT techniques, as the name implies, all circuit simulations take place after the testing process. Early publications[36][37][38] concentrated on parameter identification techniques which require one to solve the solution of systems of non-linear equations that are implied by KCL, KVL and branch voltage-current relations. Its drawback is the formidable numerical difficulty when dealing with large size circuits.

The classification of analogue fault diagnose techniques in [34] is based on how the CUT is represented . They are as follows:

1. Taxonomical method

This is synonymous with the SBT techniques.

2. Deterministic method

For a CUT with n input variables $x_0,\ x_1,\cdots\cdots,\ x_{n-1}$, the output y_j at one of the r (n>r) test points is $y_j = f_j(x_0,\ x_1,\cdots\cdots,\ x_{n-1}),\ j=0,\ 1,\cdots\cdots,\ r-1$. The deterministic method determines the deviation of the measured output responses $\Delta\mathbf{Y}$ from the expected ones which are calculated from the analytic relations $f_0,\ f_1,\cdots\cdots,\ f_{r-1}$. A criterion which can be expressed as an analytic expression involving $\Delta\mathbf{Y}$ is chosen with hypotheses on the analogue faults governing how the expected output responses are evaluated. To show how this technique diagnoses a fault, the Associated Least Squares Criterion is used as an example:

Let S_i be a factor of merit associated with each input variable $x_i,\ 0 \le i < n$,

$$S_i = \sum_{j=0}^{r-1} (g_j - y_j)^2$$

With the measured and expected outputs at the test point j denoted by g_j and y_j respectively. The hypotheses are

- single fault.
- the S_k of the input variable $x_k,\ 0 \le k < n$, which corresponds to the most likely faulty component is the minimum factor of merit.

The steps to carry out the diagnosis are as follows:

1) r measurements $g_0,\ g_1,\cdots\cdots,\ g_{r-1}$ are taken.

2) For each input variable, the value of x_i is computed which minimises S_i with the other input variables being at their nominal values.

3) The values of S_i for each x_i computed in 2) are evaluated.

4) The faulty component corresponds to the x_i in 3) which yields minimum S_i.

There are many variants in this technique depending on what hypotheses and criterion a variant adopts. Its main disadvantage is the large amount of computations to be done at the test.

3. Probabilistic method

This technique arrives at the faulty component by comparing the measured characteristics of the CUT with its nominal ones and at the same time taking into considerations the associated parameter tolerances. Its main principle is to determine the probability for each individual parameter which causes the discrepancy between the nominal and measured values of the characteristics. The component corresponding to the parameter whose probability is maximum is thus the most likely to be faulty.

Its main disadvantages are its restriction to the single fault case and linear circuits. Additionally, it can only handle soft faults.

4. Topological method

This technique is based on the knowledge of the structure of the CUT which generally comes from its netlist. As in the cases in the other three categories, there are variants in this method. The most promising variant represents the topology of the CUT as a linear graph in an exact[39] or abstract[40] manner. [39] has established a testability condition which depends on topology only. However, it is not very practical as its focus is on node-faults instead of component faults. It defines a faulty node as a node to which faulty components are connected. Diagnosis of faulty components is thus indirectly through the diagnosis of faulty nodes. [40] hierarchically decomposes the CUT into sub-circuits using the measurement nodes. Fault diagnosis is achieved by checking the consistency of KCL between decomposed sub-circuits to locate faulty sub-circuits. The decomposition carries on until faulty components are located. This approach is not practical at the IC level as it requires access to a large number of internal nodes to get a reasonable fault coverage which leads to very high overhead costs. However, it is adopted for testing faulty chips at the board level as all the chip pins are accessible. In addition to this accessability constraint, this approach is only suitable for testing bipolar IC as the current flow in CMOS circuits is generally very small, in particular the current flow into the gate inputs, which are always the most accessible nodes. The consistency of KCL is thus very difficult to check by summing very small currents to zero, particularly with small tolerance variations. Nonetheless, the topology method in general is the more efficient the larger the CUT is .

The two different classifications of fault diagnosis methods discussed so far are not exhaustive in their contents and do not overlap each other completely. The SAT techniques in the first classification do not necessarily include all methods in categories 2., 3. and 4. of the second classification. For example, the probabilistic method can be considered a SBT technique[33].

There are other techniques which were proposed after these two classifications were made. For example, the sensitivity based technique[41] which has a firm theoretical foundation and has already resulted in an automatic sensitivity analysis tool, LIMSoft[42][43], for analogue test generation. This technique is useful for detecting parameter deviation faults. Lately, techniques based on artificial neural networks has attracted increasing attention[44, 45, 46, 47, 48].

1.3. IC Development and Test Challenges of Mixed Signal ICs

Since the invention of the transistor in 1947 and the inception of IC development at the beginning of 1960, there have been a few generations of ICs in terms of integration density, and a diversity of IC applications which required different technologies. Integration density for the CMOS technology, in terms of the number of on-chip transistors, has grown from SSI with less than 100 transistors per chip to ULSI with more than 100 thousand transistors per chip. In between these extremes are MSI, LSI and VLSI. Most analogue circuits are either SSI or MSI. Mixed signal (analogue and digital) circuits are either MSI or LSI, whereas digital circuits can be from SSI (a few logic gates on a chip) to ULSI and beyond (microprocessor and memory chips) According to [49], the current 5 million transistors per chip limits is predicted to increase to 800 million by the year 2010. Most of these transistors are needed for on-chip memory and the rest are used for logic elements and interfaces so that a true system-on-a-chip can be realized. An alternative form of integration, the Multi-Chip Module (MCM), which is considered to be the fifth generation of ICs, has 10 to 20 million devices per module[50]. As for IC technologies, the first mature technology was BJT, followed by NMOS, CMOS and then BiCMOS. These technologies have their relative merits and disadvantages, and their uses depend very much on the applications[1]. Apart from these silicon based technologies, there are evolving technologies[50] which are based on SiGe (Silicon-Germanium) Heterostructures, III-V (Gallium Arsenide (GaAs) and Indium Phosphide (InP)) and II-VI compound semiconductors for specialized applications. The most widespread used compound semiconductor is GaAs, which has applications in high frequency microwave circuits and in on-chip integrated optics (integration of electronic and optical devices on the same GaAs substrate).

Digital ICs have been mainly used in the computer industry since their development. As the computer industry expanded into the personal computer (PC) market in the late seventies, and has continued its expansion into the nineties, there have been five generations of PCs which utilizes digital ICs with increasing speed, functionality and integration. This huge demand from the computer industry has fueled the research on design and test of digital ICs to the current mature state that there are now available a common design language (VHDL) and testing standard (IEEE 1149.1), which facilitate the automation of design and test for digital ICs.

As the real world is undoubtedly analogue in nature, many applications which need the use of both digital and analogue circuits had to resort to implementing the required systems on board level before process technologies were advanced enough to allow integration of mixed analogue and digital circuits. These mixed signal ICs have been made a reality by the advancement in CMOS technology and the arrival of BiCMOS technology. As a result, there has been significant growth on the demand for mixed signal ICs in recent years. This growth has been coupled with the consumerization of electronics, i.e. the convergence of multimedia, telecommunications and computing, to result in increased challenges in the design of mixed signal ICs. Compounding the design issue is the fact that commercial competitiveness calls for increased functionality on decreased chip area with better performance and faster time-to-market. Exacerbating this issue further is the trend towards system-on-a-chip and more applications which require highly complex monolithic solutions with low power and portability requirements. These design challenges are never ending and as soon as a new challenge emerges, it will soon translate itself into a test challenge as design and test are closely related issues.

1.4. *DFT: an Aid to tackle the Test Challenges*

DFT techniques for mixed signal and analogue circuits have more variety than their digital counterpart and are divided into three methodologies[51] or six different approaches[52]. Parallel to the digital scan path design (Chapter 1.2) is the so called analogue access or test point insertion based methodology[53][54] which uses an analogue scan path for controllability and observability of the internal nodes of a CUT. Specially designed analogue shift register cells are used to form the analogue scan chain for loading of test data and readout of the test results in a way similar to the digital scan path. An example of analogue shift register design utilizing a simple discrete-time current copier consisting of switches, a MOS transistor and a capacitor is described in [55]. An alternative to the analogue shift register is to use the sw-opamp (switchable opamp) concept, originally proposed by Bratt[56]. A sw-opamp is a configurable opamp with a digital control signal to switch it into either the buffer or opamp mode. A low control signal sets the sw-opamp into the opamp mode, in which it functions as a normal differential input, single-ended output opamp. When the buffer mode is selected by setting the control signal to high, the sw-opamp becomes a buffer, where its input passes directly to its output. Thus, this method is particularly suitable for circuits which can be partitioned into blocks of opamp stages. Controllability and observability of a chosen opamp block are achieved by putting the chosen block into opamp mode and all the other blocks into buffer mode. In this way, the test signal for the chosen opamp block can be injected from a primary input and the test results observed from the primary output of the CUT. Different CMOS implementation of the sw-opamp in single-ended[57] and differential-ended[58] configurations have been investigated to improve on the sw-opamp implementation proposed by Bratt[56].

Opposite to the analogue access based DFT is the reconfiguration based methodology which achieves testability improvement by reconfiguring the CUT with MOS switches. The pioneering paper in this methodology is by Soma[59]. He shows that the controllability and observability of each stage in cascaded stage active filters can be improved by systematically inserting MOS switch combinations into the CUT dynamically to increase the bandwidth of each stage in test mode. This concept is extended to switched capacitor (sc) filters[60] to take into account that sc filters already have MOS switches.

The last class of methodology is a direct parallel to the digital BIST but it is divided into functional and fault based. As their names imply, functional based BIST techniques test the functional specifications of the CUT with conventional test stimuli whereas fault based BIST methods aim at fault detection with non-conventional stimuli and signatures. Examples of functional specifications tested by BIST are signal-to-noise ratio (SNR) and frequency response. Classes of circuits employing functional based BIST are sigma-delta[61] and successive approximation[62] ADCs, and single chip speech CODEC[63]. In [63], the combined performance of the on-chip ADC and DAC are measured by a FFT analysis on the ADC output, with the ADC input derived from the output of the DAC which takes a 512-point sinusoidal input stimulus generated from the on-chip DSP cores as its input.

Ohletz[64] proposed one of the earliest fault-based BIST schemes which allows analogue test response evaluation within a digital kernel. This is done by the dual use of a special on-chip generator/analyzer, the Hybrid Test Stimulus Generator (HTSG) which provides the analogue test stimulus and evaluates the test signatures. Another fault based BIST which has both on-chip test signal generation and test response evaluation is a multifunctional test structure called ABILBO[65]. It generates test signals at different frequencies, performs signature analysis and achieves analogue scan depending on which operating mode it is in. Other fault based BIST schemes concentrate on test signature analysis to accommodate tolerances in analogue signals either by digital[66] or analogue[67] integration. Recently, a new fault based BIST scheme, the oscillation test method[68], has been proposed for opamp based circuit. It removes the need for test signal generation by turning the CUT into an oscillator. The oscillation frequency of a faulty CUT is different from that of a fault free CUT. As customary with all analogue variables, there is a tolerance band associated with the fault free oscillation frequency. This tolerance band is determined by Monte Carlo analysis taking into consideration the nominal tolerance of all important technologies and design parameters. A fault will be detected if it shifts the oscillation frequency out of the tolerance band. Another class of circuits which are ideal candidates for the oscillation test method is the biquadratic filter[69] because of its inherently oscillatory nature. To make the biquad oscillate, its damping loops are cut and filter coefficients programmed digitally so that the poles of the biquad are relocated onto the imaginary axis of the s-plane.

The reconfiguration based and analogue access methodologies correspond directly to the "set-up and execution of module test (BIST)" and "access to embedded modules/nodes" DFT approaches in [52] respectively. While the BIST based methodology maps to the "on-chip generation/evaluation of test stimuli" and "on-chip multi-module/system test" DFT approaches in [52]. The last two DFT approaches in [52] do not relate to any of the methodologies in [51]. One of the last two DFT approaches "support for external test and evaluation" refers to transient response technique[23] and Iddq testing[25]. The remaining DFT approach on "general DFT rules and guidelines" includes ad hoc DFT.

1.5. Industrial Impetus for Research on Fault Diagnosis Procedure

The research work undertaken by the author has its industrial impetus from Robert Bosch GmbH, the recognised world wide leader in the field of automotive electronics[70]. Bosch has a need to investigate the testing of mixed signal ICs, such as sigma delta ADC, with an ultimate aim to optimise testing at an early stage of the design

(DFT) in order to maximise the fault coverage for production test, thus saving expensive re-design. This investigation has been split into test preparation and generation. Activities to improve test preparation at Bosch have been ongoing since 1995: the AMITY[71] project will result in a virtual design test bench to integrate the design and test generation stage together. Linking to AMITY is the VIRTUS[72] project, its aim is to incorporate tester models into the AMITY environment in order to evaluate the interaction of the ICs and test steps before the first silicon is fabricated. Thus, effective use of the tester time is ensured. Test generation support with an interface to AMITY and VIRTUS has been dealt with by the collaboration between Bosch and the University of Bath. The aim of this collaboration is to develop fault diagnosis/test generation procedures which clarify the reachable diagnosis levels, and provide a set of test signals optimized for the activation of sensitive circuit parameters. The author's work is the first part of this collaboration to develop fault diagnosis procedure for analogue circuits. To apply the diagnosis procedure to practical circuits, realistic fault models which describe the tolerance behaviour of typical analogue building blocks, their sensitivity to critical process variations and environmental parameter influence are necessary. Analogue fault modelling[90] is the second part of this collaboration.

1.6. Fault Diagnosis Approach and Thesis Layout

Despite two decades of development, there is still a lack of software tools in analogue fault diagnosis and the issue of hierarchical testing for analogue circuits is still not tackled successfully. The author's work builds on one of the fault diagnosis methods after reviewing the others and has already laid a foundation stone towards the goal of hierarchical testing[73] which is further discussed in [74]with example test circuits.

Having reviewed a number of fault diagnosis techniques, a decision has been taken to use the approach of Wey[75] and Wu[76] as the basis of the author's work. This approach can be classified as the topological method or the SAT technique with fault verification. As opposed to the parameter identification SAT technique, this approach does not solve for parameter values, but attempts to carry out computations (instead of simulations) under different circuit component partitions and then some decision algorithms[76] are used to locate the faulty components.

Wey[75] adopts the Pseudo Circuit Approach which is based on a Self Test (ST) Algorithm[76] formulated on the Component Connection Models (CCM)[77]. The merit of Wey's approach is structured DFT, which integrates circuit diagnosability in the design stage instead of having the test problem considered after the chip has been laid out. In addition, its basis, the combined formulation of the CCM and ST Algorithm, is also the formulation of three different pieces of works [26], [27], [28] (Chapter 1.2) with different emphasis in their development of this basic formulation. Chronologically speaking, [26] was published before the author embarked on this research work in late 1994, whereas [27] and [28] were not known to the author when he had to make a decision on which diagnosis approach to develop, although the Wey's approach was recommended by Bosch at the outset of this research. With hindsight, the decision is correct as three other researchers were also inspired by Wey's work and coincidentally chose his approach for development, perhaps because of its flexibility, power and potential for low test cost. As the Wey's approach is algorithmic based, it is inherently programmable and its successful development into a practical fault diagnosis procedure will no doubt make it a complementary test tool to all the other hardware-based DFT techniques as it does not need test circuitry implemented on expensive chip area.

Wey's contribution is the proposal of a testability condition for analogue circuits once the test points and tester/testee partition of the CUT are known. This testability condition depends only on the circuit topology and is not affected by component values. He also proposed two essential test point selection rules and test point compaction. These rules select elements of the system output vector **b** as test points and are based on topology and diagnosability considerations, which reflect in the column entries of connection matrix L_{11}[75]. In the case of the number of test points selected being more than the allowable limit set by diagnosability, test point compaction is used to reduce the test point number down to the diagnosability limit by making use of an element of the system input vector **a**, which can be expressed in terms of these **b** vector elements it replaced.

The work done to date in this project has improved on Wey's work by supplementing how to generate the circuit connection equation with an Optimum Tree Generation Procedure (OTGP)[78], narrowing down the type of essential test points on which compaction is applied, clarifying how to carry out compaction iteratively and adding the minimum test node mapping scheme to address the physical measurement issue and to cover the case when the number of essential test points selected is less than the diagnosability limit. With all these improvements, a coherent Fault Diagnosis Algorithm (FDA) has been developed based on the ST Algorithm and a novel Test Point Selection (TPS)[78] Scheme to give maximum fault coverage with a minimum number of test nodes. The FDA achieves maximum fault coverage by requiring a user to input a Diagnosis Depth, which is the expected number of maximum simultaneous faults in the CUT. With the FDA as the foundation, the hierarchical approach is integrated with the FDA to realize a Hierarchical FDA with its key advantages being the efficient use of computing resources and the diagnosis of hard faults entirely embedded in hierarchical components, as well as soft faults. The former advantage is absolutely necessary for any fault diagnosis procedure to be of practical use in the testing of large scale analogue circuits.

The novelty of the author's work is the development of a hierarchical fault diagnosis procedure which does not require any expensive on-chip test circuitry. This procedure begins with the translation of the CUT netllist into a directed linear graph which has many possible mathematical representations in the form of matrix connection equations

(Chapter 2). The next step is to select a connection equation optimised for computation with the OTGP (Chapter 3) with hierarchical modification (Chapter 6.3). Central to the OTGP is the construction of the CUT node incidence matrix which is equivalent to applying the KCL to all the nodes of the CUT except the reference node. During this construction process, high sparsity matrices ($\mathbf{L}_{11}$ and $\mathbf{L}_{12}$) in the connection equation is ensured by using rules to allocate the column indices of the incidence matrix to graph edges corresponding to the CUT components. Manipulations of this matrix lead eventually to the derivation of the optimised connection equation. Once the connection equation is derived, a set of test points, which can diagnose faults up to the user specified test requirement (diagnosis depth), is obtained by the test point selection procedure (Chapters 4 and 6.4). This selection procedure applies the topology and diagnosability rules to the CUT by examining the column entries of the $\mathbf{L}_{11}$ connection matrix to select essential test points, which are either compacted or supplemented depending on whether the number of essential test points are more than or less than the diagnosability limit.

Chapter 5 explains the basic fault diagnosis algorithm which encompassed issues such as the ST Algorithm and CCM formulation (how faults are diagnosed with iterative test cycles), the mathematics behind the matrix computations and operations done in a test cycle and the required decision algorithms. Chapter 6 introduces the hierarchical approach and the required changes to the basic FDA before describing the hierarchical FDA. Chapter 7 investigates the effectiveness of the Hierarchical FDA by applying its C-language implementation, the diagnosis program, to several test circuits under different fault conditions, and evaluating the consequent diagnosis results to make recommendations on further improvements of the algorithm and identify future research areas to turn the Hierarchical FDA into a practical test tool for large scale analogue circuits.

Chapter 8 concludes on the achievements of this research work against its initial aim and recommends future areas for further work to enable the Hierarchical FDA to become a practical test tool.

2. Mathematical Description of Circuit Topology

This section describes some of the terms used in graph theory[79] and the conventions used in drawing a graph for the CUT. It also demonstrates the procedures used to derive a connection equation for the CUT by means of Example 1 in Figure 2-1.

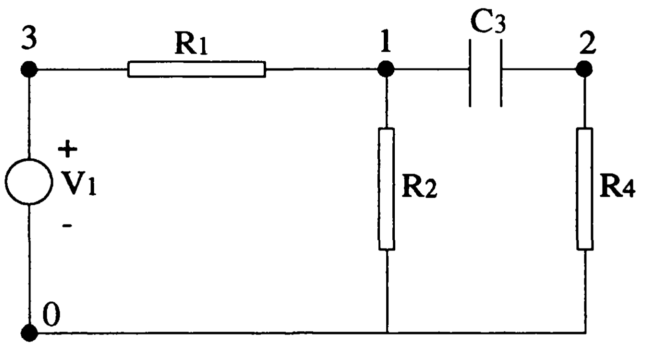


Figure 2-1 Example1

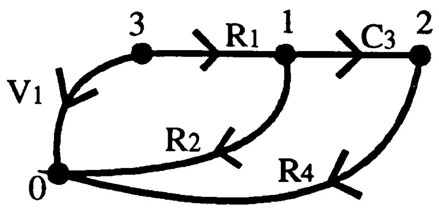


Figure 2-2 Graph of example 1

2.1. Terms and Conventions

Circuit topology is conveniently expressed as a directed linear graph[79] by representing two terminal circuit components as edges and their connecting points as nodes (see Chapter 6.1 for the graph representation of circuit components with multiple terminals). The direction of current flow or voltage drop across an edge component is denoted by an arrow on the edge. In order to have consistent current and voltage graphs, the current of the voltage or current sources is defined to flow out from its negative node instead of its positive node.

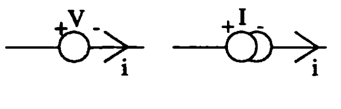


Figure 2.1-1 Conventions on voltage polarity and direction of current flow in independent sources

A node is defined as an in-node of an edge if the current flows into the edge component from that node. Otherwise, it is an out-node of the edge. For example, node 3 is the in-node and node 1 is the out-node of edge R_1 in Figure 2-2.

A tree of a graph consists of the edges which connect all the nodes without completing any closed loop. The cotree is then defined as the complement of the tree. The total number of tree/cotree combinations in a graph is equal to the value of the determinant $|AA^T|$[80], where A is the node incidence matrix of the graph. This is proved on page 260 to 269 in [79]. Listed in Figure 2.1-2 are all the possible trees of the example circuit in Figure 2-1. They can be obtained with the spanning tree algorithm in Grimbleby[80] as shown in Figure 2.1-3 or by intuition.

The degree[79] of a node is defined to be the number of edges connected to it. For practical circuits, there will not be any self looping edges, which means that the in-node and out-node of an edge are distinct.

The fundamental cut set[79] of a tree edge, S(tree edge) is defined to be the set of cotree edges of which an element edge will reconnect the tree nodes if the tree edge is removed. For example, in Figure 2.1-2d, $S(v_1)=\{R_1\}$, $S(R_2)=\{R_1,C_3\}$, $S(R_4)=\{C_3\}$.

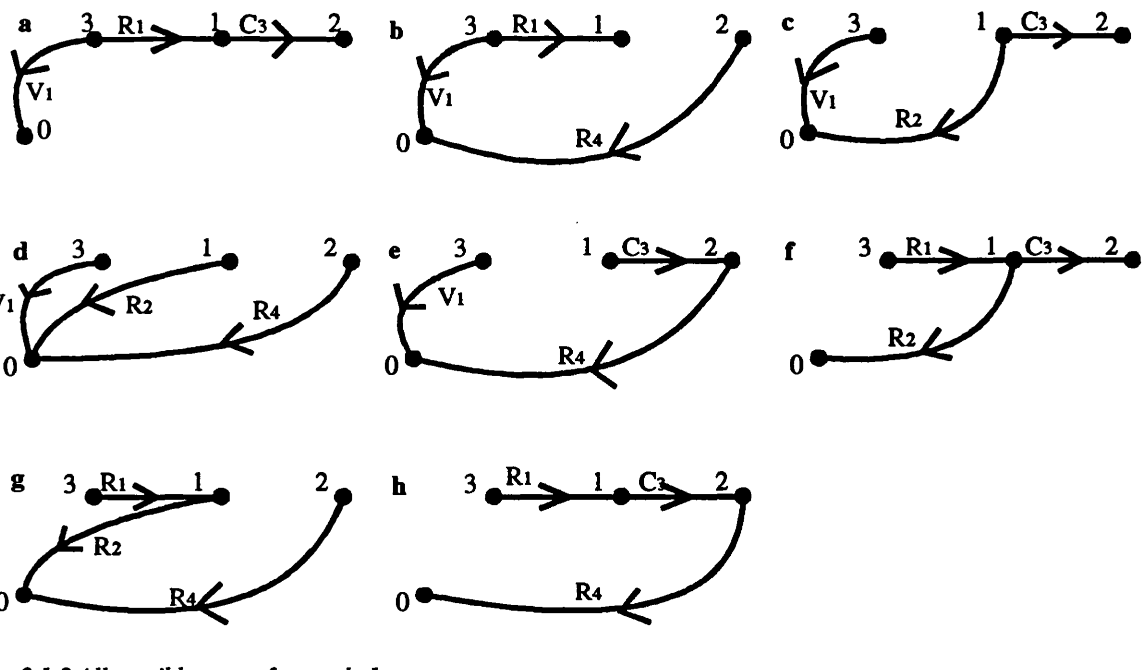
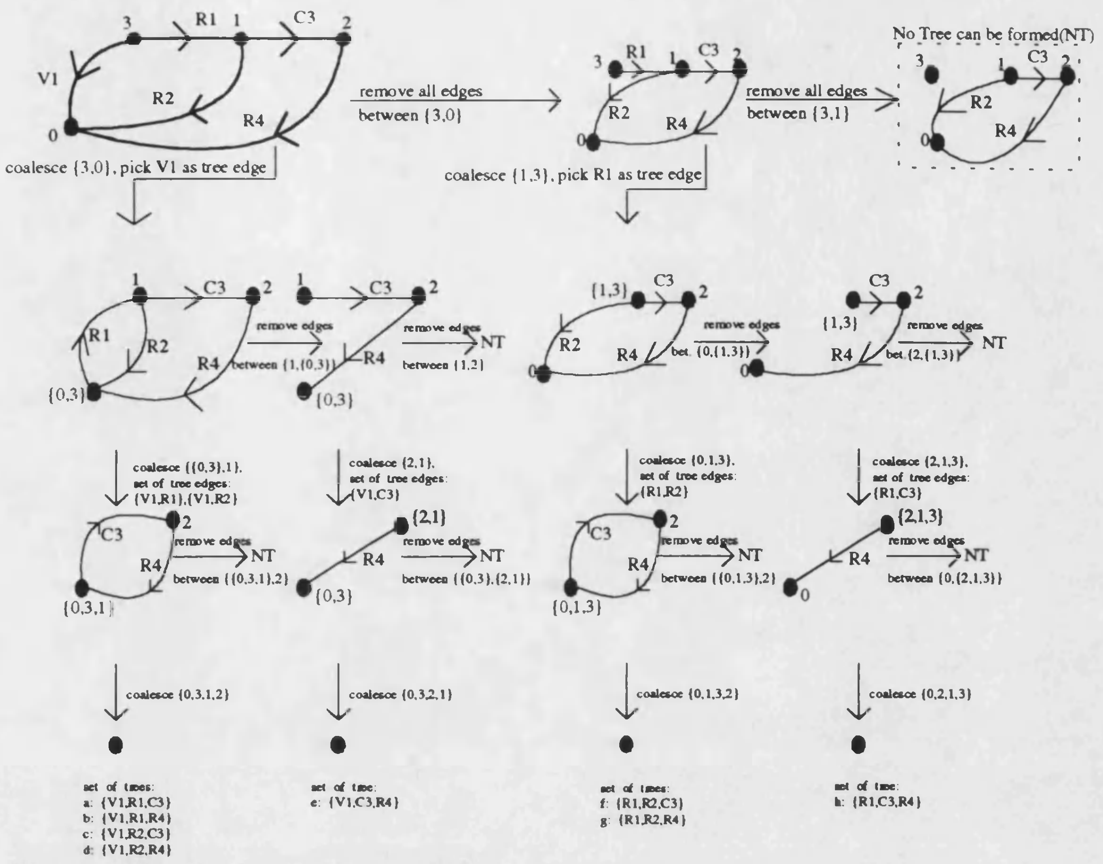


Figure 2.1-2 All possible trees of example 1



remove all edges between {3,0}
coalesce {3,0}, pick V1 as tree edge
remove all edges between {3,1}
No Tree can be formed(NT)
coalesce {1,3}, pick R1 as tree edge
remove edges between {1,{0,3}}
remove edges between {1,2}
remove edges bet. {0,{1,3}}
remove edges bet. {2,{1,3}}
coalesce {{0,3},1}, set of tree edges: {V1,R1},{V1,R2}
coalesce {2,1}, set of tree edges: {V1,C3}
coalesce {0,1,3}, set of tree edges: {R1,R2}
coalesce {2,1,3}, set of tree edges: {R1,C3}
remove edges between {{0,3,1},2}
remove edges between {{0,3},{2,1}}
remove edges between {{0,1,3},2}
remove edges between {0,{2,1,3}}
coalesce {0,3,1,2}
coalesce {0,3,2,1}
coalesce {0,1,3,2}
coalesce {0,2,1,3}
set of trees: a: {V1,R1,C3} b: {V1,R1,R4} c: {V1,R2,C3} d: {V1,R2,R4}
set of tree: e: {V1,C3,R4}
set of trees: f: {R1,R2,C3} g: {R1,R2,R4}
set of tree: h: {R1,C3,R4}

Figure 2.1-3 Spanning tree route for example 1

2.2. *Connection Equation*

Once a tree of the CUT is specified, its connection matrices $\mathbf{L}_{11}$ and $\mathbf{L}_{12}$ in

$$\mathbf{a} = \mathbf{L}_{11}\mathbf{b} + \mathbf{L}_{12}\mathbf{u} \tag{2.2-1}$$

- a: system input vector
- b: system output vector
- u: system stimulus vector
- $\mathbf{L}_{11}$, $\mathbf{L}_{12}$: sparse connection matrix

can be derived in one of two ways. They differ in the way the so-called fundamental matrix, **D**, is obtained.

The equivalence of (2.2-1) in DeCarlo[77] is

$$\begin{bmatrix} \mathbf{i}_t \\ \mathbf{v}_{ct} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & -\mathbf{D}' \\ \mathbf{D}'^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}_t \\ \mathbf{i}_{ct} \end{bmatrix} - \begin{bmatrix} \mathbf{E} & \mathbf{0} \\ \mathbf{0} & \mathbf{F} \end{bmatrix} \begin{bmatrix} \mathbf{J}_t \\ \mathbf{E}_{ct} \end{bmatrix} \qquad (2.2\text{-}2)$$

with

- $\mathbf{a} = \begin{bmatrix} \mathbf{i}_t \\ \mathbf{v}_{ct} \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} \mathbf{v}_t \\ \mathbf{i}_{ct} \end{bmatrix}$, $\mathbf{u} = \begin{bmatrix} \mathbf{J}_t \\ \mathbf{E}_{ct} \end{bmatrix}$, E, F: matrices of appropriate sizes.
- $\mathbf{J}_t$: column vector of currents of independent current sources in parallel with tree edges
- $\mathbf{E}_{ct}$: column vector of voltages of independent voltage sources in series with cotree edges.

The deficiencies in DeCarlo's approach are that it does not deal with controlled sources and independent voltage and current sources have to be in series with cotree edges and in parallel with tree edges respectively. These pose severe constraints on circuit types and topologies that approach can handle.

These deficiencies are due to the fact that all sources are excluded from the graph representation of the CUT. Improvements are made by including all sources, whether independent or controlled, as graph edges and imposing some rules based on [81] to select tree edges. These tree edge selection rules are summarised in Table 2.2-1.

Tree edges	Voltage sources	Capacitors	Resistors
Cotree edges	Current sources	Inductors	

Table 2.2-1 Tree edge selection rules

If these rules are obeyed, the fundamental matrix can always be derived analytically.

We have

$$\begin{bmatrix} \mathbf{i}_T \\ \mathbf{v}_{CT} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & -\mathbf{D} \\ \mathbf{D}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}_T \\ \mathbf{i}_{CT} \end{bmatrix} \tag{2.2-3}$$

with $\mathbf{i}_T = \begin{bmatrix} \mathbf{i}_{vs} \\ \mathbf{i}_t \end{bmatrix}$, $\mathbf{v}_{CT} = \begin{bmatrix} \mathbf{v}_{ct} \\ \mathbf{v}_{is} \end{bmatrix}$, $\mathbf{v}_T = \begin{bmatrix} \mathbf{v}_{vs} \\ \mathbf{v}_t \end{bmatrix}$, $\mathbf{i}_{CT} = \begin{bmatrix} \mathbf{i}_{ct} \\ \mathbf{i}_{is} \end{bmatrix}$

Subscript:

- is: Independent current source.
- vs: Independent voltage source.
- t: Tree edges excluding vs.
- ct: Cotree edges excluding is.

With the partition,

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_{vc} & \mathbf{D}_{vi} \\ \mathbf{D}_{tc} & \mathbf{D}_{ti} \end{bmatrix}, \quad \mathbf{D}^T = \begin{bmatrix} \mathbf{D}_{vc}^T & \mathbf{D}_{tc}^T \\ \mathbf{D}_{vi}^T & \mathbf{D}_{ti}^T \end{bmatrix}$$

An equivalence of (2.2-2) can be extracted from (2.2-3),

$$\begin{bmatrix} \mathbf{i}_t \\ \mathbf{v}_{ct} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & -\mathbf{D}_{tc} \\ \mathbf{D}_{tc}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}_t \\ \mathbf{i}_{ct} \end{bmatrix} + \begin{bmatrix} -\mathbf{D}_{ti} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{vc}^T \end{bmatrix} \begin{bmatrix} \mathbf{i}_{is} \\ \mathbf{v}_{vs} \end{bmatrix} \tag{2.2-4}$$

with $\mathbf{u} = \begin{bmatrix} \mathbf{i}_{is} \\ \mathbf{v}_{vs} \end{bmatrix}$

To derive **D** analytically, the circuit and graph in Figures 2.2-1 and 2.2-2 are used as an example.

We have,

$$\mathbf{v}_{vs}=\begin{bmatrix}\mathbf{v}_{V_3}\\ \mathbf{v}_{V_2}\\ \mathbf{v}_{V_1}\end{bmatrix},\quad \mathbf{i}_{is}=\begin{bmatrix}\mathbf{i}_{I_1}\\ \mathbf{i}_{I_2}\end{bmatrix},\quad \mathbf{i}_{t}=\begin{bmatrix}\mathbf{i}_{R_7}\\ \mathbf{i}_{R_3}\\ \mathbf{i}_{R_5}\\ \mathbf{i}_{R_2}\end{bmatrix},\quad \mathbf{i}_{ct}=\begin{bmatrix}\mathbf{i}_{R_8}\\ \mathbf{i}_{R_1}\\ \mathbf{i}_{R_4}\\ \mathbf{i}_{R_6}\end{bmatrix}$$

$$\mathbf{i}_{vs}=\begin{bmatrix}\mathbf{i}_{V_3}\\ \mathbf{i}_{V_2}\\ \mathbf{i}_{V_1}\end{bmatrix},\quad \mathbf{v}_{is}=\begin{bmatrix}\mathbf{v}_{I_1}\\ \mathbf{v}_{I_2}\end{bmatrix},\quad \mathbf{v}_{t}=\begin{bmatrix}\mathbf{v}_{R_7}\\ \mathbf{v}_{R_3}\\ \mathbf{v}_{R_5}\\ \mathbf{v}_{R_2}\end{bmatrix},\quad \mathbf{v}_{ct}=\begin{bmatrix}\mathbf{v}_{R_8}\\ \mathbf{v}_{R_1}\\ \mathbf{v}_{R_4}\\ \mathbf{v}_{R_6}\end{bmatrix}$$

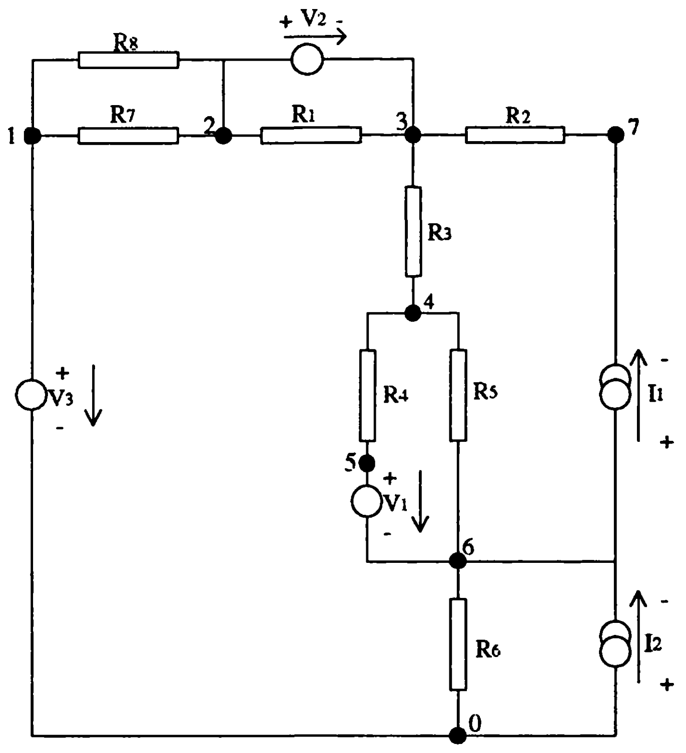


Figure 2.2-1 Example 2

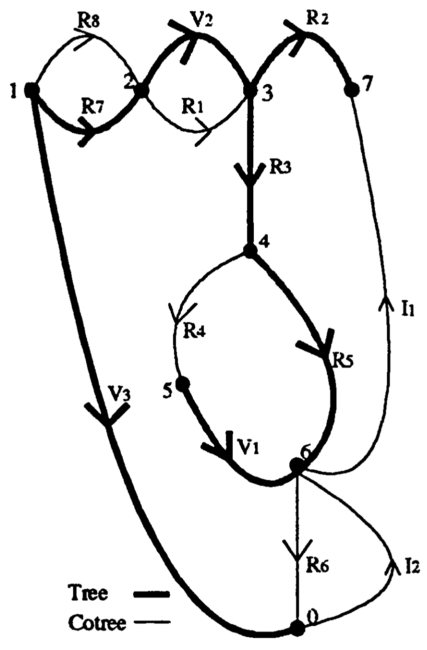


Figure 2.2-2 Graph of example 2

For Example 2, its node incidence matrix A is obtained by applying KCL for node 1 to 7 (Node 0 is taken as the reference node. A n-node circuit is completely described by n-1 nodal equations.).

	Tree edge matrix A_T							Codtree edge matrix A_{CT}					
Nodes\Edges	V3	V2	V1	R7	R3	R5	R2	R8	R1	R4	R6	I1	I2
1	1	0	0	1	0	0	0	1	0	0	0	0	0
2	0	1	0	-1	0	0	0	-1	1	0	0	0	0
3	0	-1	0	0	1	0	1	0	-1	0	0	0	0
4	0	0	0	0	-1	1	0	0	0	1	0	0	0
5	0	0	1	0	0	0	0	0	0	-1	0	0	0
6	0	0	-1	0	0	-1	0	0	0	0	1	1	-1
7	0	0	0	0	0	0	-1	0	0	0	0	-1	0

*Table 2.2-2 Node incidence matrix **A** of example 2*

Analytically, KCL states that

$$[\mathbf{A}_T \quad \mathbf{A}_{CT}]\begin{bmatrix}\mathbf{i}_T \\ \mathbf{i}_{CT}\end{bmatrix} = \mathbf{0}$$

$$\Rightarrow \quad \mathbf{A}_T\mathbf{i}_T + \mathbf{A}_{CT}\mathbf{i}_{CT} = \mathbf{0} \quad \Rightarrow \quad \mathbf{i}_T + \mathbf{A}_T^{-1}\mathbf{A}_{CT}\mathbf{i}_{CT} = \mathbf{0} \quad \Rightarrow \quad \mathbf{i}_T = -\mathbf{D}\mathbf{i}_{CT}$$

$$\mathbf{D} = \mathbf{A}_T^{-1}\mathbf{A}_{CT} \tag{2.2-5}$$

For the graph of Figure 2.2-2,

$$\mathbf{D} = \begin{bmatrix} 0 & 0 & 0 & 1 & \vdots & 0 & -1 \\ 0 & 1 & 0 & -1 & \vdots & 0 & 1 \\ 0 & 0 & -1 & 0 & \vdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & 0 & 0 & -1 & \vdots & 0 & 1 \\ 0 & 0 & 0 & -1 & \vdots & -1 & 1 \\ 0 & 0 & 1 & -1 & \vdots & -1 & 1 \\ 0 & 0 & 0 & 0 & \vdots & 1 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{D}_{vc} & \vdots & \mathbf{D}_{vi} \\ \cdots & \cdots & \cdots \\ \mathbf{D}_{tc} & \vdots & \mathbf{D}_{ti} \end{bmatrix}$$

Alternatively, DeCarlo's method of using the fundamental cut sets can be used to derive the **D** matrix as follows:

Let D_{yx} be the **D** matrix entry indexed by the xth cotree element in $\mathbf{i}_{CT}$ and yth tree element in $\mathbf{i}_T$. This is the yth row and xth column of **D**.

if $x \notin S(y)$

$$D_{yx} = 0$$

else

if cotree edge x reconnects the nodes to form a tree in the way the cut tree edge y does

$$D_{yx} = 1$$

else

$$D_{yx} = -1$$

For Figure 2.2-2,

- $S(V_3)=\{R_6, I_2\}$
- $S(R_7)=\{R_8, R_6, I_2\}$
- $S(V_2)=\{R_1, R_6, I_2\}$
- $S(R_2)=\{I_1\}$
- $S(R_3)=\{I_1, I_2, R_6\}$
- $S(R_5)=\{R_4, R_6, I_1, I_2\}$
- $S(V_1)=\{R_4\}$

Tree(y)\Cotree(x)	R8	R1	R4	R6	I1	I2
V3	0	0	0	1	0	-1
V2	0	1	0	-1	0	1
V1	0	0	-1	0	0	0
R7	1	0	0	-1	0	1
R3	0	0	0	-1	-1	1
R5	0	0	1	-1	-1	1
R2	0	0	0	0	1	0

*Table 2.2-3 **D** matrix formation with fundamental cut sets*

2.3. Conclusions

This section has explained the terms and conventions needed to understand the conventional description of circuit topology as a directed linear graph[79] which consists of tree and cotree edges. There are many possible combinations of tree and cotree edges for a given circuit topology and each combination corresponds to a mathematical description of the circuit graph, and hence the circuit topology, in the form of a matrix connection equation (2.2-1). A spanning tree algorithm[80] has been illustrated with the circuit in Example 1 to demonstrate how all the tree/cotree combinations are generated.

A set of tree edge selection rules which are based on [81] and include all sources, whether independent or controlled, has been proposed as a guideline in determining which components should be on the tree. Once a tree of the CUT is chosen, the input and output vectors of the corresponding connection equation will be defined and the respective connection matrices can be obtained by partitioning the fundamental matrix, **D**, according to the number of voltage sources on the tree and the number of current sources on the cotree. The fundamental matrix of the chosen tree is derived with one of two methods which either make use of the fundamental cut sets of all the tree edges[77] or manipulate the tree/cotree partitions of the node incidence matrix of the CUT analytically. These methods have been demonstrated with the circuit in Example 2.

3. Implementation Issues

Both methods for deriving the **D** matrix in the previous chapter require a tree of the CUT be specified. Given the large number of possible trees even for a small circuit (Figure 2-1 has 8 possible trees, Figure 2.2-1 has 300), the resource implications on computing time and computer memory is enormous. To avoid this problem, an Optimal Tree Generation (OTG) Procedure[78] has been developed. This is a pre-selection strategy in which modified rules from those in Table 2.2-1 and additional rules are used to select a tree that will result in computationally efficient, high sparsity connection matrices. With foresight, the drawback of this procedure may be that it is difficult to provide sufficient alternative trees to re-run the FDA should the test using the optimal tree fail. Its merit is its huge saving in computing resources. However, to appreciate the limit on circuit size the problem imposes, code has been written to follow a post-selection strategy in which all the possible trees of the CUT are generated before the rules in Table 2.2-1 are used to select a subset of all these trees. This subset is then used to provide the alternative trees needed to re-run the FDA should the test using the first tree from the subset fail.

The aforementioned drawback in the OTG Procedure has been overcome coincidentally by the adoption of a hierarchical approach which leads to a hierarchical OTG Procedure[73] providing more than one optimal tree. This is discussed in Chapter 6.

3.1. Post-Selection Strategy

A n-stage RLC-ladder network in Figure 3.1-1 was used to investigate the constraint of the post-selection policy on circuit size.

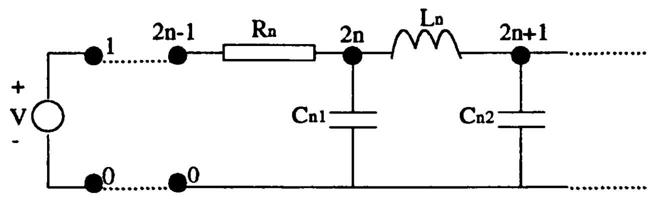


Figure 3.1-1 A n-stage RLC-ladder

The test program generates all trees of the n-stage ladder. Two subsets of trees are selected according to whether a tree satisfies all rules or only the source rule in Table 2.2-1. The results are summarised in Table 3.1-1.

Node Count	Component Count	No. of Stages	Time Taken/sec.	No.of total trees	No. of trees in the subset that satisfies	
					all rules	source rule
4	5	1	1	8	1	4
6	9	2	1	55	1	33
8	13	3	1	377	1	232
10	17	4	3	2584	1	1596
12	21	5	21	17711	1	10945
14	25	6	232	121393	1	75024
16	29	7	out of memory(amount of RAM in the SUN SPARC5 running the test program is 32 Mbytes)			

Table 3.1-1 Effect of circuit size on performance of the test program with post-selection strategy

It can be observed from the results that although the drain on computing time for the post-selection strategy is affordable, the demand on computer memory to generate all possible trees of the CUT before deriving the connection equation is unreasonable for a simple 16-node and 29-component circuit. It is thus mandatory to go for the pre-selection strategy in order for the fault diagnostic program to be of any practical use (The values recorded in the time column of the above table include the time to print out the results. They are not CPU times).

3.2. Pre-Selection Strategy

This section details the Optimal Tree Generation Procedure with reference to the circuit in Figure 3.2-2 and its graph in Figure 3.2-1.

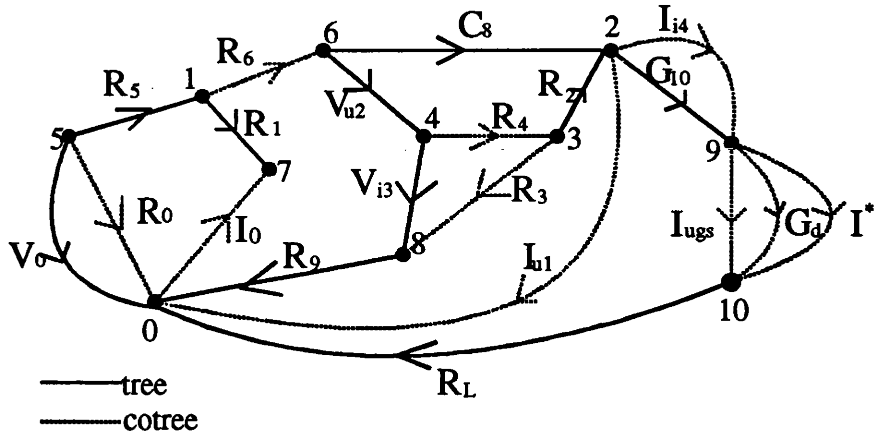


Figure 3.2-1 Graph of example 3

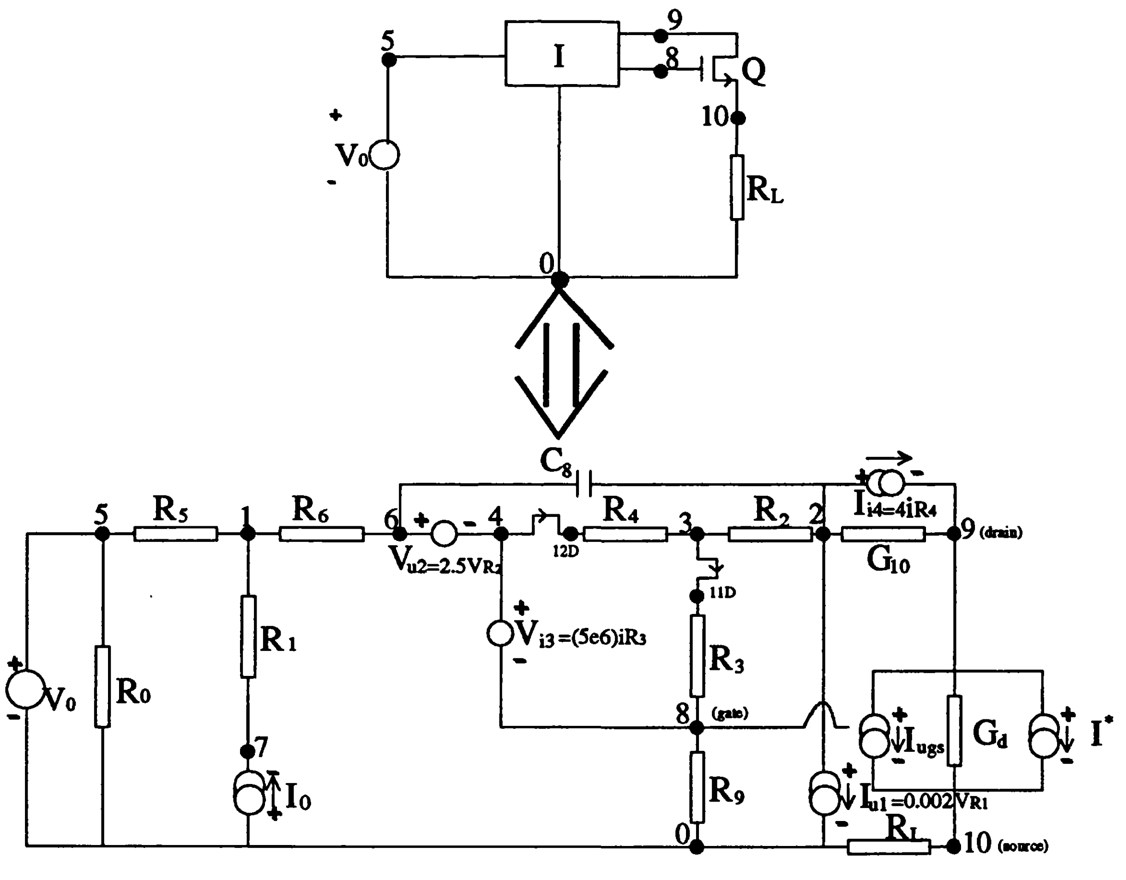


Figure 3.2-2 Example 3

Figure 3.2-2 consists of a driving source V_0, a biasing block I, a N-channel MOST Q and a load resister R_L. This is purely a fictitious circuit to include most variations of elements and their connections. Within the biasing block I, a zero resistance dummy element is put in series with the controlling element (R_3 and R_4) of every current controlled source (V_{i3} and I_{i4}) for a simulator program to fulfill KCL. The simplified model equation for Q for $v_{ds} > (v_{gs} - v_{threshold}) > 0$ is

$$I_{drain} = K(v_{gs} - v_{threshold})^2(1 + v_{ds}\lambda) \qquad (3.2\text{-}1)$$

where K is the process parameter incorporating electron mobility, gate oxide thickness and design parameters of Q. λ is a measure of the slope in the saturation region of Q.

From the iterative equation,

$$\begin{aligned} I_{drain}^{r+1} &= I_{drain}^{r} + \left(\frac{\partial I_{drain}}{\partial v_{gs}}\right)^r (v_{gs}^{r+1} - v_{gs}^{r}) + \left(\frac{\partial I_{drain}}{\partial v_{ds}}\right)^r (v_{ds}^{r+1} - v_{ds}^{r}) \\ &= G_t^r v_{gs}^{r+1} + G_d^r v_{ds}^{r+1} + I^{*r} \end{aligned} \qquad (3.2\text{-}2)$$

Q can be modelled by two controlled current sources, $I_{ugs}^r = G_t^r v_{gs}^{r+1}$, $I^{*r} = I_{drain}^r - G_t^r v_{gs}^r - G_d^r v_{ds}^r$, and a drain conductance G_d^r in each step r of the iteration.

With

$$G_t^r = 2\sqrt{I_{drain}^r K(1 + v_{ds}^r \lambda)} \qquad (3.2\text{-}3)$$

$$G_d^r = \frac{\lambda I_{drain}^r}{1 + \lambda v_{ds}^r} \qquad (3.2\text{-}4)$$

The steps of the OTG Procedure are as follows:

1. Edge Weight Allocation and Grouping of Edge Elements

The weight of an edge is defined as the sum of the degrees[79] (Chapter 2.1) of the in-node and out-node less two. If either the in-node or out-node is a dummy node, the degree of its corresponding node is used instead in edge weight calculation. In Figure 3.2-2, dummy nodes 12D and 11D correspond to nodes 4 and 3 respectively. From Figure 3.2-1, we have the degree of each node in Table 3.2-1,

Node	0	1	2	3	4	5	6	7	8	9	10
Degree	6	3	5	3	3	3	3	2	3	5	4

Table 3.2-1 Degree of nodes in example 3

Dividing the circuit elements into groups in Table 3.2-2 according to the priority 0>1>......>7 in Table 3.2-3,

Group Number	Type
0	Independent Voltage Sources connected to the reference node
1	Independent Voltage Sources not connected to the reference node
2	Controlled Voltage Sources
3	Capacitors
4	Resistors and Conductors
5	Inductors
6	Controlled Current Sources
7	Independent Current Sources

Table 3.2-2 Legend for element grouping

Group	0	1	2		3	5	6				7
Name	V_0		V_{u2}	V_{i3}	C_8		I_{i4}	I_{u1}	I_{ugs}	I*	I_0
Weight	7		4	4	6		8	9	7	7	6
Group	4										
Name	R_0	R_1	R_2	R_3	R_4	R_5	R_6	R_9	G_{10}	R_L	G_d
Weight	7	3	6	4	4	4	4	7	8	8	7

Table 3.2-3 Edge weight allocation of components in example 3

2. Construction of incidence matrix A_0

This is equivalent to applying KCL on all the nodes except the reference node 0.

- Each element is allocated a column index j according to their group priority and descending edge weight within each group in Table 3.2-4. Arranging the elements within each group in descending edge weight will minimise the number of non-zero entries in the fundamental matrix[82] , D, resulting from the OTG Procedure.

Name	V_0	V_{u2}	V_{i3}	C_8	G_{10}	R_L	R_0	R_9	G_d	R_2	R_3
j	0	1	2	3	4	5	6	7	8	9	10
Name	R_4	R_5	R_6	R_1	I_{u1}	I_{i4}	I_{ugs}	I^*	I_0		
j	11	12	13	14	15	16	17	18	19		

Table 3.2-4 Column index of node incidence matrix A_0 of example 3

- A row index i in ascending numerical order is assigned to each node except the reference node. The node is picked with the following procedures with reference to Table 3.2-4, Figures 3.2-1 and 3.2-2.

1. As the connecting nodes of group 0 edges are distinct except for the reference node, the j values of group 0 edges are allocated to the distinct nodes of corresponding group 0 edges. For example, node 5 of V_0 is allocated a row index 0. If a node is a dummy node, its corresponding node is assigned the row index instead of the dummy node.

2. Each edge in Table 3.2-4 except the group 0 edges is taken in turn for the allocation of row indices to nodes until all nodes except the reference node are assigned a row index. For each edge its j value is taken as the row index value i of its in-node subjected to three tests in the following order:

The node in test

a) is not the reference node.

b) is not a dummy node.

c) has not been already allocated.

If the in-node fails any test, the j value of the edge is assigned to its out-node provided the out-node passes tests b and c if in-node fails test a, or all tests if the in-node fails test b and c.

3. A simpler alternative to step 2 is to assign the row indices to nodes in ascending numerical order (0,1,2,...,10) using the tests in step 2. However, the resulting node incidence matrix A_0 requires more row swaps than that from step 2 to be changed to an upper triangular step matrix.

The resulting allocation of row indices to nodes using procedures {1,2} and {1,3} are shown in Tables 3.2-5 and 3.2-6 respectively.

Node	5	6	4	2	9	10	8	3	1	7
i	0	1	2	3	4	5	6	7	8	9

Table 3.2-5 Allocation of row index i with procedure {1,2}

Node	5	1	2	3	4	6	7	8	9	10
i	0	1	2	3	4	5	6	7	8	9

Table 3.2-6 Allocation of row index i with procedure {1,3}

- Fill in entries for A_0

```
for j=0 to 19
    for i=0 to 9
            do{if(in-node) entry=1; else if(out-node) entry= -1; else entry=0;}
```

The node incidence matrix using column index in Table 3.2-4 and row index in Table 3.2-5 is in Table 3.2-7.

V_0	V_{u2}	V_{i3}	C_8	G_{10}	R_L	R_0	R_9	G_d	R_2	R_3	R_4	R_5	R_6	R_1	I_{u1}	I_{i4}	I_{ugs}	I^*	I_0
1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0
0	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	-1	1	0	0	0	0	-1	0	0	0	0	0	1	1	0	0	0
0	0	0	0	-1	0	0	0	1	0	0	0	0	0	0	0	-1	1	1	0
0	0	0	0	0	1	0	0	-1	0	0	0	0	0	0	0	0	-1	-1	0
0	0	-1	0	0	0	0	1	0	0	-1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	-1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	-1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	-1

Table 3.2-7 Node incidence matrix $\boldsymbol{A_0}$ *of example 3*

3. $\mathbf{Q_0}=\mathbf{A_0}$

4. Formation of an upper triangular step matrix $\mathbf{Q_1}$

Row operations and swaps are carried out on $\mathbf{Q_0}$ so that an upper triangular step matrix $\mathbf{Q_1}$ as shown in Table 3.2-8 is formed. The element corresponding to the first matrix entry in each step (with a bold font and circled) constitutes tree edges whereas the rest are cotree edges (Page 213 and 301 in [79]).

$\mathbf{V_0}$	$\mathbf{V_{u2}}$	$\mathbf{V_{i3}}$	$\mathbf{C_8}$	$\mathbf{G_{10}}$	$\mathbf{R_L}$	R_0	$\mathbf{R_9}$	G_d	$\mathbf{R_2}$	R_3	R_4	$\mathbf{R_5}$	R_6	$\mathbf{R_1}$	I_{u1}	I_{i4}	I_{ugs}	I^*	I_0
1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
0	**1**	0	1	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0
0	0	**1**	1	0	0	0	0	0	0	0	1	0	-1	0	0	0	0	0	0
0	0	0	**-1**	1	0	0	0	0	-1	0	0	0	0	0	1	1	0	0	0
0	0	0	0	**-1**	0	0	0	1	0	0	0	0	0	0	0	-1	1	1	0
0	0	0	0	0	**1**	0	0	-1	0	0	0	0	0	0	0	0	-1	-1	0
0	0	0	0	0	0	0	**1**	1	-1	-1	1	0	-1	0	1	0	1	1	0
0	0	0	0	0	0	0	0	0	**1**	1	-1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	**-1**	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	**-1**	0	0	0	0	-1

Table 3.2-8 Upper triangular step matrix $\boldsymbol{Q_1}$ *of example 3*

The steps used to arrive at Q_1 from Q_0 are as follows:

1. From the first column following group 0 edges(column V_{u2}, j=1), a row operation is done using row 1 to zero all non-zero entries below the entry at (i=1, j=1). If the entry at (1, 1) was zero, the first row with a non-zero entry at (i>1, 1) would be swapped with row 1 and then row operations would be employed to zero all non-zero entries below the new swapped entry at (1, 1). In this case, row 1 is added to row 2.

2. For column 2, row 2 is added to row 6.

3. Similar row operations are done on row 3 and 4.

4. For column 6 (column R_0) there is no non-zero entry from row 6 downwards. The first non-zero entry of column 7 from row 6 downwards is picked. If this was in row i other than i=6, row 6 and row i would be swapped and the new row 6 would be used for row operation to zero all non-zero entries below (6, 7).

5. Similar operations are carried out until the row index i corresponds to the column j picked reaches its maximum value (9). Row operations done in sequence are:

   row 1 + row 2, row 2 + row 6, row 3 + row 6, row 4 + row 6

5. Tree/cotree partitioning of node incidence matrix A_0

This is done by exchanging columns on Q_1 to bring all tree edges together to obtain an upper triangular matrix Q_2 in Table 3.2-9 and simultaneously carrying out the same column exchanges on A_0 to obtain A_1 in Table 3.2-10 in the following manner:

```
for (row i=1; i<10; i++)
  if entry at row i, column i of Q_1 is zero
        Column i of Q_1 is swapped with a column between column j=i+1 and j=19
        which has the first non-zero element in row i. The same column exchange is
        done on A_0.
```

V_0	V_{u2}	V_{i3}	C_8	G_{10}	R_L	R_9	R_2	R_5	R_1	R_3	R_4	G_d	R_6	R_0	I_{u1}	I_{i4}	I_{ugs}	I^*	I_0
1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0	0	1	0	-1	0	0	0	0	0	0
0	0	0	-1	1	0	0	-1	0	0	0	0	0	0	0	1	1	0	0	0
0	0	0	0	-1	0	0	0	0	0	0	0	1	0	0	0	-1	1	1	0
0	0	0	0	0	1	0	0	0	0	0	0	-1	0	0	0	0	-1	-1	0
0	0	0	0	0	0	1	-1	0	0	-1	1	1	-1	0	1	0	1	1	0
0	0	0	0	0	0	0	1	0	0	1	-1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	-1	1	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	-1

Table 3.2-9 Upper triangular matrix Q_2 of example 3

Tree edge matrix A_{1T}										Cotree edge matrix $\mathbf{A_{1CT}}$									
V_0	V_{u2}	V_{i3}	C_8	G_{10}	R_L	R_9	R_2	R_5	R_1	R_3	R_4	G_d	R_6	R_0	I_{u1}	I_{i4}	I_{ugs}	I^*	I_0
1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0
0	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	-1	1	0	0	-1	0	0	0	0	0	0	0	1	1	0	0	0
0	0	0	0	-1	0	0	0	0	0	0	0	1	0	0	0	-1	1	1	0
0	0	0	0	0	1	0	0	0	0	0	0	-1	0	0	0	0	-1	-1	0
0	0	-1	0	0	0	1	0	0	0	-1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	1	-1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	-1	1	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	-1

Table 3.2-10 Tree/cotree partitioned node incidence matrix $\mathbf{A_1}$ of example 3

It can be verified with Figure 3.2-1 that the branch and node voltages are linked with

$$\mathbf{v_T = A_{1T}{}^T v_{node}} \qquad (3.2\text{-}5)$$

$$\mathbf{v_{CT} = A_{1CT}{}^T v_{node}} \qquad (3.2\text{-}6)$$

6. Derivation of the connection equation parameters

From Table 3.2-10 and section 2.2, (2.2-5),

$$\mathbf{D = A_{1T}{}^{-1}\, A_{1CT}}$$

Alternatively, **D** will equal the cotree partition of $\mathbf{Q_2}$ if row operations are used to change its tree partition to a unity matrix. With section 2.2, (2.2-4), the connection matrix parameters are:

$$\mathbf{a} = \begin{bmatrix} \mathbf{a}_t \\ \mathbf{a}_{ct} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_t \\ \mathbf{b}_{ct} \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} i_{I_0} \\ v_{V_0} \end{bmatrix}$$

$$\mathbf{a}_t = \begin{bmatrix} i_{Vu2} \\ i_{Vi3} \\ i_{C8} \\ i_{G10} \\ i_{RL} \\ i_{R9} \\ i_{R2} \\ i_{R5} \\ i_{R1} \end{bmatrix}, \mathbf{a}_{ct} = \begin{bmatrix} v_{R3} \\ v_{R4} \\ v_{Gd} \\ v_{R6} \\ v_{R0} \\ v_{Iu1} \\ v_{Ii4} \\ v_{Iugs} \\ v_{I*} \end{bmatrix} \qquad \mathbf{b}_t = \begin{bmatrix} v_{Vu2} \\ v_{Vi3} \\ v_{C8} \\ v_{G10} \\ v_{RL} \\ v_{R9} \\ v_{R2} \\ v_{R5} \\ v_{R1} \end{bmatrix}, \quad \mathbf{b}_{ct} = \begin{bmatrix} i_{R3} \\ i_{R4} \\ i_{Gd} \\ i_{R6} \\ i_{R0} \\ i_{Iu1} \\ i_{Ii4} \\ i_{Iugs} \\ i_{I*} \end{bmatrix}$$

$$\mathbf{L}_{11} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & 1 & 0 & -1 & 0 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 1 & 0 & -1 & 0 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & -1 & 0 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$$\mathbf{L}_{12} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0
\end{bmatrix}^T$$

3.3. Conclusions

This section has considered the practical issues of implementing a procedure to derive automatically the connection equation of a CUT in terms of two opposing strategies: pre-selection and post-selection strategies. The post-selection strategy generates all the possible trees of the CUT and then employs the tree selection rules in Table 2.2-1 to select a subset of all these trees. This subset of all the possible trees can then be used to derive their corresponding connection equations for use in the FDA. Since a large number of possible trees can easily result even for a relatively small circuit, post-selection strategy is not feasible as it implies an unreasonable drain on computing resources and is limited by circuit size. However, to appreciate the limit on circuit size this resource problem imposes and to justify the choice of the pre-selection strategy, codes have been written to implement the post-selection strategy and it has been briefly investigated with a n-stage RLC-ladder network.

The pre-selection strategy employs modified rules from the tree selection rules in Table 2.2-1 and additional rules have also been proposed to select a tree that will result in computationally efficient, high sparsity connection matrices. These rules, together with some specific operations and matrix manipulations, constitute an Optimal Tree Generation (OTG) Procedure[78] which selects an optimal tree as well as deriving its corresponding connection equation. This OTG procedure has been applied to a fictitious circuit in Example 3.

4. Test Point Selection

After deriving the connection equation (2.2-4), a set of test points are chosen subjected to the input Diagnosis Depth requirement by observing two Essential Rules[75], and if it is necessary, a Supplementary Rule. Since the minimum number of test points required is one plus the Diagnosis Depth[75][83], 1+t, the application of the Supplementary Rule depends on the number of test points m, selected with the Essential Rules being less than the minimum. On the other hand, Iterative Test Point Compaction will be needed if m is more than the minimum.

Essential Rules 1 and 2 originate from the consideration of the topology and diagnosability of the CUT respectively. Their effects manifest themselves in the column entries of the connection matrix $\mathbf{L}_{11}$ (2.2-1). Application of these Rules thus involves inspecting the $\mathbf{L}_{11}$ matrix on a per column basis. The **b** vector elements corresponding to those columns which fail the inspection are picked as essential test points.

As tree branch voltages and cotree branch currents constitute the output variable vector **b**, an essential test point can be either a voltage across or current through a circuit component. This fact poses two measurement problems for testing purpose. It is impractical to measure current flowing into/out of a pin or flowing between pins on a chip. If the test strategy allows the measurement of both currents and voltages, additional circuitry will have to be incorporated into the CUT to make those cotree branch currents selected as test points proportional to some branch voltages for measurement. If the test strategy is to measure voltages only and there are cotree branch currents selected as test points, the CUT will not be diagnosable.

It is only practical to measure node voltages at pins on a chip with a common reference point. The essential test points selected are therefore measured indirectly via node voltages with (3.2-5). (3.2-5) is also used with (3.2-6) to implement a Minimum Node Mapping Scheme to select supplementary test points (tree and cotree branch voltages). This mapping process will stop when the number of essential and supplementary test points equals the limit set by the Diagnosis Depth. (3.2-6) is also used in mapping compact test points to node voltages after compaction which replaces more than one diagnosability-related essential test point with an element of the a vector by means of the connection equation (2.2-1).

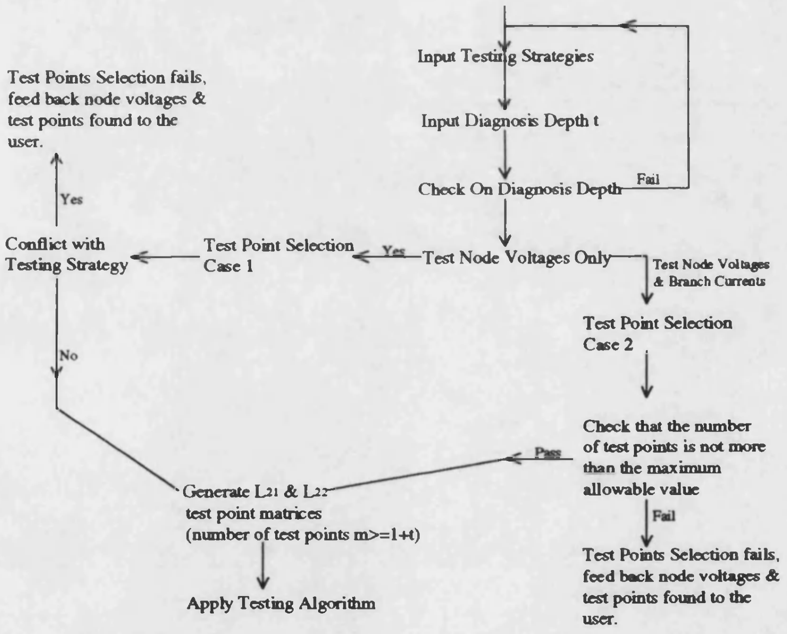
Once all the test points are selected, the test point matrices $\mathbf{L}_{21}$ and $\mathbf{L}_{22}$ in the measurement equation

$$\mathbf{y}=\mathbf{L}_{21}\mathbf{b}+\mathbf{L}_{22}\mathbf{u} \qquad (4\text{-}1)$$

y: Test point vector.

can be derived from $\mathbf{L}_{11}$ and the connection equation (2.2-1).

The flow chart for the part of the diagnostic program on Test Point Selection for a non-hierarchical circuit is depicted in Figure 4-1.



Input Testing Strategies
Input Diagnosis Depth t
Check On Diagnosis Depth
Fail
Test Node Voltages Only
Yes
Test Node Voltages & Branch Currents
Test Point Selection Case 1
Conflict with Testing Strategy
Yes
Test Points Selection fails, feed back node voltages & test points found to the user.
No
Test Point Selection Case 2
Check that the number of test points is not more than the maximum allowable value
Pass
Fail
Test Points Selection fails, feed back node voltages & test points found to the user.
Generate L21 & L22 test point matrices (number of test points m>=1+t)
Apply Testing Algorithm

Figure 4-1 Test Point Selection part of the diagnostic program in the case of a non-hierarchical circuit

4.1. *Essential Rules*

1. Topology Rule

There are four topological connections which result in identical L_{11} columns regardless of the tree used to derive the connection equation parameters. The first case of connection was mentioned by Wey[75] whereas the others have been reported in [78]. In the first and second case, all but one of tree voltages or cotree currents in the output variable vector **b** which correspond to each set of identical columns must be test points. The edge elements in these two topological connections are not independent sources. In case 3 and 4, an independent source is always involved and all the entries on the L_{11} columns and their respective rows are zero. Since **b** does not include any independent source as an element, the rule becomes: All tree voltages or cotree currents in **b** which correspond to each set of identical columns must be test points. These connections are as shown below in Figure 4.1-1:

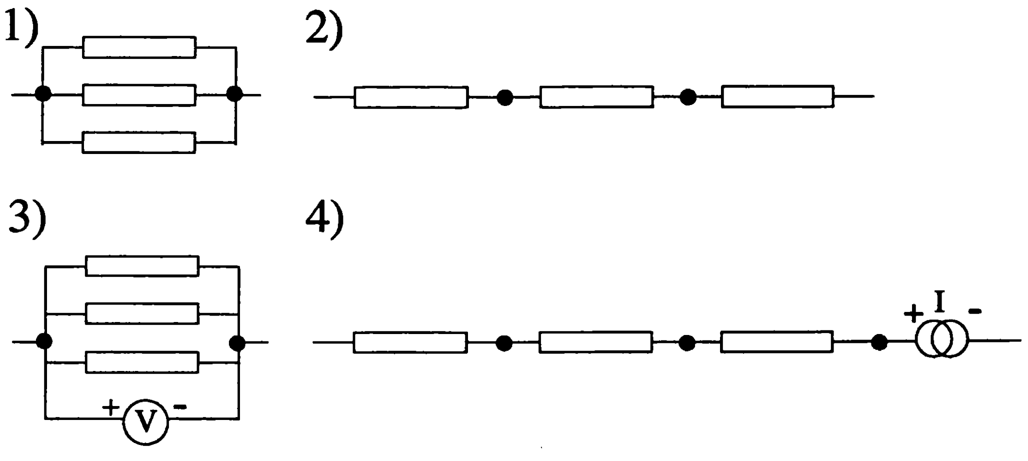


Figure 4.1-1 Topological Connections leading to identical columns in L_{11} connection matrix

1) Parallel Cotree Elements (Figure 3.2-1)

As these elements have the same voltage across them and the total current through them is known, the measurement quantity has to be the currents through all but one of the parallel cotree elements to diagnose any fault.

2) Series Tree Elements

Since these elements have the same current through them and the total voltage across them is known, the measurement quantity has to be the voltage across each but one of the series tree elements to diagnose any fault.

3) Cotree elements in parallel with an independent voltage source on the tree (Figure 3.2-1).

4) Tree elements in series with an independent current source on the cotree (Figure 3.2-1).

The intuitive reasons behind cases 3) and 4) are similar to those of cases 1) and 2) respectively. Additionally, with the assumption that an independent source is not faulty, the voltage across an independent current source and current through an independent voltage source must not be test points.

The unique features of all zero entries on the respective L_{11} rows and columns are due to the inclusion of the independent voltage source and current source in the topological connections 3) and 4) respectively. All the cotree elements in case 3) have their voltages equal to that of the independent voltage source and these cotree voltages are therefore independent of all the tree voltages. By the same argument, all the tree elements in case 4) have their currents equal to that of the independent current source and these tree currents are therefore independent of all the cotree currents. These facts manifest themselves in all zero entries on the respective L_{11} rows.

The intuitive reason for the zero column entries case is not as obvious as its counterpart. R_0 and R_1 in Figure 3.2-1 are used as an example to explain the proof behind cases 3) and 4) respectively. For the cotree element R_0, i_{R0} is independent of all the tree currents as v_{R0} depends only on the independent voltage source V_0. Likewise, v_{R1} is independent of all the cotree voltages as i_{R1} depends only on the independent current source I_0. Therefore, all the tree currents and cotree voltages are independent of i_{R0} and v_{R1} respectively. These facts result in all zero entries on the L_{11} columns corresponding to i_{R0} and v_{R1}.

The reasons behind the topology rule from the point of view of the Fault Diagnosis Algorithm will be discussed in Chapter 5.2. Applying the Topology Rule to Example 3 (Chapter 3.2), the essential test points chosen are in Table 4.1-1.

Case 1	Case 3	Case 4
i1*, iG_d	iR_0	vR_1

Table 4.1-1 Topology-related test points of example 3

The columns entries and **b** vector elements corresponding to these cases are depicted in Figure 4.1-2.

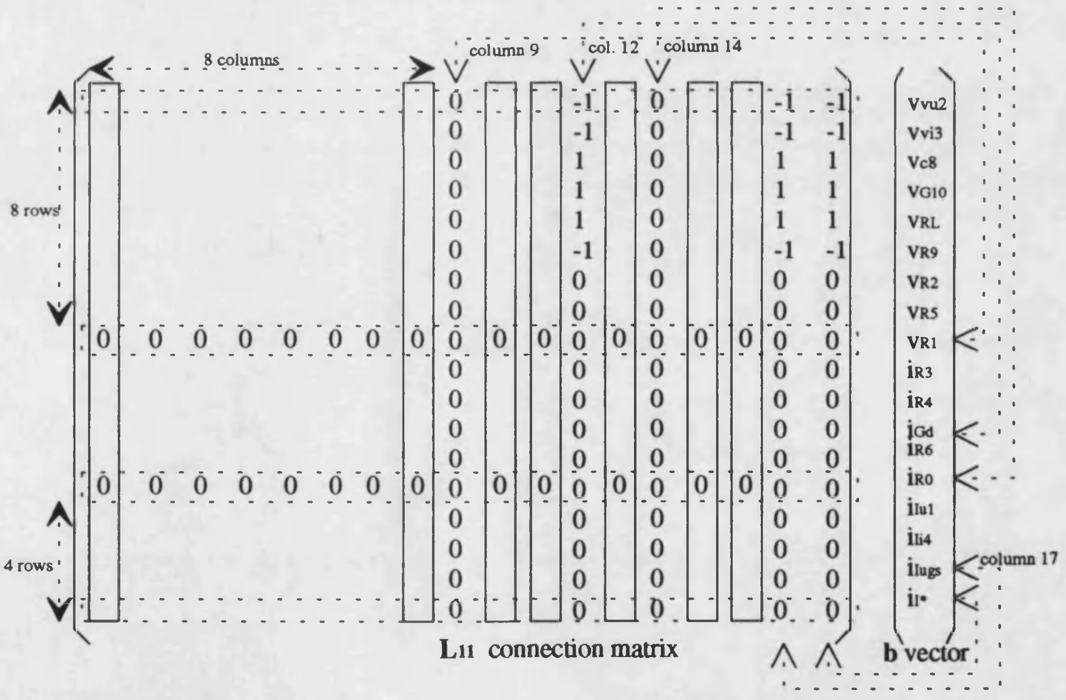


Figure 4.1-2 Correspondence between b vector elements and L_{11} matrix columns

2. Diagnosability Rule

Diagnosability dictates that a **b** vector element is a test point if its corresponding L_{11} column has its number of non-zero entries less than or equal to the number of testee components, which must be at least 1+t for a t-diagnosable non-hierarchical circuit[75]. t here is the Diagnosis Depth of the CUT. The final number of testee group components must equal the number of test points selected by all selection rules used. The underlying reasons for this rule are to deal with the necessary condition for the existence of a Pseudo Circuit for the CUT. This will be discussed in Chapter 5.2.

After applying the Topology and Diagnosability Rule to Example 3 for 1-diagnosability, the essential test points selected are shown in Table 4.1-2.

	Case 3	Case 4	Case 1	Not related to topology
Topology Rule	i_{R0}	v_{R1}	i_{I*}, i_{Gd}	
Diagnosability Rule	i_{R0}	v_{R1}		v_{R2}, v_{R5}, i_{I4}

Table 4.1-2 All essential test points of example 3

In this case all diagnosability-related test points correspond to $\mathbf{L_{11}}$ columns with 0, 1, or 2 non-zero entries. It is useful to notice that the Diagnosability Rule also picks up topology-related test points in topological connections 3) and 4). This is because all the entries on the $\mathbf{L_{11}}$ columns corresponding to the circuit components in these connections are zero. This overlap is exploited to simplify the coding of the diagnostic program by merging topological connection 3) and 4) together as a single case and always selecting all but one of the **b** vector elements corresponding to each set of identical $\mathbf{L_{11}}$ columns irrespective of the types of topological connection. If the topological connection is case 3) or 4), the test point which is left out by applying the Topology Rule will always be picked by the Diagnosability Rule.

The number of essential test points increases from 7 with a Diagnosis Depth of 1 to all the **b** vector elements with a Diagnosis Depth of 6. Iterative Test Point Compaction is always needed instead of applying the Supplementary Rule for the CUT in Example 3 for all possible values of the Diagnosis Depth. Given the sparse nature of $\mathbf{L_{11}}$, the Supplementary Rule is rarely needed and it is devised for completeness to cover all possible cases of circuits.

4.2. Supplementary Rule

Branch voltages are selected to supplement the essential test points until the diagnosability requirement (1+t for a non-hierarchical circuit) is met. The selection criterion is to pick those branches which will result in the minimum number of test nodes for voltage measurements. A minimum node mapping scheme is proposed by using this criterion with the mapping rules based on (3.2-5) and (3.2-6).

This scheme implements the criterion by assigning a preference number 0, 1 and 2 in descending priority to every remaining tree branch voltage according to the number of its corresponding node voltages by examining (3.2-5) and how many of these node voltages have already been mapped in the process of choosing tree branch voltages as essential test points. For example, if all its node voltages have been mapped, its preference number will be 0. If one of its two corresponding node voltages has been mapped, its preference number will be 1. When all its corresponding node voltages are available for mapping, its preference number will be the number of non-zero entries on its respective row in the $\mathbf{A}_{1T}{}^{T}$ matrix. This number is either 1 or 2 depending on whether the tree element with its voltage to be mapped is or is not connected to the reference node.

If the diagnosability requirement is still not met after choosing all the remaining tree branch voltages, cotree branch voltages will be used until the requirement is met. The reason for choosing all remaining tree branch voltages before using any cotree branch voltages is to make the $\mathbf{L}_{22}$ matrix as sparse as possible for high computation efficiency. If all the node voltages have been mapped by the tree branch voltages, the cotree branch voltages will be selected arbitrarily. On the other hand, if there are still unmapped node voltages, the minimum node mapping scheme will be applied to every cotree branch voltage to establish the selection order by examining (3.2-6) instead of (3.2-5) and how many of the corresponding node voltages have already been selected in the mapping of all the tree branch voltages.

From Table 3.2-10, (3.2-5) and (3.2-6), the relationship between the branch and node voltages of Example 3 are

$$\mathbf{v}_T = \begin{bmatrix} v_{V0} \\ v_{Vu2} \\ v_{Vi3} \\ v_{C8} \\ v_{G10} \\ v_{RL} \\ v_{R9} \\ v_{R2} \\ v_{R5} \\ v_{R1} \end{bmatrix} = \mathbf{A}_{1T}^{T}\,\mathbf{v}_{node} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} v_5 \\ v_6 \\ v_4 \\ v_2 \\ v_9 \\ v_{10} \\ v_8 \\ v_3 \\ v_1 \\ v_7 \end{bmatrix}$$

$$\mathbf{v}_{CT} = \begin{bmatrix} v_{R3} \\ v_{R4} \\ v_{Gd} \\ v_{R6} \\ v_{R0} \\ v_{Iu1} \\ v_{Ii4} \\ v_{Iugs} \\ v_{I*} \\ v_{I0} \end{bmatrix} = \mathbf{A}_{1CT}^{T}\,\mathbf{v}_{node} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} v_5 \\ v_6 \\ v_4 \\ v_2 \\ v_9 \\ v_{10} \\ v_8 \\ v_3 \\ v_1 \\ v_7 \end{bmatrix}$$

It can be seen that all the rows in $\mathbf{A}_{1T}{}^{T}$ and $\mathbf{A}_{1CT}{}^{T}$ has at most two non-zero entries. This is generally true since an edge connects only two nodes and implies that every branch voltage can be mapped to one or two node voltages in the vector $\mathbf{v}_{node}$. Two mapping rules are therefore deduced from this implication. Moreover, as independent sources in $\mathbf{v}_T$ and $\mathbf{v}_{CT}$ are elements of the stimulus vector **u** in the connection equation (2.2-4), they are excluded from the mapping process.

The mapping rules are as follows:

1. For a branch element (e.g. R_0) connected to the reference node, its corresponding row in the $\mathbf{A}_{1T}^T$ or $\mathbf{A}_{1CT}^T$ matrix has one non-zero entry. The mapped node voltage of the branch element is the node voltage which corresponds to the non-zero entry in the vector $\mathbf{v}_{node}$. In this case, the magnitude of the branch voltage and mapped node voltage are the same. This is illustrated in Figure 4.2-1.

VRO=V5

$$\begin{bmatrix} v_{R3} \\ v_{R4} \\ v_{Gd} \\ v_{R6} \\ \boxed{v_{R0}} \\ v_{Iu1} \\ v_{Ii4} \\ v_{Iugs} \\ v_{I*} \\ v_{I0} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} v_5 \\ v_6 \\ v_4 \\ v_2 \\ v_9 \\ v_{10} \\ v_8 \\ v_3 \\ v_1 \\ v_7 \end{bmatrix}$$

Figure 4.2-1 Mapping of a branch voltage to a node voltage

2. For a branch element (e.g. R_3) not connected to the reference node, its corresponding row in the $\mathbf{A}_{1T}^T$ or $\mathbf{A}_{1CT}^T$ matrix has two non-zero entries. The two mapped node voltages of the branch element are the node voltages which correspond to the non-zero entries in the vector $\mathbf{v}_{node}$. In this case the branch voltage is a linear combination of the two mapped tree node voltages defined by the equation corresponding to the branch element in (3.2-5) or (3.2-6). This is illustrated in Figure 4.2-2.

VR3=V3-V8

$$\begin{bmatrix} \boxed{v_{R3}} \\ v_{R4} \\ v_{Gd} \\ v_{R6} \\ v_{R0} \\ v_{Iu1} \\ v_{Ii4} \\ v_{Iugs} \\ v_{I*} \\ v_{I0} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} v_5 \\ v_6 \\ v_4 \\ v_2 \\ v_9 \\ v_{10} \\ v_8 \\ v_3 \\ v_1 \\ v_7 \end{bmatrix}$$

Figure 4.2-2 Mapping of a branch voltage to two node voltages

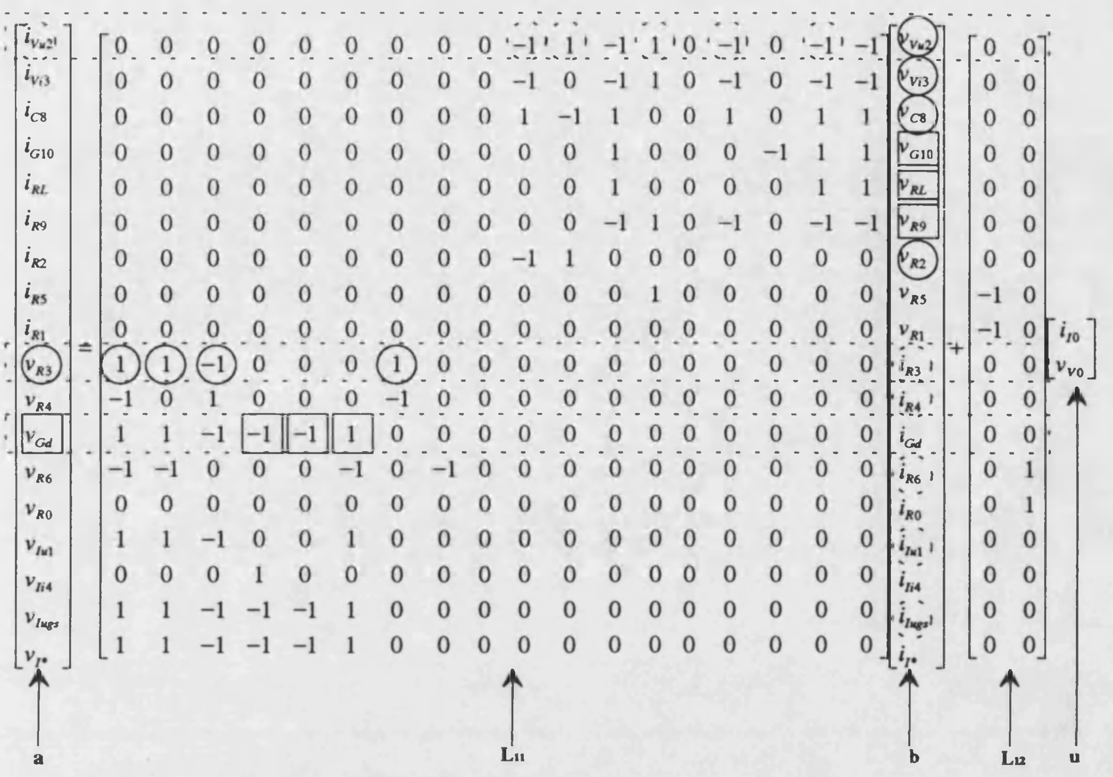
4.3. *Iterative Test Point Compaction*

Test Point Compaction is used to reduce the test point number down to the diagnosability limit which is set at 1+t for a non-hierarchical circuit initially. In subsequent compaction cycles this limit is set to the number of incomplete compact or non-compact test points in the previous compaction cycle. Compaction is only carried out on test points selected with the Diagnosability Rule alone as compaction on topology-related test points does not result in a real decrease in test point number. Compaction is done by using an element of the **a** vector to replace more than one diagnosability-related essential test point by means of the connection equation (2.2-1). For example, with reference to Table 4.3-1 on page 59 , the additional essential test points found after the second application of the Diagnosability Rule can be replaced by the **a** vector elements i_{Vu2}, v_{R3} and v_{Gd}. This is shown graphically in Figure 4.3-1.

◌ : Indicate compaction from i_{Iugs}, i_{Iu1}, i_{R4}, i_{R6} & i_{R3} to i_{Vu2}

○ : Indicate compaction from v_{Vu2}, v_{Vi3}, v_{C8} & v_{R2} to v_{R3}

□ : Indicate compaction from v_{R9}, v_{G10} & v_{RL} to v_{Gd}



$$\underset{\uparrow\ \mathbf{a}}{\begin{bmatrix} i_{Vu2} \\ i_{Vi3} \\ i_{C8} \\ i_{G10} \\ i_{RL} \\ i_{R9} \\ i_{R2} \\ i_{R5} \\ i_{R1} \\ \enclose{circle}{v_{R3}} \\ v_{R4} \\ \boxed{v_{Gd}} \\ v_{R6} \\ v_{R0} \\ v_{Iu1} \\ v_{Ii4} \\ v_{Iugs} \\ v_{I\bullet} \end{bmatrix}} = \underset{\uparrow\ \mathbf{L_{11}}}{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & 1 & 0 & -1 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 1 & 0 & -1 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & -1 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \enclose{circle}{1} & \enclose{circle}{1} & \enclose{circle}{-1} & 0 & 0 & 0 & \enclose{circle}{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & -1 & \boxed{-1} & \boxed{-1} & \boxed{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}} \underset{\uparrow\ \mathbf{b}}{\begin{bmatrix} \enclose{circle}{v_{Vu2}} \\ \enclose{circle}{v_{Vi3}} \\ \enclose{circle}{v_{C8}} \\ \boxed{v_{G10}} \\ \boxed{v_{RL}} \\ \boxed{v_{R9}} \\ \enclose{circle}{v_{R2}} \\ v_{R5} \\ v_{R1} \\ i_{R3} \\ i_{R4} \\ i_{Gd} \\ i_{R6} \\ i_{R0} \\ i_{Iu1} \\ i_{Ii4} \\ i_{Iugs} \\ i_{I\bullet} \end{bmatrix}} + \underset{\uparrow\ \mathbf{L_{12}}}{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -1 & 0 \\ -1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}} \underset{\uparrow\ \mathbf{u}}{\begin{bmatrix} i_{I0} \\ v_{V0} \end{bmatrix}}$$

Figure 4.3-1 Illustration on the Test Point Compaction Process

With Example 3, initial compaction is from 7 to 2 test points and from 18 to 7 with a Diagnosis Depth of 1 and 6 respectively. However, the initial compaction target is often not reached. Subsequent re-application of the Diagnosability Rule with the number of incomplete compact/non-compact test points replacing the "1+t" value used in the first application of the Diagnosability Rule often results in selecting additional test points, which require a second compaction. This cycle of compaction-check-apply the Diagnosability Rule-check goes on iteratively until the compaction is complete, unnecessary or all the **b** vector elements are selected by the Diagnosability Rule.

The breakdown of the Test Point Selection Procedure under two testing strategies are shown in Figures 4.3-2 and 4.3-3.

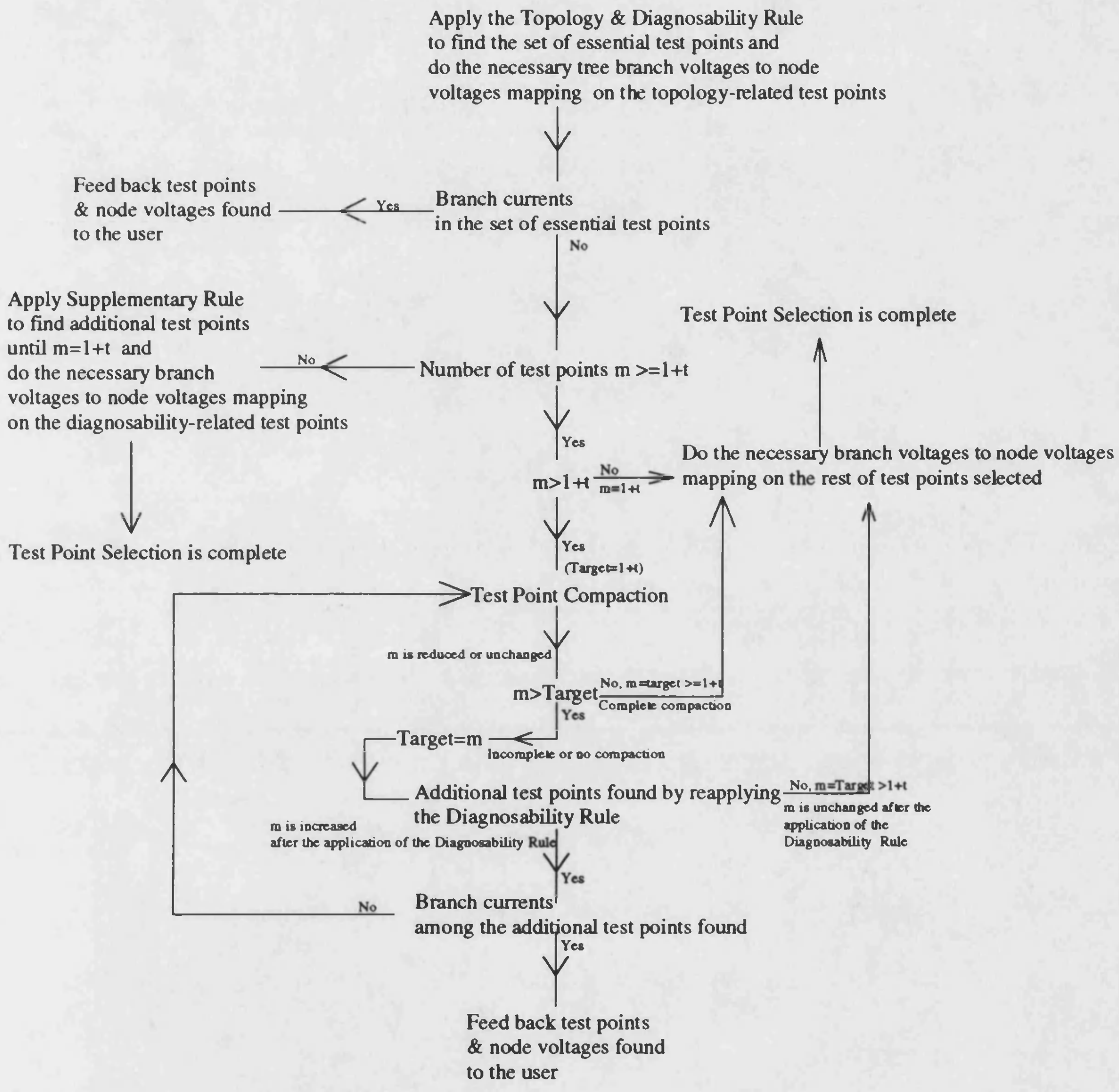

Apply the Topology & Diagnosability Rule
to find the set of essential test points and
do the necessary tree branch voltages to node
voltages mapping on the topology-related test points
Feed back test points
& node voltages found
to the user
Yes
Branch currents
in the set of essential test points
No
Apply Supplementary Rule
to find additional test points
until m=1+t and
do the necessary branch
voltages to node voltages mapping
on the diagnosability-related test points
No
Number of test points m >=1+t
Test Point Selection is complete
Yes
Do the necessary branch voltages to node voltages
mapping on the rest of test points selected
m>1+t
No
m=1+t
Test Point Selection is complete
Yes
(Target=1+t)
Test Point Compaction
m is reduced or unchanged
m>Target
No, m=target >=1+t
Complete compaction
Yes
Target=m
Incomplete or no compaction
Additional test points found by reapplying
the Diagnosability Rule
No, m=Target >1+t
m is unchanged after the
application of the
Diagnosability Rule
m is increased
after the application of the Diagnosability Rule
Yes
No
Branch currents
among the additional test points found
Yes
Feed back test points
& node voltages found
to the user

Figure 4.3-2 Test Point Selection Case 1 for a non-hierarchical circuit

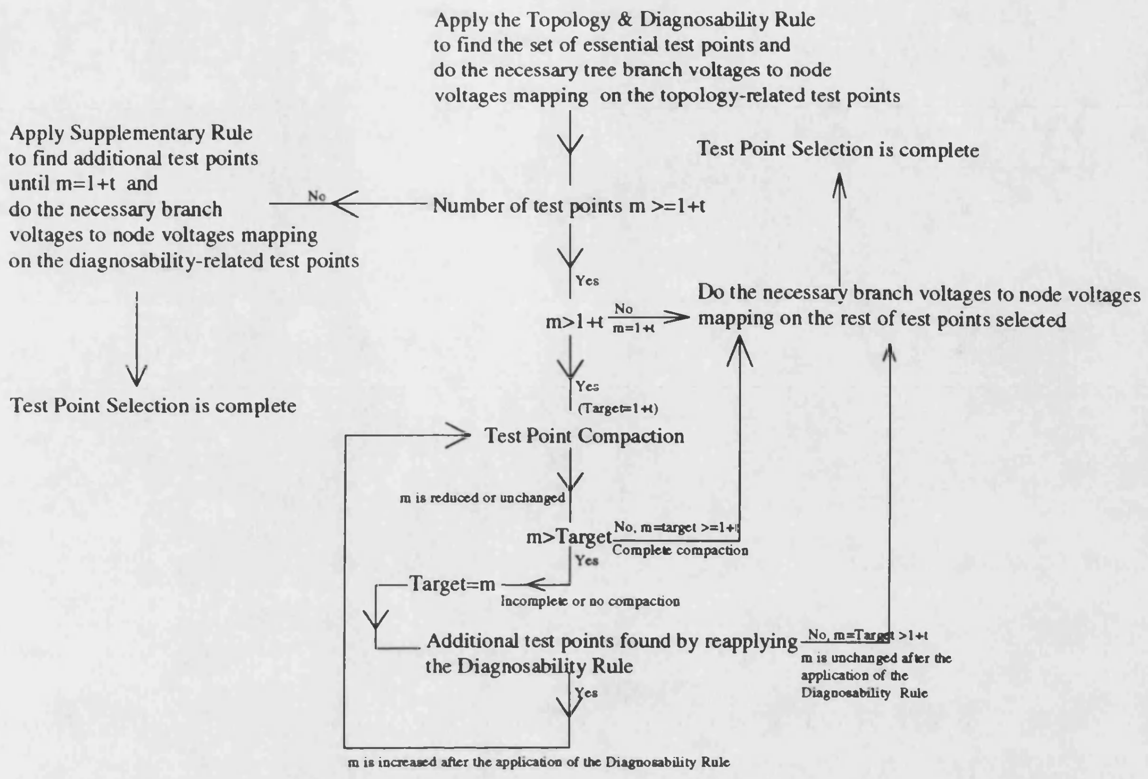

Apply the Topology & Diagnosability Rule
to find the set of essential test points and
do the necessary tree branch voltages to node
voltages mapping on the topology-related test points
Apply Supplementary Rule
to find additional test points
until m=1+t and
do the necessary branch
voltages to node voltages mapping
on the diagnosability-related test points
No
Number of test points m >=1+t
Test Point Selection is complete
Yes
m>1+t
No
m=1+t
Do the necessary branch voltages to node voltages
mapping on the rest of test points selected
Yes
(Target=1+t)
Test Point Selection is complete
Test Point Compaction
m is reduced or unchanged
m>Target
No, m=target >=1+t
Complete compaction
Yes
Target=m
Incomplete or no compaction
Additional test points found by reapplying
the Diagnosability Rule
No, m=Target >1+t
m is unchanged after the
application of the
Diagnosability Rule
Yes
m is increased after the application of the Diagnosability Rule

Figure 4.3-3 Test Point Selection Case 2 for a non-hierarchical circuit

With reference to Figure 4.3-3, the Iterative Test Point Compaction process for Example 3 with a Diagnosis Depth of 1 is shown in Table 4.3-1.

	Initial State	After first compaction	After the second application of the Diagnosability Rule	After second compaction	After the third application of the Diagnosability Rule
Target	2	2	7	7	9
No. of test points m	7	7	18	9	9
Topology-related test points	v_{R1}, i_{R0}, i_{I*} & i_{Gd}	v_{R1}, i_{R0}, i_{I*} & i_{Gd}	v_{R1}, i_{R0}, i_{I*} & i_{Gd}	v_{R1}, i_{R0}, i_{I*} & i_{Gd}	v_{R1}, i_{R0}, i_{I*} & i_{Gd}
Test points for compaction	v_{R2}, v_{R5} & i_{Li4}	None	v_{R2}, v_{R5} & i_{Li4}	None	None
Additional test points found for compaction	Not applicable	Not applicable	v_{Vu2}, v_{Vi3}, v_{C8}, v_{G10}, v_{RL}, v_{R9}, i_{R3}, i_{R4}, i_{R6}, i_{Iu1} & i_{Iugs}	Not applicable	None
No. of additional test points found	Not applicable	Not applicable	11	Not applicable	None
No compaction on these test points	Not applicable	v_{R2}, v_{R5} & i_{Li4}	None	v_{R5} & i_{Li4}	v_{R5} & i_{Li4}
Compact test points	Not applicable	Not applicable	None	$i_{Vu2} \Leftarrow i_{Iugs}$, i_{Iu1} i_{R4}, i_{R6} & i_{R3} $v_{R3} \Leftarrow v_{Vu2}$, v_{Vi3}, v_{C8} & v_{R2} $v_{Gd} \Leftarrow v_{G10}$, v_{RL} & v_{R9}	i_{Vu2}, v_{R3} & v_{Gd}

Table 4.3-1 Iterative Test Point Compaction on example 3 with a Diagnosis Depth of 1

It can be seen from the table that a third compaction is unnecessary although the second compaction is incomplete. Test Point Selection is completed after the third application of the Diagnosability Rule to confirm that no more additional test points are needed.

4.4. *Construction of Test Point Matrices*

The test points, mapped node voltages and the test point matrices $\mathbf{L}_{21}$ and $\mathbf{L}_{22}$ for Example 3 with a Diagnosis Depth of 1, 2 and 5 to 8 are:

Test Node	v1	v7	v5	v8	v3	v9	v10		
TP	vR1	iI*	iGd	vR5	iR0	iIi4	iVu2	vR3	vGd

$$\mathbf{y} = \begin{bmatrix} v_{R1} \\ i_{I*} \\ i_{Gd} \\ v_{R5} \\ i_{R0} \\ i_{Ii4} \\ i_{Vu2} \\ v_{R3} \\ v_{Gd} \end{bmatrix}, \mathbf{L}_{22} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

$$\mathbf{L}_{21} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & 1 & 0 & -1 & 0 & -1 & -1 \\ 1 & 1 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The construction of each row of the test point matrices $\mathbf{L}_{21}$ and $\mathbf{L}_{22}$ follows the sequence of the elements in the test point vector **y**. This sequence is related to the order of application of the Test Point Selection Rules. The first group picked is all the topology-related test points (v_{R1}, i_{I*} and i_{Gd}) except for one test point (i_{R0}) in the merged group of topological connections 3) and 4). This is followed by all the diagnosability-related test points which will include the unpicked test point (i_{R0}) in the merged group of topological connections 3) and 4) if there are such connections in the CUT. The remaining test points in the second group will be those diagnosability-related test points (v_{R5} and i_{Ii4}) which cannot be compacted if compaction is desired. If compaction and the application of the Supplementary Rule are unnecessary, the second group of test points selected will be the last group of elements in the vector **y**. Otherwise, the final group picked will be those compact test points (i_{Vu2}, v_{R3} and v_{Gd}) if compaction is performed or those supplementary test points if compaction is not needed and the Supplementary Rule has to be used.

There are two ways to construct a row of the test point matrices with reference to Figure 4.4-1. Initially, all entries in the matrices are set to zero. If the test point (v_{R1}) is a **b** vector element, the $\mathbf{L}_{21}$ row entry which has the same position index as the test point entry in the **b** vector will be set to one. If the test point (i_{Vu2}) is an element of the **a** vector, its corresponding row in the $\mathbf{L}_{11}$ and $\mathbf{L}_{12}$ matrix will be copied to the $\mathbf{L}_{21}$ and $\mathbf{L}_{22}$ row respectively. The test point in the former and latter case is always respectively an essential and compact test point. However, a supplementary test point can be a branch voltage element of either the **a** or **b** vector.

: Indicate the row to be copied from L11 to form a part of the L21 test point matrix.
: Indicate the row to be copied from L12 to form a part of the L22 test point matrix.
: Indicate a compact test point. : Indicate an essential test point.

$$
\underbrace{\begin{bmatrix} i_{Vu2}\\ i_{Vi3}\\ i_{C8}\\ i_{G10}\\ i_{RL}\\ i_{R9}\\ i_{R2}\\ i_{R5}\\ i_{R1}\\ v_{R3}\\ v_{R4}\\ v_{Gd}\\ v_{R6}\\ v_{R0}\\ v_{Iu1}\\ v_{E4}\\ v_{Iugs}\\ v_{I*} \end{bmatrix}}_{a}
=
\underbrace{\begin{bmatrix}
0&0&0&0&0&0&0&0&0&-1&1&-1&1&0&-1&0&-1&-1\\
0&0&0&0&0&0&0&0&0&-1&0&-1&1&0&-1&0&-1&-1\\
0&0&0&0&0&0&0&0&0&1&-1&1&0&0&1&0&1&1\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&-1&1&1\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&1\\
0&0&0&0&0&0&0&0&0&0&0&-1&1&0&-1&0&-1&-1\\
0&0&0&0&0&0&0&0&0&-1&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&-1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
-1&0&1&0&0&0&-1&0&0&0&0&0&0&0&0&0&0&0\\
1&1&-1&-1&-1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
-1&-1&0&0&0&-1&0&-1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&-1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&-1&-1&-1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&-1&-1&-1&1&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}}_{L_{11}}
\underbrace{\begin{bmatrix} v_{Vu2}\\ v_{Vi3}\\ v_{C8}\\ v_{G10}\\ v_{RL}\\ v_{R9}\\ v_{R2}\\ v_{R5}\\ v_{R1}\\ i_{R3}\\ i_{R4}\\ i_{Gd}\\ i_{R6}\\ i_{R0}\\ i_{Iu1}\\ i_{E4}\\ i_{Iugs}\\ i_{I*} \end{bmatrix}}_{b}
+
\underbrace{\begin{bmatrix}
0&0\\ 0&0\\ 0&0\\ 0&0\\ 0&0\\ 0&0\\ 0&0\\ -1&0\\ -1&0\\ 0&0\\ 0&0\\ 0&0\\ 0&1\\ 0&1\\ 0&0\\ 0&0\\ 0&0\\ 0&0
\end{bmatrix}}_{L_{12}}
\underbrace{\begin{bmatrix} i_{I0}\\ v_{V0} \end{bmatrix}}_{u}
$$

9th entry

$$
\underbrace{\begin{bmatrix} v_{R1}\\ i_{I*}\\ i_{Gd}\\ v_{R5}\\ i_{R0}\\ i_{E4}\\ i_{Vu2}\\ v_{R3}\\ v_{Gd} \end{bmatrix}}_{y}
\quad
\underbrace{\begin{bmatrix}
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&-1&1&-1&1&0&-1&0&-1&-1\\
1&1&-1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&1&-1&-1&-1&1&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}}_{L_{21}}
\quad
\underbrace{\begin{bmatrix}
0&0\\ 0&0\\ 0&0\\ 0&0\\ 0&0\\ 0&0\\ 0&0\\ 0&0\\ 0&0
\end{bmatrix}}_{L_{22}}
$$

Figure 4.4-1 Construction of the Test Point Matrices L_{21} and L_{22}

For a Diagnosis Depth of 3 and 4, v_{R2} is selected instead of v_{R3}, and v_2 is picked instead of v_8.

4.5. Conclusions

This section has discussed issues related to test point selection and proposed test point selection rules which are divided into two Essential Rules[75] and a Supplementary Rule. These issues are test strategies, iterative test point compaction and the construction of test point matrices. Essential Rules 1 and 2 are also called the Topology and Diagnosability Rules respectively as they have their origin from the consideration of the effects of the circuit topology and diagnosability on the column entries of the connection matrix $\mathbf{L}_{11}$ (2.2-1). Their application thus involves inspecting the $\mathbf{L}_{11}$ matrix on a per column basis. The matrix columns which fail this inspection will have their corresponding **b** vector elements selected as essential test points.

As its name suggests, the Supplementary Rule is needed to select test points to supplement the essential test points when the required minimum number of test points, one plus the Diagnosis Depth[75][83] (1+t), is more than the number of essential test points. This rule selects branch voltages, which will result in the minimum number of test nodes for voltage measurements, as test points until the requirement on the minimum number of test points is met. On the other hand, when the number of essential test points is more than the required minimum, iterative test point compaction is needed to reduce the number of test points down to the limit set by diagnosability. This limit is set to 1+t initially. To achieve compaction, diagnosability test points are replaced with elements of the **a** vector using the connection equation (2.2-1). However, the initial compaction target 1+t is often not achieved. This results in the number of teetees in the testee group being equal to the number of incomplete compact or non-compact test points, which is more than 1+t. Thus, subsequent re-application of the Diagnosability Rule, with the number of incomplete compact/non-compact test points replacing the "1+t" value used in the first application of the rule, often results in selecting additional test points, which necessitates a second compaction. This iterative cycle of compaction-check-apply the Diagnosability Rule-check goes on until the compaction is complete, unnecessary or all the **b** vector elements are selected by the Diagnosability Rule.

Once all the test points are selected, the test point matrices in the measurement equation can be derived from $\mathbf{L}_{11}$ and the connection equation (2.2-1). The selection of test points and the construction of the test point matrices have been shown with the circuit in Example 3 in Chapter 3.2.

As test points selected by the test point selection rules are always branch currents or voltages in a CUT, two measurement problems for testing purpose are foreseen: it is impractical to measure current flowing into/out of a pin or flowing between pins on a chip. If the test strategy allows the measurement of both currents and voltages, the CUT will require additional circuitry to make those branch currents, selected as test points, proportional to some branch voltages for measurement. On the other hand, if the test strategy is to measure voltages only and there are branch currents selected as test points, the CUT will not be diagnosable.

5. Basic Fault Diagnosis Algorithm

Having discussed the mathematical description of circuit topology, its implementation and test point selection with testing strategies, this section builds on these issues and proposes a testing algorithm for Non-hierarchical Fault Diagnosis which is depicted in Figure 5-2. The overall Non-hierarchical Fault Diagnosis Algorithm is shown in Figure 5-1 and all the operations performed in a test cycle is illustrated in Figure 5-3.

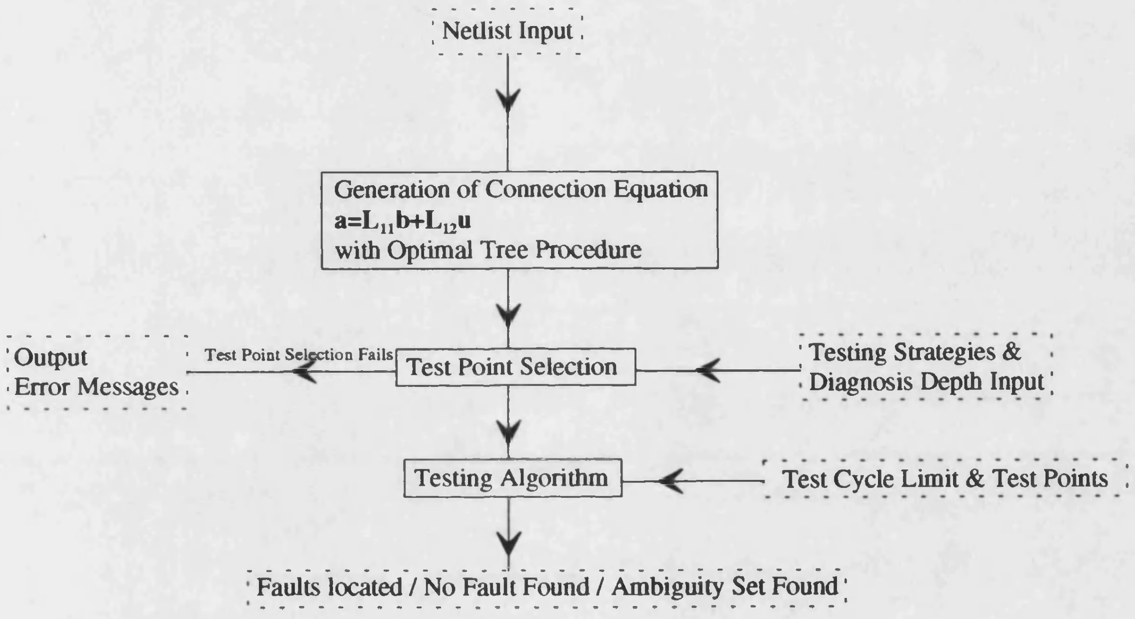

Netlist Input
Generation of Connection Equation
$a=L_{11}b+L_{12}u$
with Optimal Tree Procedure
Output
Error Messages
Test Point Selection Fails
Test Point Selection
Testing Strategies &
Diagnosis Depth Input
Testing Algorithm
Test Cycle Limit & Test Points
Faults located / No Fault Found / Ambiguity Set Found

Figure 5-1 Non-hierarchical Fault Diagnosis Algorithm

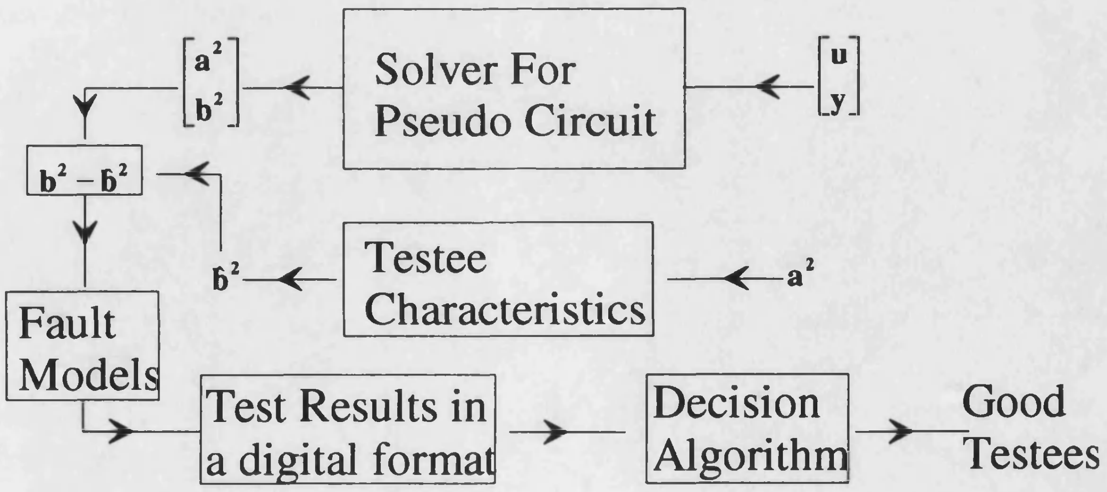

$\begin{bmatrix} a^2 \\ b^2 \end{bmatrix}$
Solver For Pseudo Circuit
$\begin{bmatrix} u \\ y \end{bmatrix}$
$b^2 - \hat{b}^2$
$\hat{b}^2$
Testee Characteristics
a^2
Fault Models
Test Results in a digital format
Decision Algorithm
Good Testees

Figure 5-3 Operations Performed in a Test Cycle

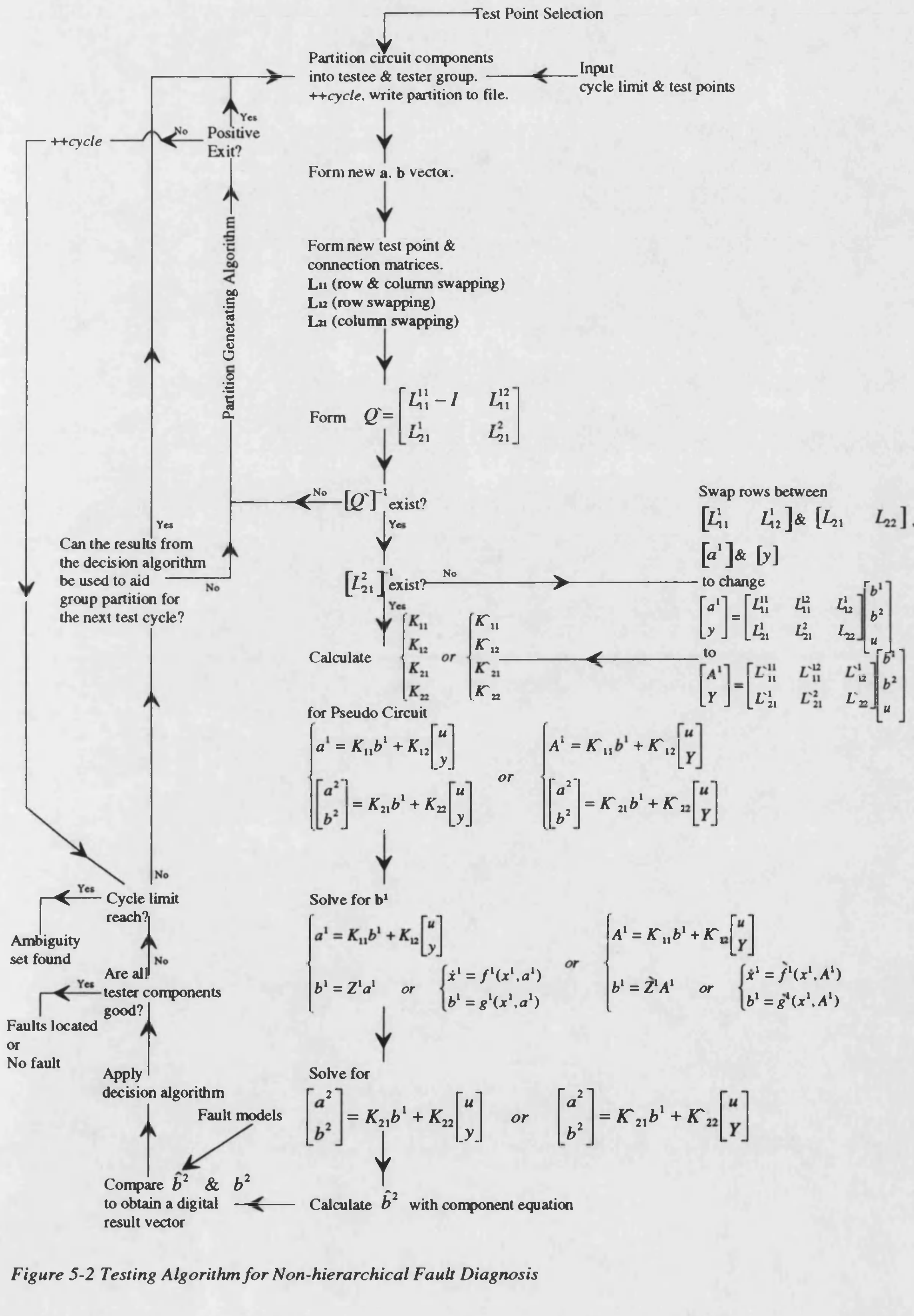

Test Point Selection
Partition circuit components into testee & tester group. ++cycle, write partition to file.
Input cycle limit & test points
Form new a, b vector.
Form new test point & connection matrices.
L11 (row & column swapping)
L12 (row swapping)
L21 (column swapping)
Form Q' =
[Q']^-1 exist?
[L^2_21]^-1 exist?
Calculate K11 K12 K21 K22 or K'11 K'12 K'21 K'22 for Pseudo Circuit
Swap rows between [L^1_11 L^1_12] & [L21 L22], [a^1] & [y] to change ... to ...
Solve for b^1
Solve for
Calculate b̂^2 with component equation
Compare b̂^2 & b^2 to obtain a digital result vector
Fault models
Apply decision algorithm
Are all tester components good?
Faults located or No fault
Cycle limit reach?
Ambiguity set found
Can the results from the decision algorithm be used to aid group partition for the next test cycle?
Partition Generating Algorithm
Positive Exit?
++cycle
Yes
No

Figure 5-2 Testing Algorithm for Non-hierarchical Fault Diagnosis

The following sub-sections serve the purposes of explaining the mathematical basis and salient features of the Non-hierarchical FDA.

5.1. Self Test Algorithm with CCM Formulation

DeCarlo[77] proposed a Component Connection Model (CCM) which consists of the connection (2.2-1) and measurement (4-1) equation, with the characteristic of components being described by a component equation

$$\mathbf{b} = \mathbf{Z}\mathbf{a} \qquad (5.1\text{-}1)$$

Z: Component transfer matrix

(5.1-1) cannot represent the characteristics of non-linear components, a pair of decoupled state equations is used instead.

$$\left.\begin{aligned} a)\cdots\cdots\cdots \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x},\mathbf{a}) \\ b)\cdots\cdots\cdots \mathbf{b} &= \mathbf{g}(\mathbf{x},\mathbf{a}) \end{aligned}\right\} \qquad (5.1\text{-}2)$$

x: Component state vector.

With the Self Test Algorithm[75][76], circuit components exclusive of the independent sources are subdivided into tester and testee group in each test cycle with the assumption that all the tester components are fault free. This is depicted in Figure 5-4 below.

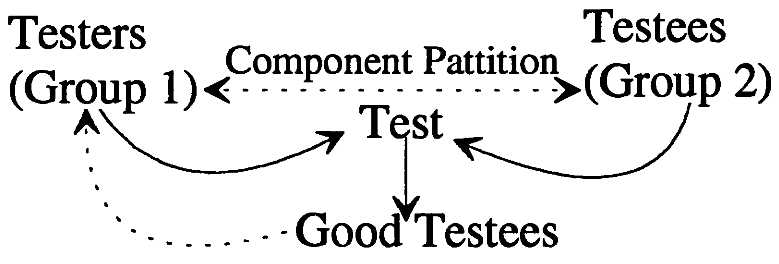


Figure 5-4 Self Test Algorithm

Within a test cycle (Figure 5-3), the characteristics of the tester components, system stimuli and test points data are then used to determine fault-free testee components with the aid of a decision algorithm such as the Exact[76], Heuristic[76] and Boolean[84]. The fault free testee components found will be included in the tester group in the next test cycle. This procedure is continued until all the tester components are good. The results of the last test cycle is thus reliable and any faulty component will be identified in the testee group.

Using the superscript ^{1} and ^{2} to denote tester and testee groups respectively. The Self Test Algorithm and CCM are coupled together and the new sets of equations for the CCM are:

$$(5.1\text{-}1)\Rightarrow \quad \left.\begin{aligned} &a)\cdots\cdots\cdots \mathbf{b}^1 = \mathbf{Z}^1\mathbf{a}^1 \\ &b)\cdots\cdots\cdots \mathbf{b}^2 = \mathbf{Z}^2\mathbf{a}^2 \end{aligned}\right\} \tag{5.1-3}$$

$$(5.1\text{-}2)\Rightarrow \quad \left.\begin{aligned} &a)\begin{cases}\cdots\cdots\cdots \dot{\mathbf{x}}^1 = \mathbf{f}^1(\mathbf{x}^1,\mathbf{a}^1) \\ \cdots\cdots\cdots \mathbf{b}^1 = \mathbf{g}^1(\mathbf{x}^1,\mathbf{a}^1)\end{cases} \\ &b)\begin{cases}\cdots\cdots\cdots \dot{\mathbf{x}}^2 = \mathbf{f}^2(\mathbf{x}^2,\mathbf{a}^2) \\ \cdots\cdots\cdots \mathbf{b}^2 = \mathbf{g}^2(\mathbf{x}^2,\mathbf{a}^2)\end{cases} \end{aligned}\right\} \tag{5.1-4}$$

$$(2.2\text{-}1)\Rightarrow \quad \begin{bmatrix}\mathbf{a}^1\\ \mathbf{a}^2\end{bmatrix} = \begin{bmatrix}\mathbf{L}_{11}^{11} & \mathbf{L}_{11}^{12} \\ \mathbf{L}_{11}^{21} & \mathbf{L}_{11}^{22}\end{bmatrix}\begin{bmatrix}\mathbf{b}^1\\ \mathbf{b}^2\end{bmatrix} + \begin{bmatrix}\mathbf{L}_{12}^{1}\\ \mathbf{L}_{12}^{2}\end{bmatrix}\mathbf{u} \tag{5.1-5}$$

$$(4\text{-}1)\Rightarrow \quad \mathbf{y} = \begin{bmatrix}\mathbf{L}_{21}^1 & \mathbf{L}_{21}^2\end{bmatrix}\begin{bmatrix}\mathbf{b}^1\\ \mathbf{b}^2\end{bmatrix} + \mathbf{L}_{22}\mathbf{u} \tag{5.1-6}$$

A Pseudo Circuit described by

$$\begin{bmatrix}\mathbf{a}^1\\ \mathbf{y}^{\mathrm{P}}\end{bmatrix} = \begin{bmatrix}\mathbf{K}_{11} & \mathbf{K}_{12}\\ \mathbf{K}_{21} & \mathbf{K}_{22}\end{bmatrix}\begin{bmatrix}\mathbf{b}^1\\ \mathbf{u}^{\mathrm{P}}\end{bmatrix} \tag{5.1-7}$$

$$\mathbf{y}^{\mathrm{P}} = \begin{bmatrix}\mathbf{a}^2\\ \mathbf{b}^2\end{bmatrix} \qquad \mathbf{u}^{\mathrm{P}} = \begin{bmatrix}\mathbf{u}\\ \mathbf{y}\end{bmatrix}$$

$$\left.\begin{aligned} \mathbf{K}_{11} &= \mathbf{L}_{11}^{11} - \mathbf{L}_{11}^{12}[\mathbf{L}_{21}^2]^{-1}\mathbf{L}_{21}^1 \\ \mathbf{K}_{12} &= \begin{bmatrix}\mathbf{L}_{12}^1 - \mathbf{L}_{11}^{12}[\mathbf{L}_{21}^2]^{-1}\mathbf{L}_{22} & \mathbf{L}_{11}^{12}[\mathbf{L}_{21}^2]^{-1}\end{bmatrix} \\ \mathbf{K}_{21} &= \begin{bmatrix}\mathbf{L}_{11}^{21} - \mathbf{L}_{11}^{22}[\mathbf{L}_{21}^2]^{-1}\mathbf{L}_{21}^1 \\ -[\mathbf{L}_{21}^2]^{-1}\mathbf{L}_{21}^1\end{bmatrix} \\ \mathbf{K}_{22} &= \begin{bmatrix}\mathbf{L}_{12}^2 - \mathbf{L}_{11}^{22}[\mathbf{L}_{21}^2]^{-1}\mathbf{L}_{22} & \mathbf{L}_{11}^{22}[\mathbf{L}_{21}^2]^{-1} \\ -[\mathbf{L}_{21}^2]^{-1}\mathbf{L}_{22} & [\mathbf{L}_{21}^2]^{-1}\end{bmatrix} \end{aligned}\right\} \tag{5.1-8}$$

is formed by eliminating $\mathbf{b}^2$ in (5.1-5) with

$$\mathbf{b}^2 = [\mathbf{L}_{21}^2]^{-1}(\mathbf{y} - \mathbf{L}_{21}^1\mathbf{b}^1 - \mathbf{L}_{22}\mathbf{u}) \qquad (5.1\text{-}9)$$

and combine (5.1-9) with the resulting equation from (5.1-5).

In a test cycle (Figure 5-3), (5.1-7) is solved with the Pseudo Circuit input $\mathbf{u}^P$ (system stimuli and test points) and with either (5.1-3)a or (5.1-4)a (characteristics of tester components and tester itself) to yield the Pseudo Circuit output $\mathbf{y}^P$, which is checked by substituting $\mathbf{a}^2$ into either (5.1-3)b or (5.1-4)b to obtain $\hat{\mathbf{b}}^2$. The test results, in the form of a percentage error vector, $(\mathbf{b}^2 - \hat{\mathbf{b}}^2)/\hat{\mathbf{b}}^2$, is converted to a digital format by comparing with threshold values from fault models or a blanket threshold value from experience(see Appendix IV). A decision algorithm is then applied to the resulting digital error vector to determine the fault-free testee components.

The component transfer matrix $\mathbf{Z}^1$ or $\mathbf{Z}^2$ will be a diagonal matrix with its diagonal entries equal to the respective component admittance or impedance. If it includes controlled sources in the partition, the respective diagonal entry for each controlled source will be zero. The proportional constant for the controlling component of each controlled source will be on one of the entries corresponding to the controlling element along the row corresponding to the controlled source if the controlling element and controlled source are in the same partition. This is illustrated in Figure 5.1-1 for a current controlled voltage source V_{iR1}.

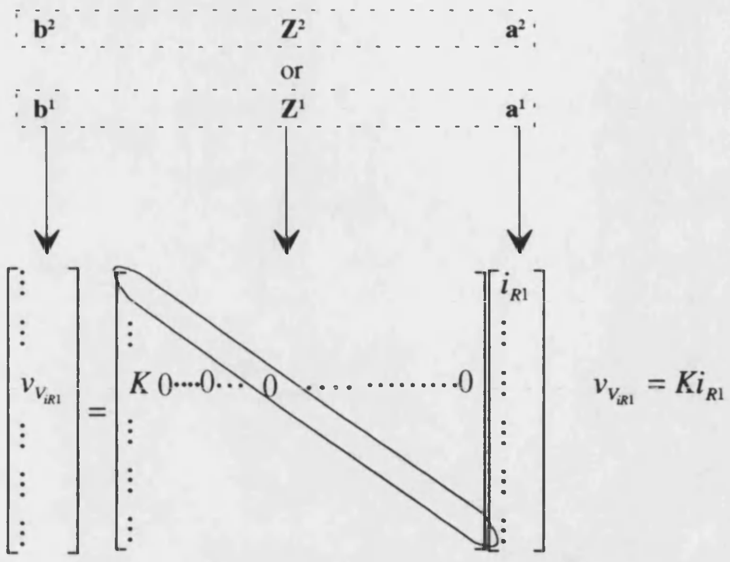


Figure 5.1-1 Construction of the component transfer matrix involving controlled source

5.2. *Necessary Condition for the Existence of the Pseudo Circuit*

The essential step in constructing a Pseudo Circuit for the CCM with respect to a component partition is the use of (5.1-9) to eliminate $\mathbf{b}^2$ in (5.1-5). It is apparent that the existence of the Pseudo Circuit relies on finding the left inverse of $\mathbf{L}^2_{21}$. However, this is only a sufficient but not necessary condition (p.129 in [75]).

The existence of a Pseudo Circuit with respect to a component partition requires the existence of the inverse of the matrix

$$\mathbf{Q}` = \begin{bmatrix} -\mathbf{I} + \mathbf{L}^{11}_{11} & \mathbf{L}^{12}_{11} \\ \mathbf{L}^{1}_{21} & \mathbf{L}^{2}_{21} \end{bmatrix} \qquad (5.2\text{-}1)$$

It can be seen that $\mathbf{Q}`$ depends only on the circuit topology and for $\mathbf{Q}`^{-1}$ to exist, the number of test points must equal the number of testee components to make $\mathbf{Q}`$ a square matrix.

This section does not attempt to prove (5.2-1) in detail as its proof is already in [75]. Nevertheless, it is the author's intention to elaborate on the unclear part in order to aid the reader to understand the subtlety of the proof. Moreover, the reasons behind the Essential Rules discussed in Chapter 4.1 can also be explained by examining (5.2-1):

1. Topology Rule

   Depending on the partition of components, the sets of identical $\mathbf{L}_{11}$ columns will result in sets of identical columns in $\mathbf{L}_{11}^{11}$, $\mathbf{L}_{11}^{12}$ or both sub-block matrices. If there are identical $\mathbf{L}_{11}^{12}$ columns, the determinant of $\mathbf{Q}`$ ($|\mathbf{Q}`|$) will be zero as all the $\mathbf{L}_{21}$ entries are initialised to zero and the Pseudo Circuit for that particular partition will not exist. Therefore, for topological connections 1) and 2), all but one of the **b** vector elements which correspond to each set of the identical $\mathbf{L}_{11}$ columns must be made test points so that $|\mathbf{Q}`|$ is not zero for all possible partitions. For topological connections 3) and 4), even a zero column in $\mathbf{L}_{11}^{12}$ implies that $\mathbf{Q}`$ is singular unless all the **b** vector elements which correspond to all zero $\mathbf{L}_{11}$ columns are made test points.

2. Diagnosability Rule

   As the $\mathbf{L}_{11}$ matrix needs to be rearranged to reflect the updated component partition at the beginning of each test cycle, if a partition results in all the non-zero entries of a $\mathbf{L}_{11}$ column in the sub-block matrix $\mathbf{L}_{11}^{22}$, there will be a zero column in the $\mathbf{Q}`$ matrix. To prevent this scenario from happening for all possible partitions, a **b** vector element must be made a test point if its corresponding $\mathbf{L}_{11}$ column has its number of non-zero entries less than or equal to the number of testee components.

The proof in [75] is based on the Pseudo-nominal Tableau[85] approach which stacks the rearranged form of all equations in the CCM apart from the component equation for the testee partition. These new equations are:

(5.1-3)a⇒

$$\mathbf{Z}^1\mathbf{a}^1 - \mathbf{b}^1 = \mathbf{0} \qquad (5.2\text{-}2)$$

(5.1-4) a and b⇒

$$\left.\begin{aligned} &a)\cdots\cdots\cdots \dot{\mathbf{x}}^1 - \mathbf{f}^1(\mathbf{x}^1,\mathbf{a}^1) = \mathbf{0} \\ &b)\cdots\cdots\cdots \mathbf{b}^1 - \mathbf{g}^1(\mathbf{x}^1,\mathbf{a}^1) = \mathbf{0} \end{aligned}\right\} \qquad (5.2\text{-}3)$$

$$(5.1\text{-}5) \Rightarrow \quad \left.\begin{aligned} &a)\cdots\cdots\cdots -\mathbf{a}^1 + \mathbf{L}_{11}^{11}\mathbf{b}^1 + \mathbf{L}_{11}^{12}\mathbf{b}^2 = -\mathbf{L}_{12}^1\mathbf{u} \\ &b)\cdots\cdots\cdots -\mathbf{a}^2 + \mathbf{L}_{11}^{21}\mathbf{b}^1 + \mathbf{L}_{11}^{22}\mathbf{b}^2 = -\mathbf{L}_{12}^2\mathbf{u} \end{aligned}\right\} \qquad (5.2\text{-}4)$$

$$(5.1\text{-}6) \Rightarrow \quad -\mathbf{y} + \mathbf{L}_{21}^1\mathbf{b}^1 + \mathbf{L}_{21}^2\mathbf{b}^2 = -\mathbf{L}_{22}\mathbf{u} \qquad (5.2\text{-}5)$$

with the tableau equations for the linear case

$$\begin{bmatrix} \mathbf{Z}^1 & -\mathbf{I} & 0 & 0 \\ -\mathbf{I} & \mathbf{L}_{11}^{11} & 0 & \mathbf{L}_{11}^{12} \\ 0 & \mathbf{L}_{11}^{21} & -\mathbf{I} & \mathbf{L}_{11}^{22} \\ 0 & \mathbf{L}_{21}^1 & 0 & \mathbf{L}_{21}^2 \end{bmatrix} \begin{bmatrix} \mathbf{a}^1 \\ \mathbf{b}^1 \\ \mathbf{a}^2 \\ \mathbf{b}^2 \end{bmatrix} = \begin{bmatrix} 0 \\ -\mathbf{L}_{12}^1\mathbf{u} \\ -\mathbf{L}_{12}^2\mathbf{u} \\ \mathbf{y} - \mathbf{L}_{22}\mathbf{u} \end{bmatrix} \qquad (5.2\text{-}6)$$

and for the non-linear case

$$\begin{bmatrix} \dot{\mathbf{x}}^1 - \mathbf{f}^1(\mathbf{x}^1,\mathbf{a}^1) \\ \mathbf{b}^1 - \mathbf{g}^1(\mathbf{x}^1,\mathbf{a}^1) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \begin{bmatrix} & & & \\ -\mathbf{I} & \mathbf{L}_{11}^{11} & 0 & \mathbf{L}_{11}^{12} \\ 0 & \mathbf{L}_{11}^{21} & -\mathbf{I} & \mathbf{L}_{11}^{22} \\ 0 & \mathbf{L}_{21}^1 & 0 & \mathbf{L}_{21}^2 \end{bmatrix} \begin{bmatrix} \mathbf{a}^1 \\ \mathbf{b}^1 \\ \mathbf{a}^2 \\ \mathbf{b}^2 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \cdots\cdots\cdots \\ \\ -\mathbf{L}_{12}^1\mathbf{u} \\ -\mathbf{L}_{12}^2\mathbf{u} \\ \mathbf{y} - \mathbf{L}_{22}\mathbf{u} \end{bmatrix} \qquad (5.2\text{-}7)$$

(5.2-6) and (5.2-7) are the compact form of the CCM. The Pseudo Circuit approach described in Chapter 5.1 (the operation to express $\mathbf{y}^P=[\mathbf{a}^2,\mathbf{b}^2]^T$ in terms of $\mathbf{u}^P=[\mathbf{u}, \mathbf{y}]^T$ and $\mathbf{b}^1$) is equivalent to solving (5.2-6) or (5.2-7). The derivation of (5.2-1) arises from considering the solvability of the tableau equations.

For the linear case, (5.2-6) is transformed by a sequence of operations:

- Swap row 2 and row 3 (row and column numbers begin from 1).
- Swap column 1 and column 2 $\Rightarrow$ Position of $\mathbf{b}^1$ and $\mathbf{a}^1$ in the column vector are swapped.
- Swap column 2 and column 3 $\Rightarrow$ Position of $\mathbf{a}^1$ and $\mathbf{a}^2$ in the column vector are swapped.

to

$$T\begin{bmatrix} b^1 \\ a^2 \\ \cdots \\ a^1 \\ b^2 \end{bmatrix} = \begin{bmatrix} -I & 0 & \vdots & Z^1 & 0 \\ L_{11}^{21} & -I & \vdots & 0 & L_{11}^{22} \\ \cdots & \cdots & \vdots & \cdots & \cdots \\ L_{11}^{11} & 0 & \vdots & -I & L_{11}^{12} \\ L_{21}^{1} & 0 & \vdots & 0 & L_{21}^{2} \end{bmatrix} \begin{bmatrix} b^1 \\ a^2 \\ a^1 \\ b^2 \end{bmatrix} = \begin{bmatrix} 0 \\ -L_{12}^{2}u \\ -L_{12}^{1}u \\ y - L_{22}u \end{bmatrix} \tag{5.2-8}$$

with

$$T = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix}$$

The purpose of the transformation is to make T_{11} an invertible matrix so that the following Lemma[75] can be applied to derive Q`:

If T_{11} is invertible, T is invertible iff $Q=T_{22}-T_{21}[T_{11}]^{-1}T_{12}$ is invertible.

After simplification,

$$Q = \begin{bmatrix} -I + L_{11}^{11}Z^1 & L_{11}^{12} \\ L_{21}^{1}Z^1 & L_{21}^{2} \end{bmatrix}$$

It is shown on p.129 in [75] that Q is invertible iff Q` (page 70) is invertible.

For the non-linear case, the solvability of (5.2-7) depends on the invertibility of its Jacobian matrix J[85]. The definition of the Jacobian matrix is as follows:

For a vectorial vector function $\mathbf{y} = \mathbf{y}(\mathbf{x}) = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_m \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix}$

J(y)=Gradient of **y** with respect to **x**.

$J_{i,k} = \partial y_i / \partial x_k$ where $J_{i,k}$ is the element in the i-th row and k-th column of

$$\mathbf{J} = \begin{bmatrix} \partial y_1/\partial x_1 & \partial y_1/\partial x_2 & \cdots\cdots & \partial y_1/\partial x_n \\ \partial y_2/\partial x_1 & \partial y_2/\partial x_2 & \cdots\cdots & \partial y_2/\partial x_n \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ \partial y_m/\partial x_1 & \partial y_m/\partial x_2 & \cdots\cdots & \partial y_m/\partial x_n \end{bmatrix}$$

This is shown with the following example:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} x_1 x_3 \\ x_2{}^2 \\ x_3{}^3/x_2 \\ x_1 + x_2 - x_3 \end{bmatrix}, \quad \mathbf{J}(\mathbf{y},\mathbf{x}) = \begin{bmatrix} x_3 & 0 & x_1 \\ 0 & 2x_2 & 0 \\ 0 & -x_3{}^3/x_2{}^2 & 3x_3{}^2/x_2 \\ 1 & 1 & -1 \end{bmatrix}$$

The n-th order discretised approximation of $\dot{\mathbf{x}}^1$, $\dot{\mathbf{x}}^1{}_{(t=t_m)} = \sum_{k=0}^{n} d_k \mathbf{x}^1{}_{(t_m-k)}$ is used to replace the first row of (5.2-7) before taking its Jacobian matrix **J** with respect to the vector $[\mathbf{x}^1, \mathbf{a}^1, \mathbf{b}^1, \mathbf{a}^2, \mathbf{b}^2]^T$.

$$\dot{\mathbf{x}}^1 - \mathbf{f}^1(\mathbf{x}^1,\mathbf{a}^1) = \mathbf{0} \Rightarrow d_0\mathbf{x}^1 - \mathbf{f}^1(\mathbf{x}^1,\mathbf{a}^1) = -\sum_{k=1}^{n} d_k \mathbf{x}^1{}_{(t_m-k)}$$

We have

$$\mathbf{J} = \begin{bmatrix} d_0\mathbf{I} - \partial\mathbf{f}^1/\partial\mathbf{x}^1 & -\partial\mathbf{f}^1/\partial\mathbf{a}^1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\partial\mathbf{g}^1/\partial\mathbf{x}^1 & -\partial\mathbf{g}^1/\partial\mathbf{a}^1 & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{I} & \mathbf{L}^{11}_{11} & \mathbf{0} & \mathbf{L}^{12}_{11} \\ \mathbf{0} & \mathbf{0} & \mathbf{L}^{21}_{11} & -\mathbf{I} & \mathbf{L}^{22}_{11} \\ \mathbf{0} & \mathbf{0} & \mathbf{L}^{1}_{21} & \mathbf{0} & \mathbf{L}^{2}_{21} \end{bmatrix} \qquad (5.2\text{-}9)$$

To derive (5.2-1), $\mathbf{J}$ is multiplied by a non-singular matrix $\hat{\mathbf{T}}$ of the same dimensions so that the following condition can be used:

$$\mathbf{J}^{-1} \text{ exists iff } (\mathbf{J}\hat{\mathbf{T}})^{-1} \text{ exists} \Rightarrow (\mathbf{J}\hat{\mathbf{T}})^{-1} \text{ exists iff exists} \left|\mathbf{J}\hat{\mathbf{T}}\right| \neq 0$$

With

$$\hat{\mathbf{T}} = \begin{bmatrix} \mathbf{I} & \mathbf{P} & 0 & 0 & 0 \\ 0 & \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I} \end{bmatrix}, \quad \mathbf{P} = (d_0\mathbf{I} - \partial\mathbf{f}^1/\partial\mathbf{x}^1)^{-1}(\partial\mathbf{f}^1/\partial\mathbf{a}^1)$$

and the assumption that $(d_0\mathbf{I} - \partial\mathbf{f}^1/\partial\mathbf{x}^1)$ is square and has full rank.

$$\mathbf{J}\hat{\mathbf{T}} = \begin{bmatrix} d_0\mathbf{I} - \partial\mathbf{f}^1/\partial\mathbf{x}^1 & 0 & 0 & 0 & 0 \\ -\partial\mathbf{g}^1/\partial\mathbf{x}^1 & [-\partial\mathbf{g}^1/\partial\mathbf{x}^1]\mathbf{P} - \partial\mathbf{g}^1/\partial\mathbf{a}^1 & \mathbf{I} & 0 & 0 \\ 0 & -\mathbf{I} & \mathbf{L}_{11}^{11} & 0 & \mathbf{L}_{11}^{12} \\ 0 & 0 & \mathbf{L}_{11}^{21} & -\mathbf{I} & \mathbf{L}_{11}^{22} \\ 0 & 0 & \mathbf{L}_{21}^{1} & 0 & \mathbf{L}_{21}^{2} \end{bmatrix}$$

$$\left|\mathbf{J}\hat{\mathbf{T}}\right| = [d_0\mathbf{I} - \partial\mathbf{f}^1/\partial\mathbf{x}^1] \cdot [\text{cofactor[86] of } d_0\mathbf{I} - \partial\mathbf{f}^1/\partial\mathbf{x}^1\ ]$$

$$\Rightarrow \mathbf{J}^{-1} \text{ exists iff } \begin{bmatrix} [-\partial\mathbf{g}^1/\partial\mathbf{x}^1]\mathbf{P} - \partial\mathbf{g}^1/\partial\mathbf{a}^1 & \mathbf{I} & 0 & 0 \\ -\mathbf{I} & \mathbf{L}_{11}^{11} & 0 & \mathbf{L}_{11}^{12} \\ 0 & \mathbf{L}_{11}^{21} & -\mathbf{I} & \mathbf{L}_{11}^{22} \\ 0 & \mathbf{L}_{21}^{1} & 0 & \mathbf{L}_{21}^{2} \end{bmatrix} \text{ is invertible.}$$

Applying the same matrix transformation for the linear case on the above reduced size cofactor matrix, the tableau matrix for the non-linear case is

$$\mathbf{T}=\begin{bmatrix} \mathbf{I} & \mathbf{0} & \vdots & [-\partial \mathbf{g}^1/\partial \mathbf{x}^1]\mathbf{P}-\partial \mathbf{g}^1/\partial \mathbf{a}^1 & \mathbf{0} \\ \mathbf{L}_{11}^{21} & -\mathbf{I} & \vdots & \mathbf{0} & \mathbf{L}_{11}^{22} \\ \cdots & \cdots & \vdots & \cdots\cdots\cdots\cdots\cdots\cdots & \cdots \\ \mathbf{L}_{11}^{11} & \mathbf{0} & \vdots & -\mathbf{I} & \mathbf{L}_{11}^{12} \\ \mathbf{L}_{21}^{1} & \mathbf{0} & \vdots & \mathbf{0} & \mathbf{L}_{21}^{2} \end{bmatrix} \qquad (5.2\text{-}10)$$

The Lemma used for the linear case is used on the sub-block matrices of **T** to obtain the counterpart of **Q**,

$$\mathbf{Q}^{N}=\begin{bmatrix} -\mathbf{I}+\mathbf{L}_{11}^{11}(-\mathbf{P}\partial \mathbf{g}^1/\partial \mathbf{x}^1-\partial \mathbf{g}^1/\partial \mathbf{a}^1) & \mathbf{L}_{11}^{12} \\ \mathbf{L}_{21}^{1}(-\mathbf{P}\partial \mathbf{g}^1/\partial \mathbf{x}^1-\partial \mathbf{g}^1/\partial \mathbf{a}^1) & \mathbf{L}_{21}^{2} \end{bmatrix}$$

With similar argument on p.129 in [75], $\mathbf{Q}^{N}$ is invertible iff $\mathbf{Q}'$ is invertible.

5.3. Partition of Circuit Components

At the beginning of each test cycle apart from the first one, circuit components are repartitioned according to the results in the previous test cycle (Figure 5-2 on page 66). From the view point of the Self Test Algorithm, the repartitioning operation involves exchanges of components between the tester and testee groups. Any testee component which is found to be good in the previous test cycle is exchanged by a tester component which is still to be proved fault free. From an algorithmic stand point, this is equivalent to the swapping of entries within the **a** and **b** vectors which correspond to the exchanged tester and testee components. The reordering of elements of the **a** and **b** vectors prompts the need to rearrange the $\mathbf{L}_{11}$ and $\mathbf{L}_{12}$ connection matrices and the test point matrix $\mathbf{L}_{21}$. As the Testing Algorithm iterates, $\mathbf{L}_{11}$, $\mathbf{L}_{12}$, $\mathbf{L}_{21}$, **a** and **b** are rearranged on a per cycle basis. These rearrangements are based only on the element entries of the matrices and vectors in the previous cycle. Assuming the tester and testee entries occupy the top and bottom positions in the **a** and **b** vectors respectively, the rearrangement operations are illustrated in Figure 5.3-1 on the next page.

When the previous test cycle does not find any good testee components, $[\mathbf{Q}']^{-1}$ does not exists (Figure 5-2) or, $\mathbf{L}_{21}^{2\;-1}$ does not exist and $\mathbf{L}_{21}^{2\;-1}$ cannot be formed by the Inter-matrix Row Exchanges, the Fault Diagnostic Program will select a new partition to continue the Testing Algorithm. In this situation there will also be exchanges within the tester and testee groups as well as between the groups. This is because the selected new partition does not always have the same element sequence as that of the partition in the previous test cycle.

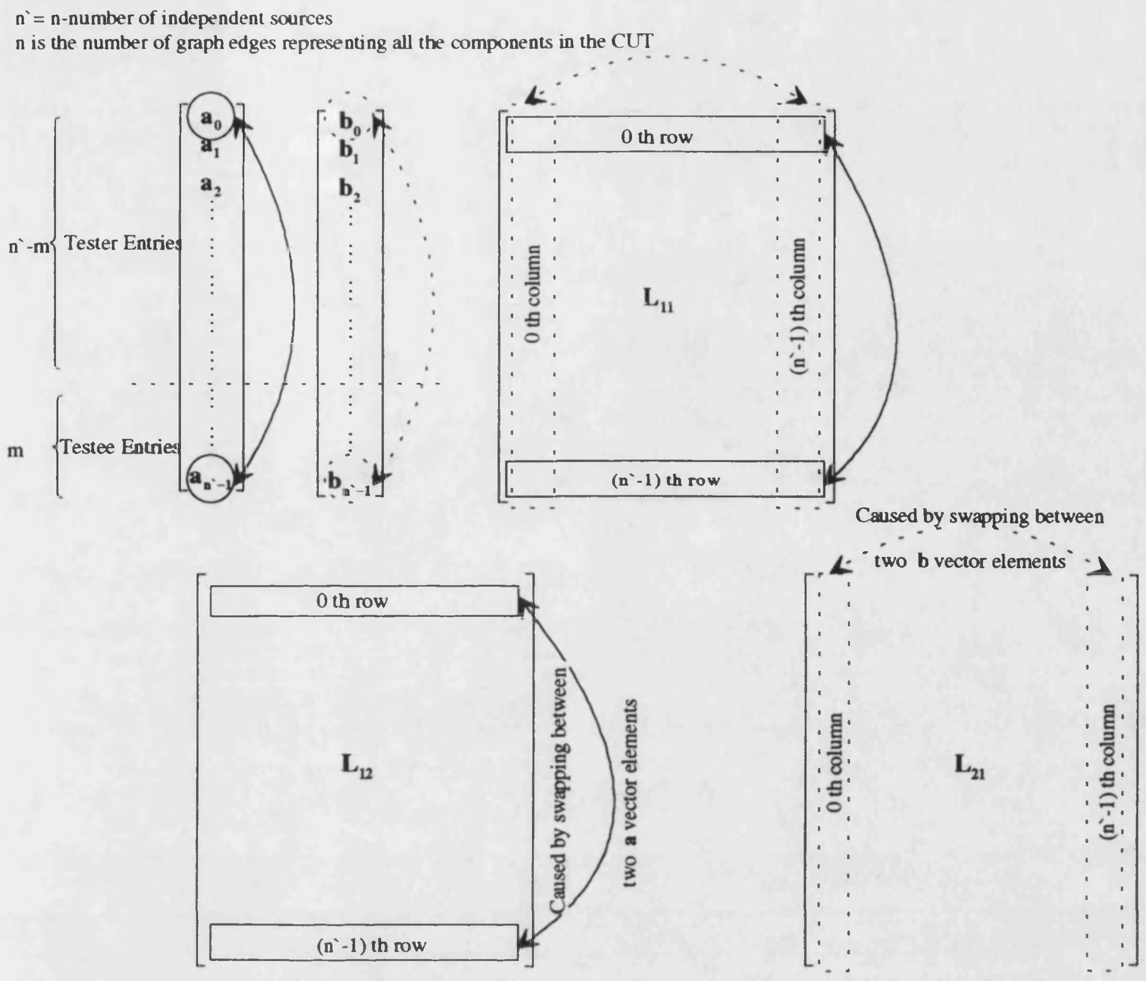
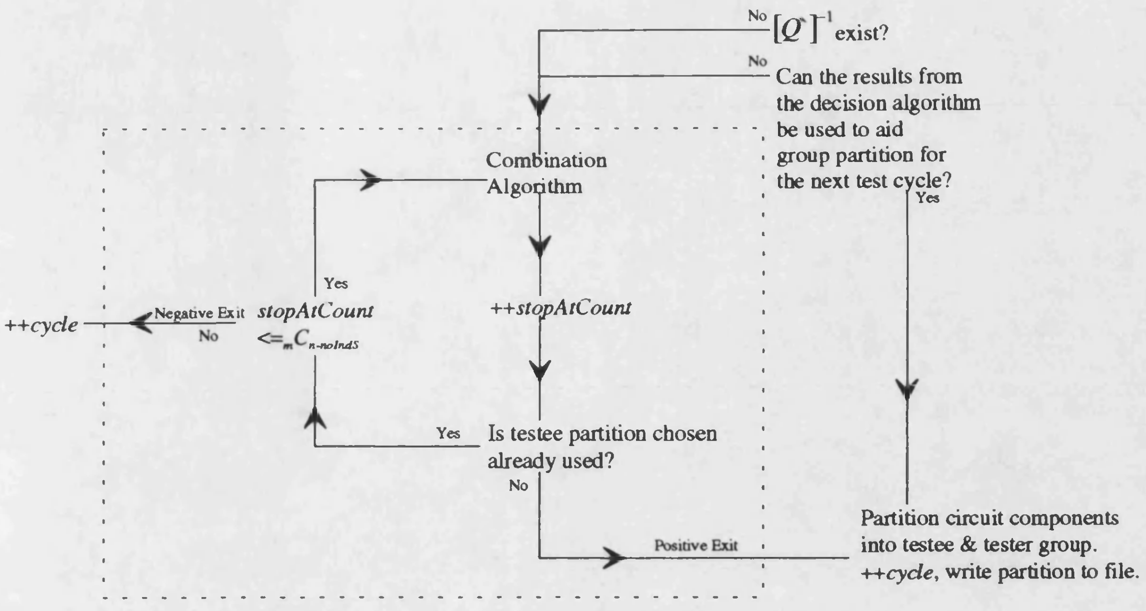

n` = n-number of independent sources
n is the number of graph edges representing all the components in the CUT
n`-m Tester Entries
m Testee Entries
a_0
a_1
a_2
$a_{n`-1}$
b_0
b_1
b_2
$b_{n`-1}$
0 th row
0 th column
L_{11}
(n`-1) th column
(n`-1) th row
Caused by swapping between two b vector elements
0 th row
L_{12}
(n`-1) th row
Caused by swapping between two a vector elements
0 th column
L_{21}
(n`-1) th column

*Figure 5.3-1 Rearrangement of the matrices and vectors L_{11}, L_{12}, L_{21}, **a** and **b** due to component partition*

Since the circuit topology is stored as an array *graph[ ]* of data structure *Edge* for the properties of each edge element which is accessed by *graph[i]* with $0 \le i \le n-1$ for a n-edge element circuit, the problem of selecting a new partition is reduced to choosing a combination of m (m is the number of testee components) numbers from the set N={0, 1, 2, ..., n-1} without repeating the combinations which correspond to the partitions used in all previous test cycles and at the same time skipping any combination in which the elements correspond to independent sources. This problem is solved by devising a Partition Generating Algorithm which is shown in Figure 5.3-2 on the following page.



$[Q]^{-1}$ exist?
No
No
Can the results from the decision algorithm be used to aid group partition for the next test cycle?
Yes
Combination Algorithm
Yes
++cycle
Negative Exit
No
stopAtCount $<={}_mC_{n-noIndS}$
++stopAtCount
Yes
Is testee partition chosen already used?
No
Positive Exit
Partition circuit components into testee & tester group. ++cycle, write partition to file.

Figure 5.3-2 Partition Generating Algorithm

This algorithm has its basis on a Combination Algorithm which chooses a collection of numbers representing the testee partition from the ${}_mC_{n\text{-}noIndS}$ combinations available from the set **N** which corresponds to the n edge elements representing all the components in the CUT (noIndS is an acronym for the number of independent sources). The Combination Algorithm is coded as a recursive procedure to pick all the possible set of m numbers from the set **N** but it only allows storage for the most recent set chosen. It utilises a variable *count* which counts the number of sets of m numbers selected. This procedure will terminate when *count* equals another variable *stopAtCount*. Its operation is explained with an example in which a testee partition of three edge elements is picked from a 5-edge element circuit. With reference to Figure 5.3-3 on the next page, we have **N**={0, 1, 2, 3, 4} and the number 3 corresponds to an independent source.

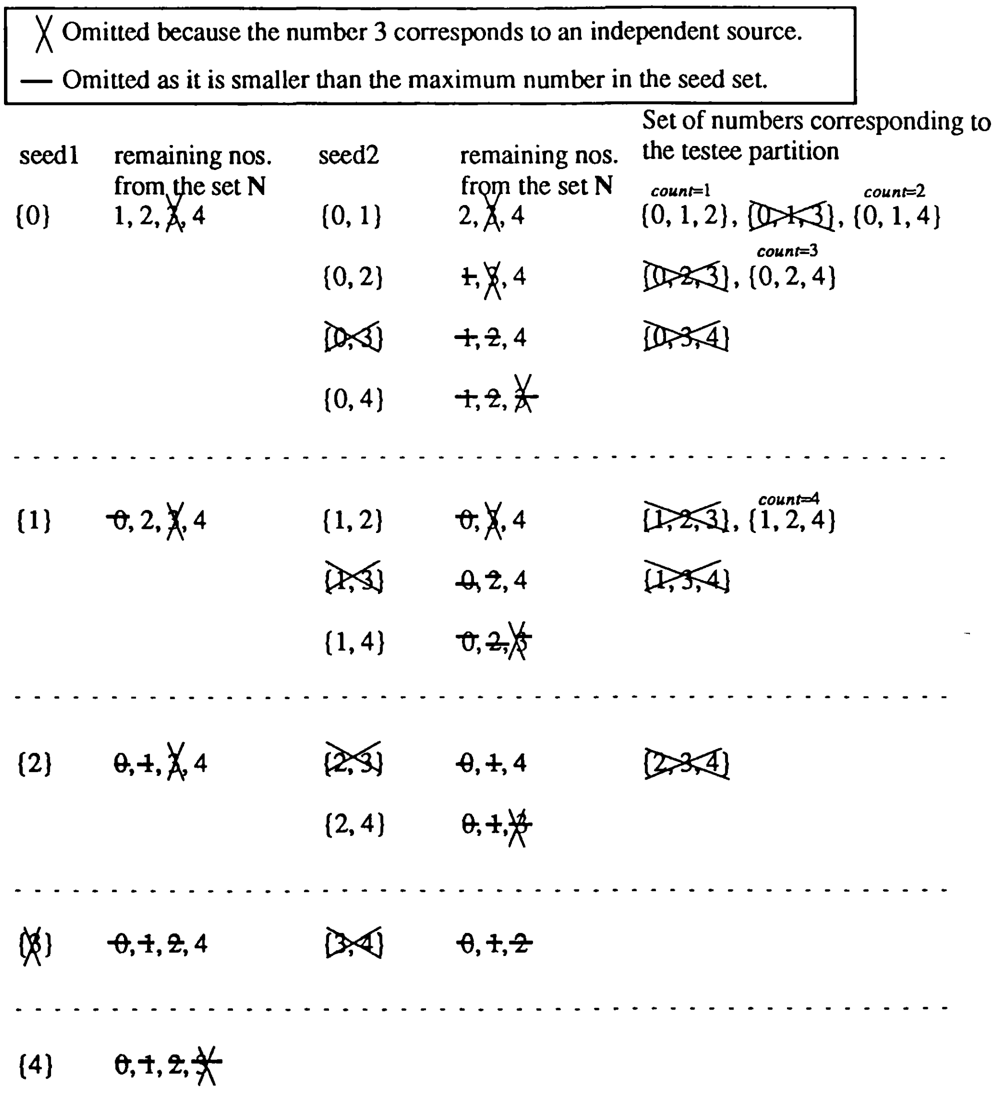

Omitted because the number 3 corresponds to an independent source.
— Omitted as it is smaller than the maximum number in the seed set.
seed1
remaining nos. from the set N
seed2
remaining nos. from the set N
Set of numbers corresponding to the testee partition
count=1
count=2
count=3
count=4
{0}
{1}
{2}
{4}
{0, 1}
{0, 2}
{0, 4}
{1, 2}
{1, 4}
{2, 4}
{0, 1, 2}
{0, 1, 4}
{0, 2, 4}
{1, 2, 4}

Figure 5.3-3 Combination Algorithm

The steps of this procedure in a non-recursive manner are as follows:

1) Each number in the set N is taken as a seed in turn. If the seed does not correspond to an independent source, each number from the remaining four numbers in N will be picked in turn to form a new seed with the old seed number provided that the number picked is larger than the seed number and it does not correspond to an independent source.

2) For each of the seed of two numbers, a number from the remaining three numbers in N is picked in turn to form the next seed combination of three numbers provided that the

number picked is larger than any of the number in the seed combination of two numbers and it does not correspond to an independent source.

3) Steps similar to 1) and 2) are carried out until the required number of numbers in a combination is met (In this case, the procedure ends at step 2. A check is also made on the variable *count* after each combination of three numbers is picked to determine whether an early exit of the procedure is necessary or not.).

5.4. Inter-matrix and Inter-vector Row Exchanges

The partition of circuit components at the beginning of each test cycle necessitates row exchanges within the matrices $\mathbf{L_{11}}$ and $\mathbf{L_{12}}$ and the vectors **a** and **b**, as well as column exchanges within the matrices $\mathbf{L_{11}}$ and $\mathbf{L_{21}}$. Parallels for these intra-matrix and intra-vector transformations are the inter-matrix and inter-vector row exchanges demanded by the construction of Pseudo Circuit when $\mathbf{L}_{21}^{2}{}^{-1}$ does not exist (Figure 5-2 on page 66). The aim of these row exchanges is to construct an invertible matrix $\mathbf{L}_{21}^{2}$.

Mathematically speaking, these inter-matrix and inter-vector row exchanges are equivalent to the swapping of equations between the matrix equation (5.1-6) and the $\mathbf{a}^1$ part of the matrix equation (5.1-5) on page 68. These swaps are broken down to the row exchanges between a pair of vectors and three pairs of matrices as shown in Figure 5.4-1.

$$\begin{tikzcd}
\mathbf{a}^1 \arrow[ddd, leftrightarrow] & = & & \mathbf{L}_{11}^{1}\mathbf{b} \arrow[dl, Rightarrow] \arrow[dr, Rightarrow] & & + \mathbf{L}_{12}^{1}\mathbf{u} \arrow[ddd, leftrightarrow] \\
& & {[\mathbf{L}_{11}^{11}} \arrow[d, leftrightarrow] & & {\mathbf{L}_{11}^{12}]} \arrow[d, leftrightarrow] & \\
& & {[\mathbf{L}_{21}^{1}} & & {\mathbf{L}_{21}^{2}]} & \\
\mathbf{y} & = & & \mathbf{L}_{21}\mathbf{b} \arrow[ul, Rightarrow] \arrow[ur, Rightarrow] & & + \mathbf{L}_{22}\mathbf{u}
\end{tikzcd}$$

Figure 5.4-1 Inter-matrix and Inter-vector Row Exchanges

An Interswap Algorithm has been devised to facilitate as far as possible the formation of an invertible matrix $\mathbf{L}_{21}^{2}$. Row exchanges involving $\mathbf{a}^1$, y and $\mathbf{L}_{22}$ are carried out on their copies $\mathbf{A}^1$, Y and $\mathbf{L}'_{22}$ respectively as their original copies are needed in the next test cycle. Since the matrices $\mathbf{L}_{11}^{1}$, $\mathbf{L}_{12}$ and $\mathbf{L}_{21}$ already contain $\mathbf{L}_{11}^{11}$, $\mathbf{L}_{11}^{12}$, $\mathbf{L}_{12}^{1}$, $\mathbf{L}_{21}^{1}$ and $\mathbf{L}_{21}^{2}$, the latter group of matrices are transformed to the matrices $\mathbf{L}'^{11}_{11}$, $\mathbf{L}'^{12}_{11}$, $\mathbf{L}'^{1}_{12}$, $\mathbf{L}'^{1}_{21}$ and $\mathbf{L}'^{2}_{21}$ respectively during the row exchanging process. A 2-dimensional binary array A₁Yswap[i][j] with $0 \le i \le n - noIndS - m - 1$ (number of testers -1), $0 \le j \le m - 1$ (number of testees -1) is used to record the row exchanges between the matrix pair $\{\mathbf{L}_{11}^{12},\ \mathbf{L}_{21}^{2}\}$. If the ith row of $\mathbf{L}_{11}^{12}$ is swapped with the jth row of $\mathbf{L}_{21}^{2}$, the entry at A₁Yswap[i][j] will be assigned the binary value *true*. The contents of A₁Yswap are initialised to *false* as there is no inter-matrix row exchange initially. When $\mathbf{L}'^{2}_{21}$ has an inverse, A₁Yswap will be used to perform the row exchanges it recorded on the remaining matrix pairs $\{\mathbf{L}_{11}^{11},\ \mathbf{L}_{21}^{1}\}$, $\{\mathbf{L}_{12}^{1},\ \mathbf{L}'_{22}\}$ and vector pair $\{\mathbf{A}^1,\ \mathbf{Y}\}$. The steps of the algorithm are as follows:

1) For $\mathbf{L}_{21}^{2}$ to become invertible, all its zero rows, and all but one of its identical, opposite and matching identical and opposite rows must be exchanged with rows from $\mathbf{L}_{11}^{12}$. Additionally, one of the identical, opposite and matching identical and opposite rows, and none of the zero rows in $\mathbf{L}_{11}^{12}$ are to be swapped with rows in $\mathbf{L}_{21}^{2}$. When a row has both identical and opposite rows, it is described as having matching identical and opposite rows to distinguish it from a row which has identical or opposite rows. To ensure these rules are observed, a two-step rearrangement on the row numbers of the matrices $\mathbf{L}_{21}^{2}$ and $\mathbf{L}_{11}^{12}$ are performed as illustrated in Figure 5.4-2 before any row swapping between the matrices.

$l = m$ (number of testees),	when	$x = 2,\ y = 2$
$l = n` - m$ (number of testers),	when	$x = 1,\ y = 12$

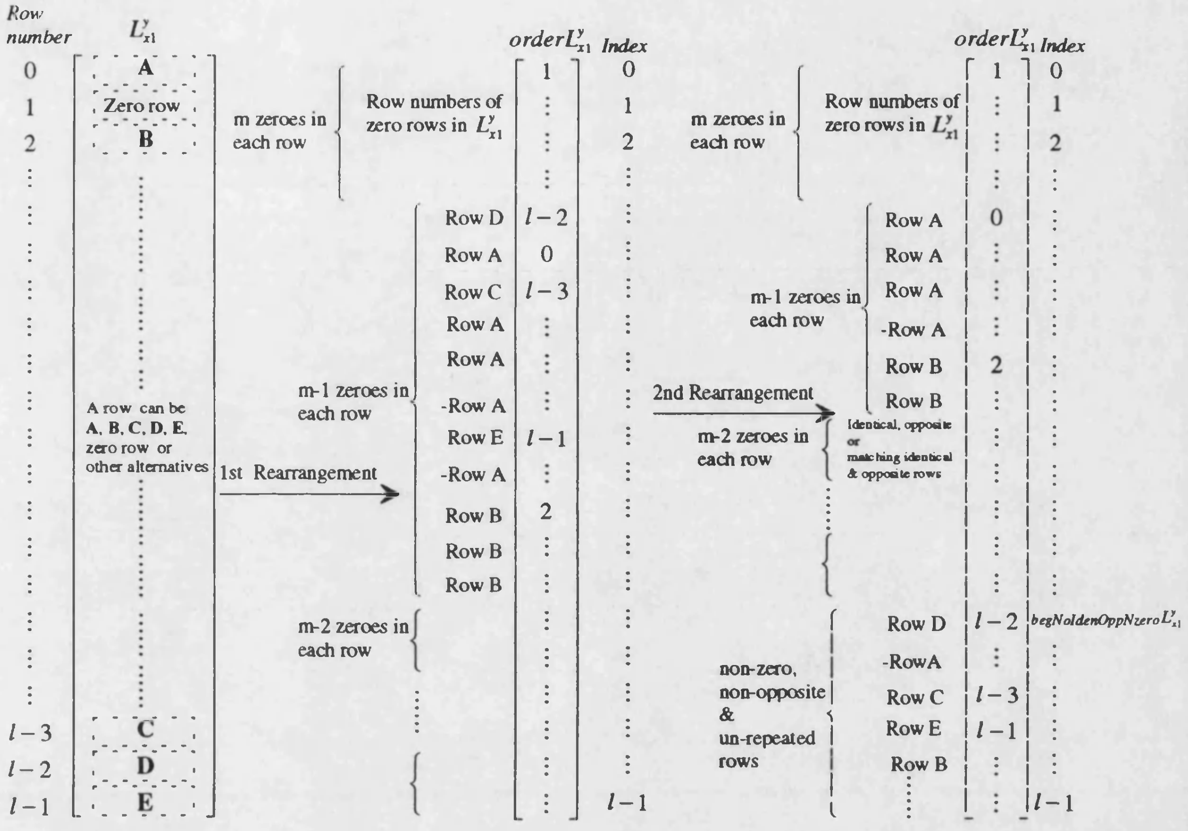


Figure 5.4-2 Two-step rearrangement of the row numbers of $\mathbf{L}^2_{21}$ *or* $\mathbf{L}^{12}_{11}$

For the convenience of illustration, the labels A, B, C, D and E are used to identify five rows of either matrices $\mathbf{L}^{12}_{11}$ or $\mathbf{L}^2_{21}$, and their positions in the matrices are assumed to be 0, 2, l-3, l-2 and l-1 for rows A, B, C, D and E respectively. The first rearrangement is to group the row numbers in descending number of zero entries per row in the array $orderL^y_{x1}$. It is assumed that the labelled rows have one non-zero entries (m-1 zeroes) in each row, row A has matching identical and opposite rows, row B has identical rows, rows C, D and E do not have any opposite or identical row. The pattern of rows in other groups apart from the zero row group are similar to that of the group with m-1 zero entries in each row.

The second rearrangement is to be performed on the row numbers in each of the groups resulted from the first rearrangement except for those row numbers with zero rows.

Within each group, the row numbers of one of the row from the identical, opposite and matching identical and opposite rows are moved together with the row numbers of all the rows without any identical or opposite row to the bottom of the array $orderL^{y}_{x1}$ to form a new group of non-zero, non-opposite and unrepeated rows, and at the same time the group is re-grouped into sub-groups which correspond to different identical, opposite or matching identical and opposite rows.

2) All zero rows in $\mathbf{L}^{2}_{21}$ are swapped with rows in $\mathbf{L}^{12}_{11}$ which have their row numbers recorded from $orderL^{12}_{11}[begNoIdenOppNzeroL^{12}_{11}]$ to $orderL^{12}_{11}[n`-m-1]$. Before each row exchange, a check for identical and opposite rows is made on the row from $\mathbf{L}^{12}_{11}$ and every $\mathbf{L}^{2}_{21}$ row between the range $orderL^{2}_{21}[begNoIdenOppNzeroL^{2}_{21}]$ and $orderL^{2}_{21}[m-1]$. If the check fails, the next available row from the group of non-zero, non-opposite and un-repeated rows in $\mathbf{L}^{12}_{11}$ will be used to repeat the check until it passes. All row exchanges are recorded in A_1Yswap.

3) All identical, opposite and matching identical and opposite rows in $\mathbf{L}^{2}_{21}$ are swapped with the remaining rows from the group of non-zero, non-opposite and un-repeated rows in $\mathbf{L}^{12}_{11}$. The check described in 2) is also carried out before each row exchange. All row exchanges are recorded in A_1Yswap.

4) If the inverse of the new $\mathbf{L}^{2}_{21}$ exists, the algorithm will stop.

5) If the inverse of the new $\mathbf{L}^{2}_{21}$ does not exist, a row from the group of non-zero, non-opposite and un-repeated rows in $\mathbf{L}^{2}_{21}$ will be picked to swap with a row from the remaining non-zero, non-opposite and un-repeated rows of $\mathbf{L}^{12}_{11}$. A check similar to that in 2) is carried out but the rows to be tested against the row from $\mathbf{L}^{12}_{11}$ are restricted to

those remaining un-swapped rows from $\mathbf{L}_{21}^{2}$. The row exchanges done are recorded in A₁Yswap.

6) Steps 4) and 5) are repeated.

However, this algorithm does not take into account the non-existence of an inverse of a matrix caused by elementary row operations. This is shown in the following example:

$$\mathbf{L}_{21}^{2} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ -1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{L}_{11}^{12} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad orderL_{21}^{2} = \begin{bmatrix} 1 \\ 0 \\ 4 \\ 2 \\ 3 \end{bmatrix}, \quad orderL_{11}^{12} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

After the first row exchange between row 1 of $\mathbf{L}_{21}^{2}$ and row 0 of $\mathbf{L}_{11}^{12}$, row 3 of $\mathbf{L}_{21}^{2}$ becomes a zero row with the addition of row 4 and the new row 1. This zero row leads to a zero determinant of $\mathbf{L}_{21}^{2}$ and hence the non-existence of its inverse. A further row exchange between row 1 of $\mathbf{L}_{11}^{12}$ and row 0 of $\mathbf{L}_{21}^{2}$ terminates the algorithm but $\mathbf{L}_{21}^{2}$ still does not have an inverse. There are no general rules apart from intuition to identify a row which will become a zero row if elementary row operations are applied on it as there can be up to r-1 such operations (which are either addition or subtraction of rows) for a r-row matrix. For the same reason, there are no rules to identify the rows which are needed for the operations to make a particular row zero. As such, the Interswap Algorithm does not always re-construct an invertible $\mathbf{L}_{21}^{2}$ matrix.

As the inter-vector swapping of elements changes $\mathbf{A}^1$ but $\mathbf{b}^1$ is left unchanged, the formation of the component transfer matrix $\mathbf{Z'}^1$ so that $\mathbf{b}^1 = \mathbf{Z'}^1\mathbf{A}^1$ is not as straight forward as the case of $\mathbf{Z}^1$ for $\mathbf{a}^1$ and $\mathbf{b}^1$. A swapped element in $\mathbf{A}^1$ can be either a $\mathbf{a}^1$, $\mathbf{a}^2$, $\mathbf{b}^1$ or $\mathbf{b}^2$ vector element because of the Test Point Selection process and there may be more than one swapped element in $\mathbf{A}^1$.

For an element $\mathbf{b}^1[i]$ ($0 \leq i < n`-m$ (number of testers)), $\mathbf{b}^1[i]$ and $\mathbf{a}^1[i]$ may also be among the swapped elements in $\mathbf{A}^1$. This fact has to be taken into account when $\mathbf{Z}`^1$ is formed. If $\mathbf{b}^1[i]$ is in $\mathbf{y}$, its respective row in $\mathbf{L}^1_{21}$ will be unique and $\mathbf{L}^2_{21}$ will be a zero row. Hence, $\mathbf{b}^1[i]$ will be among the swapped elements in $\mathbf{A}^1$.

There are many scenarios which lead to two $\mathbf{a}^1[i]$ in $\mathbf{A}^1$. An obvious case is when the respective rows of $\mathbf{a}^1[i]$ in $\mathbf{L}^2_{21}$ and $\mathbf{L}^{12}_{11}$ are zero rows, the zero row in $\mathbf{L}^{12}_{11}$ will not be swapped whereas the other in $\mathbf{L}^2_{21}$ will be swapped. One of the $\mathbf{a}^1[i]$ is the unswapped element corresponding to $\mathbf{b}^1[i]$ ($\mathbf{A}^1[i]=\mathbf{a}^1[i]$) and the other is the swapped element corresponding to $\mathbf{b}^1[j]$ ($i \neq j,\ \ 0 \leq j < n`-m,\ \ \mathbf{A}^1[j] = \mathbf{a}^1[i] \neq \mathbf{a}^1[j]$). Since $\mathbf{a}^1[i]$ will be a compacted test point if it is in $\mathbf{y}$, its respective rows in $\mathbf{L}^2_{21}$ and $\mathbf{L}^{12}_{11}$ are identical.

When the respective row of $\mathbf{a}^1[i]$ in $\mathbf{L}^{12}_{11}$ is considered for swapping, it will only be left unswapped (hence $\mathbf{A}^1[i]=\mathbf{a}^1[i]$) if the respective row of $\mathbf{a}^1[i]$ in $\mathbf{L}^2_{21}$ and any $\mathbf{L}^2_{21}$ row that is identical or opposite to this row are still unswapped. This case will lead to two $\mathbf{a}^1[i]$ in $\mathbf{A}^1$ if the row number of the respective row of $\mathbf{a}^1[i]$ in $\mathbf{L}^2_{21}$ is one of the groups of identical, opposite or matching identical and opposite rows in $orderL^2_{21}$. However, if the row number is in the group of non-zero, non-opposite and unrepeated rows, this case will lead to either two $\mathbf{a}^1[i]$ in $\mathbf{A}^1$ ($\mathbf{A}^1[i]=\mathbf{a}^1[i]$, $\mathbf{A}^1[j]=\mathbf{a}^1[i]$) or one unswapped $\mathbf{a}^1[i]$ in $\mathbf{A}^1$ ($\mathbf{A}^1[i]=\mathbf{a}^1[i]$).

When the respective row of $\mathbf{a}^1[i]$ in $\mathbf{L}^2_{21}$ and any $\mathbf{L}^2_{21}$ row that is identical or opposite to this row are swapped before the respective row of $\mathbf{a}^1[i]$ in $\mathbf{L}^{12}_{11}$ is considered for swapping, the respective row of $\mathbf{a}^1[i]$ in $\mathbf{L}^{12}_{11}$ will be swapped. This case leads to one swapped $\mathbf{a}^1[i]$ in $\mathbf{A}^1$ ($\mathbf{A}^1[i] \neq \mathbf{a}^1[i]$, $\mathbf{A}^1[j]=\mathbf{a}^1[i]$).

To take into consideration the above scenarios, the different cases for the possible values of each row i of $\mathbf{Z'}^1$, $\mathbf{Z'}^1[i][j]$, with $0 \le i,\ \ j,\ \ q < n'-m$, are:

1) $\mathbf{A}^1[i]$=$\mathbf{a}^1[i]$ and $\mathbf{b}^1[i]$ is the current or voltage of a passive circuit component.

$\mathbf{Z'}^1[i][i]$=component admittance if $\mathbf{b}^1[i]$ is a current quantity

$\mathbf{Z'}^1[i][i]$=component impedance if $\mathbf{b}^1[i]$ is a voltage quantity

$\mathbf{Z'}^1[i][j \ne i] = 0$

2) $\mathbf{A}^1[i]$=$\mathbf{a}^1[i]$ and $\mathbf{b}^1[i]$ is the current or voltage of a controlled source.

$\mathbf{Z'}^1[i][i]$=0 and

if the controlling element of the controlled source is in $\mathbf{A}^1$ ($q \ne i,\ \ \mathbf{b}^1[i] = K\mathbf{A}^1[q]$) and $\mathbf{A}^1[q] \ne \mathbf{A}^1[p]$ $(p \ne i,\ \ p < q,\ \ 0 \le p < n'-m)$

$\mathbf{Z'}^1[i][q] = K$
$\mathbf{Z'}^1[i][j \ne q \ \ and \ \ j \ne i] = 0$

else if the controlling element of the controlled source is not in $\mathbf{A}^1$ and $\mathbf{A}^1[q]$=$\mathbf{b}^1[i]$ $(q \ne i)$

$\mathbf{Z'}^1[i][q] = 1$
$\mathbf{Z'}^1[i][j \ne q \ \ and \ \ j \ne i] = 0$

else if the controlling element of the controlled source is not in $\mathbf{A}^1$ and $\mathbf{A}^1[j] \ne \mathbf{b}^1[i]$ $(j \ne i)$

$\mathbf{Z'}^1[i][j \ne i] = 0$

3) $\mathbf{A}^1[i] \neq \mathbf{a}^1[i]$ and $\mathbf{b}^1[i]$ is the current or voltage of a passive circuit component.

if $\mathbf{A}^1[q \neq i]=\mathbf{a}^1[i]$

$\mathbf{Z}'^1[i][q] =$ component admittance if $\mathbf{b}^1[i]$ is a current quantity

$\mathbf{Z}'^1[i][q] =$ component impedance if $\mathbf{b}^1[i]$ is a voltage quantity

$\mathbf{Z}'^1[i][j \neq q] = 0$

else if $\mathbf{A}^1[q \neq i] \neq \mathbf{a}^1[i]$ and $\mathbf{A}^1[q]=\mathbf{b}^1[i]$

$\mathbf{Z}'^1[i][q] = 1$

$\mathbf{Z}'^1[i][j \neq q] = 0$

else if $\mathbf{A}^1[j \neq i] \neq \mathbf{a}^1[i]$ and $\mathbf{A}^1[j] \neq \mathbf{b}^1[i]$

$\mathbf{Z}'^1[i][j] = 0$

4) $\mathbf{A}^1[i] \neq \mathbf{a}^1[i]$ and $\mathbf{b}^1[i]$ is the current or voltage of a controlled source.

if the controlling element of the controlled source is in $\mathbf{A}^1$ $(\mathbf{b}^1[i] = K\mathbf{A}^1[q])$ and $\mathbf{A}^1[q] \neq \mathbf{A}^1[p]$ $(p < q,\ 0 \leq p < n' - m)$

$\mathbf{Z}'^1[i][q] = K$
$\mathbf{Z}'^1[i][j \neq q] = 0$

else if the controlling element of the controlled source is not in $\mathbf{A}^1$ and $\mathbf{A}^1[q]=\mathbf{b}^1[i]$

$\mathbf{Z}'^1[i][q] = 1$
$\mathbf{Z}'^1[i][j \neq q] = 0$

else if the controlling element of the controlled source is not in $\mathbf{A}^1$ and $\mathbf{A}^1[j] \neq \mathbf{b}^1[i]$

$\mathbf{Z}'^1[i][j] = 0$

5.5. Analytical Expression for the Pseudo Circuit Output

It is straight forward to derive an analytical expression for the output $\mathbf{y}^{\mathrm{P}}$ of the Pseudo Circuit in the linear case by solving $\mathbf{b}^1$ with (5.1-3)a and $\mathbf{a}^1$ part of (5.1-7).

$$\mathbf{a}^1 = \mathbf{K}_{11}\mathbf{Z}^1\mathbf{a}^1 + \mathbf{K}_{12}\mathbf{u}^{\mathrm{P}}, \qquad \mathbf{u}^{\mathrm{P}} = \begin{bmatrix} \mathbf{u} \\ \mathbf{y} \end{bmatrix}$$

$$\mathbf{a}^1 = [\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1]^{-1}\mathbf{K}_{12}\mathbf{u}^{\mathrm{P}}$$

$$\Rightarrow \mathbf{b}^1 = \mathbf{Z}^1[\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1]^{-1}\mathbf{K}_{12}\mathbf{u}^{\mathrm{P}} \tag{5.5-1}$$

$$\Rightarrow \mathbf{y}^{\mathrm{P}} = \{\mathbf{K}_{21}\mathbf{Z}^1[\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1]^{-1}\mathbf{K}_{12} + \mathbf{K}_{22}\}\mathbf{u}^{\mathrm{P}} = \begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \\ \mathbf{M}_{21} & \mathbf{M}_{22} \end{bmatrix}\mathbf{u}^{\mathrm{P}} \tag{5.5-2}$$

With

$$\left.\begin{aligned}
\mathbf{M}_{11} &= [\mathbf{L}^{21}_{11} - \mathbf{L}^{22}_{11}\mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}^{1}_{21}]\mathbf{Z}^1[\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1]^{-1}[\mathbf{L}^{1}_{12} - \mathbf{L}^{12}_{11}\mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}_{22}] \\
&\quad + \mathbf{L}^{2}_{12} - \mathbf{L}^{22}_{11}\mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}_{22} \\
\mathbf{M}_{12} &= [\mathbf{L}^{21}_{11} - \mathbf{L}^{22}_{11}\mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}^{1}_{21}]\mathbf{Z}^1[\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1]^{-1}\mathbf{L}^{12}_{11}\mathbf{L}^{2}_{21}{}^{-1} + \mathbf{L}^{22}_{11}\mathbf{L}^{2}_{21}{}^{-1} \\
\mathbf{M}_{21} &= -\mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}^{1}_{21}\mathbf{Z}^1[\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1]^{-1}[\mathbf{L}^{1}_{12} - \mathbf{L}^{12}_{11}\mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}_{22}] - \mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}_{22} \\
\mathbf{M}_{22} &= -\mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}^{1}_{21}\mathbf{Z}^1[\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1]^{-1}\mathbf{L}^{12}_{11}\mathbf{L}^{2}_{21}{}^{-1} + \mathbf{L}^{2}_{21}{}^{-1}
\end{aligned}\right\} \tag{5.5-3}$$

As $\mathbf{Z}^1$ is a function of frequency, a frequency is a testable frequency so long as the inverse of $\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1$ exists. The existence of $[\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1]^{-1}$ is checked with the Crout's Algorithm[87] which performs a LU decomposition on a rowise permutation of $\mathbf{I} - \mathbf{K}_{11}\mathbf{Z}^1$. If inter-matrix and inter-vector row exchanges are needed for a particular component partition, $\mathbf{a}^1$, $\mathbf{Z}^1$ and $\mathbf{y}$ in the above equations will be replaced by $\mathbf{A}^1$, $\mathbf{Z}'^1$ and $\mathbf{Y}$ respectively, and the constants $\mathbf{K}_{11}$, $\mathbf{K}_{12}$, $\mathbf{K}_{21}$ and $\mathbf{K}_{22}$ will be evaluated using the changed sub-block connection matrices as described in the previous section.

5.6. Decision Algorithms

After solving for the Pseudo circuit Output $\mathbf{y}^P = \begin{bmatrix} \mathbf{a}^2 \\ \mathbf{b}^2 \end{bmatrix}$, the next step in a test cycle is to compare $\mathbf{b}^2$ with $\hat{\mathbf{b}}^2 = \mathbf{Z}^2\mathbf{a}^2$ (Figure 5-2 on page 66) to yield a digital result vector with the use of suitable fault models. The test results of a testee edge element corresponding to the ith element of $\mathbf{b}^2$ is a fail if $\left|\mathbf{b}_i^2 - \hat{\mathbf{b}}_i^2\right| > \tau$, it is a pass if $\left|\mathbf{b}_i^2 - \hat{\mathbf{b}}_i^2\right| \leq \tau$, with τ determined by the fault models. However, if appropriate fault models are not available, a normalised blanket decision threshold obtained from experience(see Appendix IV) will be used instead (Version 1 of the diagnosis program has been coded using this approach. As an alternative to using a global decision threshold, the diagnosis program also has an option to use normalised local decision thresholds which are specific to different edge elements on the circuit graph.). When all the testees are tested in the aforementioned manner, the test results will be expressed in a digital vector format with a '1' and '0' to identify the failed and passed testees respectively. However, a failed testee may not be actually faulty and may be tested failed because of the presence of faulty testers. On the other hand, a passed testee edge element may not be fault-free and may be tested passed because of faulty testers. With the application of the Exact and Heuristic Decision Algorithms [76] to the test results for the cases of single and multiple faults respectively, any testee which is tested passed is fault-free and can be used in the tester group in the next test cycle. These algorithms are described as follows:

1. Exact Algorithm

   This algorithm assumes a single fault case and deals with the three possible cases of test results.

   a) All testees are tested passed. (Test results of all testees are all passes)

If a passed testee was actually faulty, there would be at least a faulty tester which changed the test result of the testee from '1' to '0'. This contradicts the single fault assumption and thus any testee which is tested passed must be actually fault-free. In this case all the testees are good and can be used as testers in all succeeding test cycles.

b) More than one testees are tested failed.

If the faulty edge element was one of the failed testees, the other failed testees would be actually good and their test results would be changed by faulty testers from '0' to '1'. This again contradicts the single fault assumption and thus the faulty edge element must be in the tester group. In this case all the testees are good and can be used as testers in all succeeding test cycles.

c) One testee is tested failed.

The failed testee can be either faulty or fault-free. The latter case is caused by a faulty tester. In this case all but the failed testee are good and can be used as testers in all succeeding test cycles.

2. Heuristic Algorithm

This algorithm is for the multiple-fault case and makes use of an analogue heuristic that two independent analogue failures will never cancel. When this heuristic is imposed on a testee which is tested passed, it implies that the testee is actually good. If a passed testee was actually faulty, there would be a faulty tester whose effect is to mask the faulty testee so that it appears as a pass in the test results. The presence of such a faulty tester is not allowed by the heuristic and as such, any testee which is tested passed must be actually fault-free.

The treatments of the algorithm of the test results are the same as those of the Exact Algorithm except for the treatment of case b) of the test results. In case b) only the passed testees are good and can be used as testers in all succeeding test cycles as no information is implied from the failed testees.

Additionally, this algorithm can be improved at the expense of carrying out an off line simulation to generate a coupling table for the component partition used in a test cycle. A coupling table indicates whether or not a faulty tester component will yield erroneous test results on the testees. With reference to Figure 5.6-1,

Results	Testees\Testers	h_0	h_1	h_2	h_3	h_4
0	k_0	1	0	1	0	1
1	k_1	1	1	0	0	0
0	k_2	0	1	1	0	0

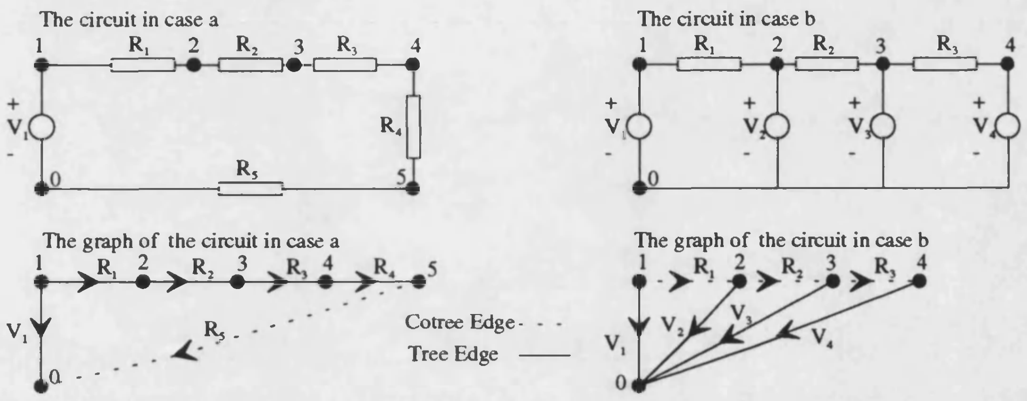
Figure 5.6-1 Coupling table for the Heuristic Algorithm

If the tester h_0 is faulty, the test results on k_0 and k_1 will be erroneous . The testees k_0 and k_1 are said to be coupled to the tester h_0. The entries for the rest of the columns in the coupling table are interpreted in the same way as the column entries for h_0.

With the aid of such coupling table, if the test results are as shown in Figure 5.6-1, the testers h_0, h_1, h_2 and h_4 will be good as the testees k_0 and k_2 are tested passed. Either k_0 or k_2 used together with h_0, h_1, h_2 and h_4 as testers in the next test cycle will yield reliable test results if the next test cycle is testable.

5.7. Non-diagnosable Circuit Topologies

No algorithm can diagnose faults on all possible circuit topologies. The Non-hierarchical FDA proposed in this report is not an exception to this fact. The non-hierarchical circuits shown in Figure 5.7-1 are just two of the examples of known topologies which are not diagnosable by the FDA.



The connection equation of the circuit in case a

$$\underset{\mathbf{a}}{\begin{bmatrix} i_{R1} \\ i_{R2} \\ i_{R3} \\ i_{R4} \\ v_{R5} \end{bmatrix}} = \underset{\mathbf{L_{11}}}{\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & -1 & -1 & 0 \end{bmatrix}} \underset{\mathbf{b}}{\begin{bmatrix} v_{R1} \\ v_{R2} \\ v_{R3} \\ v_{R4} \\ i_{R5} \end{bmatrix}} + \underset{\mathbf{L_{12}}}{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}} \underset{\mathbf{u}}{v_1}$$

The connection equation of the circuit in case b

$$\underset{\mathbf{a}}{\begin{bmatrix} v_{R2} \\ v_{R1} \\ v_{R3} \end{bmatrix}} = \underset{\mathbf{L_{11}}}{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}} \underset{\mathbf{b}}{\begin{bmatrix} i_{R2} \\ i_{R1} \\ i_{R3} \end{bmatrix}} + \underset{\mathbf{L_{12}}}{\begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}} \underset{\mathbf{u}}{\begin{bmatrix} v_2 \\ v_3 \\ v_1 \\ v_4 \end{bmatrix}}$$

Figure 5.7-1 Examples of Non-diagnosable Circuits

In case a, v_{R1}, v_{R2}, v_{R3} and v_{R4} are both topology- and diagnosability-related test points for all possible values of Diagnosis Depth and hence they cannot be compacted using v_{R5}. Subsequent re-application of the Diagnosability Rule with the test points v_{R1}, v_{R2}, v_{R3} and v_{R4} will also select i_{R5} as a diagnosability-related test point. This is not allowed as this will leave no component in the tester group. In case b, the $\mathbf{L_{11}}$ connection matrix is a null matrix which implies that all the **b** vector elements must be test points and compaction is impossible. This will again leave no component in the tester group and is not allowed.

5.8. Conclusions

This section has built on the topics discussed in the previous sections to propose a non-hierarchical Fault Diagnosis Algorithm (FDA). The understanding of this FDA requires the discussion of various related issues encompassing the Self Test (ST) Algorithm[75][76] with the Component Connection Model (CCM)[77] formulation, the necessary condition for the existence of the Pseudo Circuit[75], the partition of circuit components, inter-matrix and inter-vector row exchanges, and decision algorithms. In addition, two examples of circuit topologies non-diagnosable with the FDA have been given and an analytical expression for the output of the Pseudo Circuit in terms of its input, the connection and test point matrices, and the tester partition of the circuit component transfer matrix has been derived.

The essence of the ST Algorithm is the partition of circuit components into tester and testee groups with the assumptions that all the testers are fault free. This assumption will be true and the results of the last test cycle will be reliable so that the test results can be used to see if any testees are faulty when the ST Algorithm converges after iterations of test cycles consisting of component partition and test as described in Chapter 5.1, which has also explained the CCM of a CUT and the partitioned form of the CCM equations due to component partition at the beginning of each test cycle. These partitioned forms of the CCM equations excluding the partitioned component equations, after some manipulations, become the so called Pseudo Circuit equations which relate the tester partition of the input variable vector ($\mathbf{a}^1$) and their output (testee partition of the input and output variable vectors, $\mathbf{a}^2$ and $\mathbf{b}^2$) with the tester partition of the output variable vector ($\mathbf{b}^1$) and their input (stimulus and test point vectors, $\mathbf{u}$ and $\mathbf{y}$). Additionally, the tester and testee partitions of the circuit component transfer matrix are diagonal for a non-hierarchical circuit.

The necessary condition for being able to construct the Pseudo Circuit equations is the existence of the inverse of the matrix $\mathbf{Q}$` (Chapter 5.2) which depends only on circuit topology for a non-hierarchical circuit. Its proof originates from considering the solvability of the

Tableau equations[85] which stack a re-arranged form of the partitioned CCM equations apart from the testee partition of the component equation. These Tableau equations are a compact form of the partitioned CCM equations and therefore are equivalent to the Pseudo Circuit equations. Chapter 5.2 has elaborated on the unclear part of the detailed proof for this necessary condition in [75] and also explained the reasons behind the Essential Rules discussed in Chapter 4.1 using this necessary condition.

As circuit components are partitioned at the beginning of every test cycle, this implies that the connection ($\mathbf{L}_{11}$, $\mathbf{L}_{12}$) and test point ($\mathbf{L}_{21}$, $\mathbf{L}_{22}$) matrices, the input (**a**) and output (**b**) variable vectors are re-arranged on a per test cycle basis. Component partition necessitates the swapping of elements between the vectors $\mathbf{a}^1$ and $\mathbf{a}^2$, and also between the vectors $\mathbf{b}^1$ and $\mathbf{b}^2$. These intra-vector swapping of elements in the input variable vector in turn prompt for intra-matrix swapping of the rows, corresponding to the swapping elements in the vector **a**, in the matrices $\mathbf{L}_{11}$ and $\mathbf{L}_{12}$, whereas the equivalent operations on the output variable vector result in intra-matrix swapping of the columns, corresponding to the swapping elements in the vector **b**, in the matrices $\mathbf{L}_{11}$ and $\mathbf{L}_{21}$. Moreover, the convergence of the ST Algorithm requires the FDA to generate an arbitrary component partition when one of the situations described in Chapter 5.3 occurs. A partition generating algorithm has also been described in Chapter 5.3 to satisfy this requirement. In this case, the intra-vector swapping of elements will also occur within the tester and testee partitions, as well as between the tester and testee partitions of the vectors **a** and **b**. This is because the arbitrarily generated component partition does not always have the same element sequence as that of the partition in the previous test cycle.

When the inverse of $\mathbf{Q}`$ exists but the inverse of $\mathbf{L}^2_{21}$ does not exist, the construction of the Pseudo Circuit equations will require a new invertible $\mathbf{L}^2_{21}$ matrix to be built by inter-matrix and inter-vector row exchanges. These operations have been described in detail in Chapter 5.4. In essence, these operations are equivalent to the swapping of equations between the measurement equation (5.1-6) and the $\mathbf{a}^1$ part of the matrix equation (5.1-5).

Decision algorithms are needed to identify fault free testees at the end of a test cycle so that these fault free testees can be moved to the tester partition in the beginning of the next test cycle. The Exact and Heuristic Algorithms are for single and multiple fault diagnosis respectively and they have been explained in detail in Chapter 5.6.

6. Hierarchical Fault Diagnosis

The Basic Fault Diagnosis Algorithm proposed in the previous section is theoretically sound but it is impractical for diagnosing faults in large analogue circuits. A rudimentary version of this algorithm had been reported in [88] and [84] to diagnose faults in two small analogue circuits. The improvements of the proposed Basic FDA over the reported rudimentary algorithm are the use of the Optimal Tree Generation (Chapter 3.2) and Test Point Selection (Chapter 4) Procedures. In addition, a component partition can be used for testing even though its associated $\mathbf{L}_{21}^{2}{}^{-1}$ does not exist and there is no need to generate a table of component partitions, whose corresponding $\mathbf{L}_{21}^{2}$ matrices have inverses, prior to the actual testing. The impracticality of the FDA and its rudimentary counterpart has its root in the underlying matrix analysis as the sizes of the matrices required, and hence the computing resources, are proportional to the size of a CUT. Performing the matrix analysis at the discrete component level of circuit description with either algorithm will impose maximum drain on computing resources and limit the fault diagnosis to circuits which can be handled by finite computer primary memory.

At the expense of diagnosability, efficient use of computer resources can be achieved by hiding parts of the CUT into several "black-box" components. The use of these black-box components effectively decreases the size of the CUT as seen by the FDA and hence the reduction on the requirement for computing resources. What is lost is the diagnosability within these black-box components. In other words, the FDA can only perform go/no-go testing[89] on these black-box components. To take this approach further, each black-box component can also consist of black-box components which embed smaller black-box components. For example, practical circuits such as filter and integrator are the black box components at the system level. One level below this highest level are complex circuits like opamp and comparator. The next level downwards are simple circuits such as differential input stage, current mirror, level shifter and inverter. This is equivalent to having hierarchies of circuit blocks within the CUT of which the circuit blocks in the lowest hierarchy are discrete components at the device level of circuit description. Fault diagnosis can then be performed from a higher hierarchy to a lower hierarchy if the faults inside a black-box component have to be identified. For this reason, the black-box component is termed the "hierarchical component". In the following sections, the necessary enhancements and modifications on the Basic FDA for adopting the hierarchical approach together with the resulting Hierarchical FDA are discussed.

6.1. Hierarchical Approach

As the hierarchical component envisaged in the previous paragraph is an analogue circuit block at any hierarchical level with the aim of reducing the complexity of the CUT, it has, by its very nature, multiple terminals. To include such a multiple terminal component into the formulation of the FDA, the initial task must be to investigate all the possible graph representations of the hierarchical component since an edge on a circuit has inherently two nodes.

The first approach is to use a node to represent a hierarchical or non-hierarchical component, and an edge to represent a single connection between two components. This is not feasible as the voltage information of a component cannot be conveyed.

In [79] and [80], two separate graphs are used to represent a circuit containing active components such as transistors and opamps. These graphs are called voltage and current graphs as their edges represent either voltage or current quantities respectively. They are identical for a circuit with purely passive components but they are different for a circuit with active components. This two graph approach requires the use of a multi-port description for the hierarchical component.

In the example below,

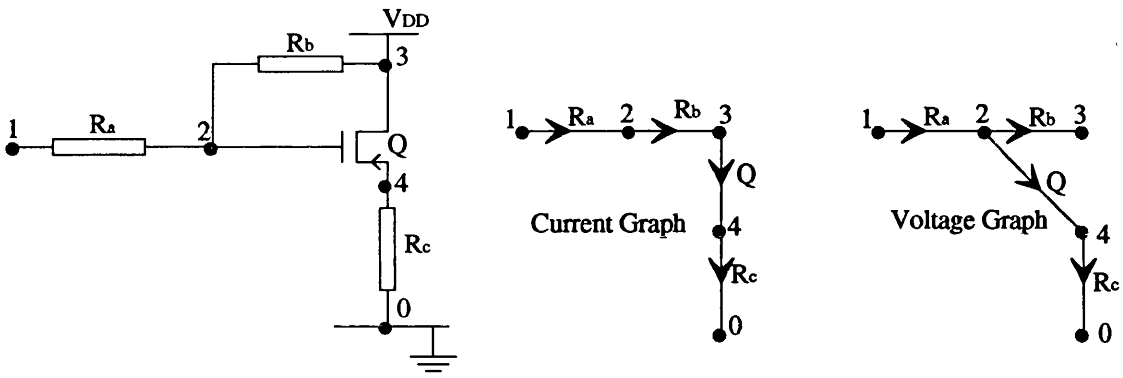


Figure 6-1 Example for the current/voltage graph approach

a MOST is treated as a 2-port VCCS. As the variables of interest are the input voltage (v_{gs}) and the output current (i_{drain}), the current graph has an edge corresponding to i_{drain} and the voltage graph has an edge corresponding to v_{gs}.

Another example is a BJT, which is treated as a 2-port CCCS. Although the variables of interest are the base and collector currents, the base-emitter and collector-emitter voltages must also be represented on the voltage graph to make the number of edges on both current and voltage graphs the same. Otherwise, the connection matrix $\mathbf{L}_{11}$ will not always be a square matrix which is the assumption behind the CCM. To generalise the current/voltage graph representation of the hierarchical component, as many current and voltage edges as necessary can be used to represent the current and voltage variables of the hierarchical component. This generalisation relies on the assumption that a multi-port description of the hierarchical component exists. Such a description is possible and will be explained in the next paragraph discussing the graph representation of the hierarchical component finally adopted. What stops the use of the current/voltage graph approach is the fact that the connection equation (2.2-4) will no longer be symmetrical if this approach is adopted. This means that the edges in $\mathbf{i}_t$ and $\mathbf{v}_{ct}$ (vector **a**) do not necessarily correspond to those in $\mathbf{v}_t$ and $\mathbf{i}_{ct}$ (vector **b**) respectively as the voltage and current graphs are different. If the two graph approach had been adopted, the matrix equation connecting $\mathbf{i}_t$, $\mathbf{i}_{ct}$ and $\mathbf{i}_{ts}$ would have been derived by applying the OTGP on the current graph of the CUT, whereas the matrix equation connecting $\mathbf{v}_{ct}$, $\mathbf{v}_t$ and $\mathbf{v}_{vs}$ would have been obtained by applying an Optimal Cotree Generation Procedure(OCGP) on the voltage graph. This OCGP will be based on KVL to construct a loop matrix which is then subjected to operations similar to those in the OTGP to pick an optimal cotree. If **B** is the loop matrix of the voltage graph, KVL states that

$$[\mathbf{B}_T \quad \mathbf{B}_{CT}]\begin{bmatrix}\mathbf{v}_T\\ \mathbf{v}_{CT}\end{bmatrix} = \mathbf{0} \Rightarrow \mathbf{B}_T\mathbf{v}_T + \mathbf{B}_{CT}\mathbf{v}_{CT} = \mathbf{0} \Rightarrow \mathbf{B}_{CT}^{-1}\mathbf{B}_T\mathbf{v}_T + \mathbf{v}_{CT} = \mathbf{0} \Rightarrow \mathbf{v}_{CT} = \mathbf{D}_B\mathbf{v}_T$$

$$\Rightarrow \mathbf{D}_B = -\mathbf{B}_{CT}^{-1}\mathbf{B}_T$$

where $\mathbf{D}_B$ is the fundamental matrix for the voltage graph.

Once $\mathbf{D}_B$ is derived, it is partitioned in a way similar to that of **D** to extract the relevant block matrices to form the other half of the connection equation.

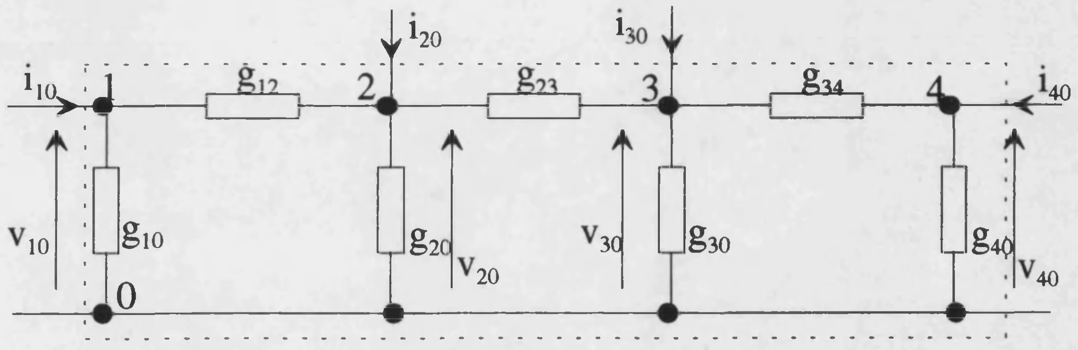
The only way to maintain the symmetry of the connection equation is to use the one graph approach which carries both current and voltage information on an edge. For a hierarchical component with n terminals, n edges are used to represent the current and voltage at the terminals with respect to a common node which can be anywhere in the CUT. If this common node is on the hierarchical component itself, it will have n-1 instead of n constituent edges. The common node is always the out-node of all the constituent edges whereas the in-node of each of the constituent edges is the individual terminal on the hierarchical component. This star connection structure is typical for the graph representation of the hierarchical component.

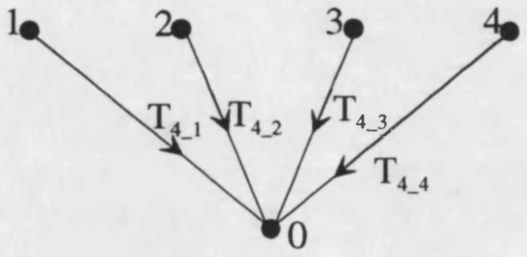
To avoid making the mathematical analysis over-complicated, the hierarchical components investigated in this research work are limited to those whose constituent edge currents and voltages are related in a linear manner. In other words, the edge currents and voltages of the hierarchical component are described by an equation similar to (5.1-1):

$$\mathbf{b}_{\text{hier}} = \mathbf{z}\mathbf{a}_{\text{hier}} \qquad (6.1\text{-}1)$$

The hierarchical component transfer matrix **z**, is part of the circuit component transfer matrix **Z** in (5.1-1), whereas the edge currents and voltages are elements of the column vectors $\mathbf{b}_{\text{hier}}$

and $\mathbf{a}_{hier}$ which are parts of the $\mathbf{b}$ and $\mathbf{a}$ vectors respectively. $\mathbf{z}$ depends on the test frequency in general and its derivation is component specific and out of the scope of this research work. Details on the methodology for modelling of the hierarchical component are presented in [90]. Below are two examples of the hierarchical components and their associated hierarchical component transfer matrices:





$$\mathbf{b_{T4}} = \mathbf{Z_{T4}a_{T4}}$$

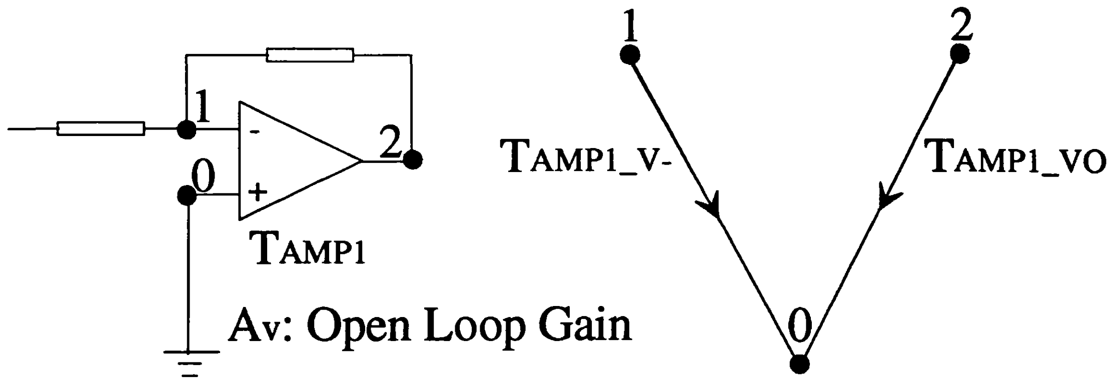
g_{ij}: admittance between node i, j

$$\begin{bmatrix} i_{10} \\ i_{20} \\ i_{30} \\ i_{40} \end{bmatrix} = \begin{bmatrix} g_{10}+g_{12} & -g_{12} & 0 & 0 \\ -g_{12} & g_{12}+g_{20}+g_{23} & -g_{23} & 0 \\ 0 & -g_{23} & g_{23}+g_{30}+g_{34} & -g_{34} \\ 0 & 0 & -g_{34} & g_{34}+g_{40} \end{bmatrix} \begin{bmatrix} v_{10} \\ v_{20} \\ v_{30} \\ v_{40} \end{bmatrix}$$

Figure 6.1-1 A passive network and its representation as a hierarchical component

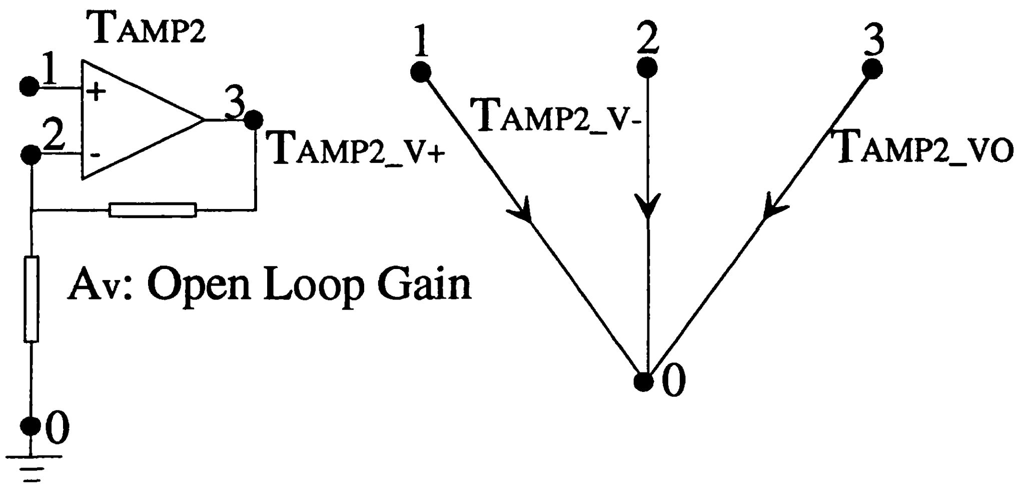
In Figure 6.1-1, each of the constituent admittance of the passive network can be capacitive, inductive or resistive. The network is abstracted as a hierarchical component T_4 with five terminals. As the common node is on the network itself and is also the reference node, T_4 has four instead of five star-connected edges of which their voltages are node voltages. When the common node is not the reference node, the edge voltages are branch voltages and are mapped to the node voltages, which are the actual physical quantities measured or simulated, with equations (3.2-5) and (3.2-6). The component transfer matrix of T_4 is just the network admittance matrix and its construction is demonstrated in [86].

The next example is a single-ended opamp in inverting and non-inverting amplifier configurations in Figures 6.1-2 and 6.1-3.



$$\begin{bmatrix} iT_{AMP1\_v-} \\ vT_{AMP1\_vo} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ -A_v & 0 \end{bmatrix} \begin{bmatrix} vT_{AMP1\_v-} \\ iT_{AMP1\_vo} \end{bmatrix}$$

Figure 6.1-2 Simplest hierarchical representation of an opamp in inverting amplifier configuration



$$\begin{bmatrix} i_{TAMP2\_v+} \\ i_{TAMP2\_v-} \\ v_{TAMP2\_vo} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ A_v & -A_v & 0 \end{bmatrix} \begin{bmatrix} v_{TAMP2\_v+} \\ v_{TAMP2\_v-} \\ i_{TAMP2\_vo} \end{bmatrix}$$

Figure 6.1-3 Simplest hierarchical representation of an opamp in non-inverting amplifier configuration

The opamp in Figures 6.1-2 and 6.1-3 is assumed to be near the ideal case except that it has a finite open loop gain A_v. In fact, for a CMOS opamp operating in the low frequency region, the input admittance at its v^+ and v^- nodes are approaching the ideal case, zero input admittance.

Therefore, it is true for the matrix entries $z_{\text{inverting}}[0][0]$, $z_{\text{non-inverting}}[0][0]$ and $z_{\text{non-inverting}}[1][1]$ to be zero. Non-zero values at these matrix entries represent admittance which models the input leakage currents of a real opamp or an ideal CMOS opamp with input leakage current fault. If these hierarchical component transfer matrices have to be more realistic, parameters such as non-zero output impedance (R_{out}) and CMRR of an opamp will need to be represented in the matrices. To model the non-zero output impedance, the matrix entries $z_{\text{inverting}}[1][1]$ and $z_{\text{non-inverting}}[2][2]$ are set to R_{out}. The common mode effect is due to the slight mismatch of the open loop gain at the v^+ and v^- input nodes so that the amplification of the common mode input voltage (v_{cm}) is not subtracted out. This effect can be neglected in the inverting amplifier case as the common mode input voltage is zero because of the negative feedback and the grounded v^+ node. For the non-inverting amplifier case, the feedback causes the voltage at v^- to follow the input voltage at v^+ and hence the common mode input voltage varies directly with the input voltage. This common mode gain can be referred to the opamp input as an equivalent common mode input error voltage (v_{ecm})[91], with $v_{ecm}= v_{cm}/\text{CMRR}$, so that the common mode effect is modelled by an additional term $A_v/(2*\text{CMRR})$ in the matrix entries $z_{\text{non-inverting}}[2][0]$ and $z_{\text{non-inverting}}[2][1]$.

The hierarchical representations of a differential-ended opamp in inverting and non-inverting amplifier configurations can be obtained with an argument similar to the one discussed above for the single-ended cases, except that one more output edge is needed for the inverting output which makes the required component transfer matrices have dimensions of 3x3 and 4x4 respectively for inverting and non-inverting amplifier configurations.

In the next section, the general condition for the existence of a tree/cotree edges partition on the constituent edges of a hierarchical component is discussed and elaborated to all possible tree/cotree edge partitions.

6.2. Tree/ Cotree Edge Partitions on a Hierarchical Component

In the previous section, a hierarchical component is characterised by specific pairs of voltage and current which are abstracted as star-connected edges. When the circuit block of a CUT is treated as a hierarchical component, the constituent edge currents and voltages of the hierarchical component must be part of the vectors **a** and **b** which are partitioned into tree and cotree edge parts (Chapter 2.2). Thus, whether one of its constituent edges is a tree or cotree edge is obvious from the type of physical quantities (current or voltage) represented by each element in its corresponding $\mathbf{b}_{\mathbf{hier}}$ or $\mathbf{a}_{\mathbf{hier}}$ vector. A rule on partitioning the constituent edges of a hierarchical component into tree and cotree edges results from examining its associated vector $\mathbf{b}_{\mathbf{hier}}$: if an element of the $\mathbf{b}_{\mathbf{hier}}$ vector is a current or voltage quantity, its corresponding edge will be on the cotree or tree respectively. The first tree/cotree edge partition arrived at with this rule and the hierarchical component transfer matrix corresponding to the vectors $\mathbf{b}_{\mathbf{hier}}$ and $\mathbf{a}_{\mathbf{hier}}$ are called the root partition and root partition matrix of the hierarchical component respectively.

Circuit topology may dictate that a tree cannot be formed with the root partitions of some hierarchical components in the CUT. In this circumstance, alternative tree/cotree edge partitions have to be used for these hierarchical components. These alternatives are derived from the root partition matrices: with each root partition, an alternative partition is obtained by swapping any numbers of elements from the vector $\mathbf{b}_{\mathbf{hier}}$ with the corresponding elements from the vector $\mathbf{a}_{\mathbf{hier}}$. This alternative partition is only possible if the root partition matrix can be manipulated to a new matrix relating the edge currents and voltages in the new tree/cotree edge partition. Generally speaking, the existence of an alternative partition depends on the existence of the inverse of the block matrix corresponding to the elements swapped to arrive at the alternative partition from the root partition. This is illustrated in Table 6.2-1 which shows all the possible alternative tree/cotree edge partitions of the 5 terminal hierarchical component T_4 in Figure 6.1-1.

Elements of vector b_{T4} to be swapped with its respective elements in vector a_{T4}	Tree Edge	Condition
i_{10}, i_{20}, i_{30}, i_{40}	T_{4_1}, T_{4_2}, T_{4_3}, T_{4_4}	Z_{T4}^{-1} exists
i_{10}, i_{20}, i_{40}	T_{4_1}, T_{4_2}, T_{4_4}	$\begin{bmatrix} g_{10}+g_{12} & -g_{12} & 0 \\ -g_{12} & g_{12}+g_{20}+g_{23} & 0 \\ 0 & 0 & g_{34}+g_{40} \end{bmatrix}^{-1}$ exists
i_{10}, i_{30}, i_{40}	T_{4_1}, T_{4_3}, T_{4_4}	$\begin{bmatrix} g_{10}+g_{12} & 0 & 0 \\ 0 & g_{23}+g_{30}+g_{34} & -g_{34} \\ 0 & -g_{34} & g_{34}+g_{40} \end{bmatrix}^{-1}$ exists
i_{10}, i_{20}, i_{30}	T_{4_1}, T_{4_2}, T_{4_3}	$\begin{bmatrix} g_{10}+g_{12} & -g_{12} & 0 \\ -g_{12} & g_{12}+g_{20}+g_{23} & -g_{23} \\ 0 & -g_{23} & g_{23}+g_{30}+g_{34} \end{bmatrix}^{-1}$ exists
i_{20}, i_{30}, i_{40}	T_{4_2}, T_{4_3}, T_{4_4}	$\begin{bmatrix} g_{12}+g_{20}+g_{23} & -g_{23} & 0 \\ -g_{23} & g_{23}+g_{30}+g_{34} & -g_{34} \\ 0 & -g_{34} & g_{34}+g_{40} \end{bmatrix}^{-1}$ exists
i_{10}, i_{40}	T_{4_1}, T_{4_4}	$\begin{bmatrix} g_{10}+g_{12} & 0 \\ 0 & g_{34}+g_{40} \end{bmatrix}^{-1}$ exists
i_{10}, i_{20}	T_{4_1}, T_{4_2},	$\begin{bmatrix} g_{10}+g_{12} & -g_{12} \\ -g_{12} & g_{12}+g_{20}+g_{23} \end{bmatrix}^{-1}$ exists
i_{10}, i_{30}	T_{4_1}, T_{4_3}	$\begin{bmatrix} g_{10}+g_{12} & 0 \\ 0 & g_{23}+g_{30}+g_{34} \end{bmatrix}^{-1}$ exists
i_{20}, i_{40}	T_{4_2}, T_{4_4}	$\begin{bmatrix} g_{12}+g_{20}+g_{23} & 0 \\ 0 & g_{34}+g_{40} \end{bmatrix}^{-1}$ exists
i_{30}, i_{40}	T_{4_3}, T_{4_4}	$\begin{bmatrix} g_{23}+g_{30}+g_{34} & -g_{34} \\ -g_{34} & g_{34}+g_{40} \end{bmatrix}^{-1}$ exists
i_{20}, i_{30}	T_{4_2}, T_{4_3}	$\begin{bmatrix} g_{12}+g_{20}+g_{23} & -g_{23} \\ -g_{23} & g_{23}+g_{30}+g_{34} \end{bmatrix}^{-1}$ exists
i_{10}	T_{4_1}	$g_{10}+g_{12} \neq 0$
i_{40}	T_{4_4}	$g_{34}+g_{40} \neq 0$
i_{20}	T_{4_2}	$g_{12}+g_{20}+g_{23} \neq 0$
i_{30}	T_{4_3}	$g_{23}+g_{30}+g_{34} \neq 0$

Table 6.2-1 All possible alternative tree/cotree edge partitions of the 5 terminal hierarchical component T_4

For the hierarchical representations of the opamp in Figures 6.1-2 and 6.1-3, there is no alternative tree/cotree edge partition because of the sparseness of the root partition matrices.

In order to facilitate the implementation of the automatic generation of alternative tree/cotree edge partitions for hierarchical components, the condition for the existence of an alternative partition is formulated as follows:

For a n-edge hierarchical component described by

$$\begin{bmatrix} b_{hier\_0} \\ b_{hier\_1} \\ \vdots \\ b_{hier\_n-1} \end{bmatrix} = \mathbf{z} \begin{bmatrix} a_{hier\_0} \\ a_{hier\_1} \\ \vdots \\ a_{hier\_n-1} \end{bmatrix}, \quad \mathbf{z} = \begin{bmatrix} z_{00} & z_{01} & \cdots & z_{0,n-1} \\ z_{10} & z_{11} & \cdots & z_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ z_{n-1,0} & z_{n-1,1} & \cdots & z_{n-1,n-1} \end{bmatrix},$$

if a different partition is to be used, the root partition matrix, **z**, will need to be altered according to the new definition of tree and cotree edges in the different partition. A new hierarchical component transfer matrix, $\mathbf{z}'$, is associated with each possible partition:

- General case: number of swap elements is more than 1 and less than n

  The matrix **z** has to be transformed to $\mathbf{z}''$, with $\begin{bmatrix} \mathbf{B}_{hier\_0} \\ \mathbf{B}_{hier\_1} \end{bmatrix} = \begin{bmatrix} \mathbf{z}''_{00} & \mathbf{z}''_{01} \\ \mathbf{z}''_{10} & \mathbf{z}''_{11} \end{bmatrix} \begin{bmatrix} \mathbf{A}_{hier\_0} \\ \mathbf{A}_{hier\_1} \end{bmatrix}$, such that all the elements of the vectors $\mathbf{b}_{hier}$ and $\mathbf{a}_{hier}$ to be swapped are in $\mathbf{B}_{hier\_0}$ and $\mathbf{A}_{hier\_0}$ respectively. After the swapping,

  $$\begin{bmatrix} \mathbf{A}_{hier\_0} \\ \mathbf{B}_{hier\_1} \end{bmatrix} = \mathbf{z}''' \begin{bmatrix} \mathbf{B}_{hier\_0} \\ \mathbf{A}_{hier\_1} \end{bmatrix}, \mathbf{z}''' = \begin{bmatrix} {\mathbf{z}''_{00}}^{-1} & -{\mathbf{z}''_{00}}^{-1}\mathbf{z}''_{01} \\ \mathbf{z}''_{10}{\mathbf{z}''_{00}}^{-1} & \mathbf{z}''_{11} - \mathbf{z}''_{10}{\mathbf{z}''_{00}}^{-1}\mathbf{z}''_{01} \end{bmatrix},$$

  $\mathbf{z}'$ is obtained by a reverse transformation on $\mathbf{z}'''$.

- Special case (transformation on **z** is unnecessary):
  - Number of swap element is one,

b_{hier_r} is swapped with a_{hier_r} so that $\begin{bmatrix} b_{hier\_0} \\ b_{hier\_1} \\ \vdots \\ b_{hier\_r-1} \\ a_{hier\_r} \\ b_{hier\_r+1} \\ \vdots \\ b_{hier\_n-1} \end{bmatrix} = z^{`} \begin{bmatrix} a_{hier\_0} \\ a_{hier\_1} \\ \vdots \\ a_{hier\_r-1} \\ b_{hier\_r} \\ a_{hier\_r+1} \\ \vdots \\ a_{hier\_n-1} \end{bmatrix}$

$z^{`}_{xy} = {}^{-z_{xy}}\!/_{z_{rr}}$ if x=r and y≠r, $z^{`}_{xy} = {}^{1}\!/_{z_{rr}}$ if x=r=y,

$z^{`}_{xy} = z_{xy} - {}^{z_{xr}z_{ry}}\!/_{z_{rr}}$ if x≠r and y≠r, $z^{`}_{xy} = {}^{z_{xr}}\!/_{z_{rr}}$ if x≠r and y=r

x and y are the row and column indices of $z^{`}$ and z respectively.

- All the vector elements are swapped, $z^{`}=z^{-1}$

From the discussion in the preceding paragraphs, a set of tree/cotree edge partitions for a hierarchical component can be defined from its root partition matrix and b_{hier} vector. This fact provides the possibilities of having more than one circuit testing tree for a CUT with hierarchical components, which will inevitably enhance the usability and efficiency of the FDA. On the usability issue, the chance of having incomplete diagnosis (failure in test point selection or having a set of ambiguous components which can be faulty or fault free) on the CUT is minimised because of the availability of multiple circuit testing trees. In the case of incomplete diagnosis, as some components have already been diagnosed fault free, the ST Algorithm can be begun with all these fault free components in the tester group to speed up the convergence of the algorithm.

To include this provision of multiple testing trees in the FDA, a priority scheme must be devised to pick combinations of tree/cotree edge partitions of all the hierarchical components in a CUT. The discussion of this scheme is postponed until all the other enhancements and modifications are explored in the forthcoming sections.

6.3. Hierarchical Pre Selection

Once a combination of tree/cotree edge partitions of all the hierarchical components in a CUT is chosen, their constituent edges are effectively divided into two groups, namely, hierarchical tree edges and hierarchical cotree edges. These two new component groups are a direct consequence of adopting the hierarchical approach which calls for a new priority grouping for circuit components prior to the construction of the node incidence matrix (Chapter 3.2). This is shown in Table 6.3-1 below:

Group Number	Circuit Component
0	Independent Voltage Sources connected to the reference node
1	Independent Voltage Sources not connected to the reference node
2	Hierarchical Tree Edges
3	Capacitors
4	Resistors and Conductors
5	Inductors
6	Hierarchical Cotree Edges
7	Independent Current Sources

Table 6.3-1 Grouping of circuit components for hierarchical preselection

As a controlled source and its controlling element are conceptually a hierarchical component with 4 terminals and they are not real physical devices, the old controlled voltage and current sources groups in Table 3.2-2 are absorbed into the new hierarchical tree and cotree edges groups. This simplification removes the need to have dummy nodes (Chapter 3.2) in the input netlist file (see Appendix II) of a CUT with current controlled sources. There are still two reasons to treat controlled sources and their controlling element as hierarchical components: the component equation **b=Za** specifies that the controlling element of a current controlled or voltage controlled source must be on the tree or cotree respectively as the controlling element appears as one of the entries in the vector **a** which is partitioned as $[\mathbf{i}_{tree} \quad \mathbf{v}_{cotree}]^T$. This rule can be violated if the controlling element is a passive component, with the use of Ohm's Law. If a controlling element is a constituent edge of a hierarchical component, the relationship between the voltage and current of the controlling element will not be as simple as in the case of a passive component and this rule cannot always be violated. The effect of having one of the constituent edges of a hierarchical component be the controlling element of a controlled source is a reduction of the possible tree/cotree partitions for the hierarchical component. Another reason for keeping a controlled source and its controlling element as a hierarchical component is to simplify the coding of the FDA.

To avoid the ambiguity which arises during the allocation of column indices of the node incidence matrix to equal weight graph edges within the same component group (Chapter 7.1.2), component values are used as a second criterion in addition to using edge weight as the first criterion: edges corresponding to resistors and inductors are arranged in ascending component values whereas edges corresponding to capacitors and conductors are ordered in descending component values. Hierarchical edges are arranged in ascending values of their respective diagonal entries in **z`** if these entries signify the impedance of edges. Otherwise, hierarchical edges are arranged in descending values of their respective diagonal entries in **z`** if these entries signify the admittance of edges.

6.4. Test Point Selection Revisited

The Test Point Selection rules will be the same as those discussed in Chapter 4 except for the interpretation of the minimum number of test points, 1+t, for a t-diagnosable circuit with hierarchical components[75]. The sum 1+t originally refers to the minimum number of testee components required for t-diagnosability[83] regardless of the graph representation of the components. This should still be true at the hierarchical level. As the mathematical analysis behind the FDA is performed at the graph edge level, the required minimum number of testee edges becomes a function (Fn(1+t)) of the required minimum testee components 1+t. In [75], the number of test points must equal the number of testee edges for the construction of the testability matrix **Q**. Hence, 1+t is also a measure on the minimum number of test points to satisfy a given diagnosis depth (t) requirement for a non-hierarchical CUT. In the case of a hierarchical CUT, the required minimum number of test points is simply Fn(1+t). The flow charts in Figures 4-1, 4.3-2 and 4.3-3 for Test Point Selection can also be applied to a hierarchical circuit if all the occurrences of 1+t are replaced by Fn(1+t). For example, with 1-diagnosability, the testee group must have at least 2 components and at least 2 test points are needed for the non-hierarchical CUT. For the hierarchical CUT, if a hierarchical component is in the testee group, all of its constituent edges must be in the testee group as the hierarchical component is a single entity and its constituent edges are only an abstraction of its characteristic voltage/current pairs. The required minimum number of test points will be at a maximum if the testee partition contains the largest and second largest hierarchical components. Therefore, the required minimum number of test points to account for all possible combinations of testee partitions will be the sum of the constituent edges of the two hierarchical components which have the maximum number of constituent edges among all the hierarchical components. If there is only one hierarchical component whose number of constituent graph edges is j_1, the required minimum number of test points will be $1+j_1$. If there is one more hierarchical component in the circuit whose number of constituent graph edges is j_2, the required minimum number of test points will be j_1+j_2 instead.

The fact that a hierarchical component is a single entity suggests that all of its constituent edges must be kept together during component partition. The mathematical justification behind this hierarchical partition rule is discussed in the next section.

6.5. *Hierarchical Rules*

As the component partition in the ST Algorithm causes a partitioning of the circuit component transfer matrix **Z** into the form $\begin{bmatrix} \mathbf{Z}^1 & \mathbf{0} \\ \mathbf{0} & \mathbf{Z}^2 \end{bmatrix}$ which must be maintained for the algorithm to work, all the constituent edges of each hierarchical component must be kept within the tester or testee group in each test cycle.

If this partition rule is not imposed, the component equations (5.1-3) and (5.1-4) will have alternative forms when the constituent edges of at least one hierarchical component are in different partitions. The proof for (5.2-1) (Chapter 5.1) will have to be extended to take into consideration these alternative forms of the component equations and will result in a different expression for $\mathbf{Q}$` if the mathematics works out.

The component equation in the linear case is used as an example:

$$\begin{bmatrix}\mathbf{b}^1\\ \mathbf{b}^2\end{bmatrix}=\begin{bmatrix}\mathbf{Z}^1 & \mathbf{Z}^{12}\\ \mathbf{Z}^{21} & \mathbf{Z}^2\end{bmatrix}\begin{bmatrix}\mathbf{a}^1\\ \mathbf{a}^2\end{bmatrix}$$

Alternative form of (5.2-2) $\Rightarrow$ $\mathbf{Z}^1\mathbf{a}^1+\mathbf{Z}^{12}\mathbf{a}^2-\mathbf{b}^1=\mathbf{0}$

Alternative form of (5.2-6) $\Rightarrow \begin{bmatrix}\mathbf{Z}^1 & -\mathbf{I} & \mathbf{Z}^{12} & \mathbf{0}\\ -\mathbf{I} & \mathbf{L}^{11}_{11} & \mathbf{0} & \mathbf{L}^{12}_{11}\\ \mathbf{0} & \mathbf{L}^{21}_{11} & -\mathbf{I} & \mathbf{L}^{22}_{11}\\ \mathbf{0} & \mathbf{L}^{1}_{21} & \mathbf{0} & \mathbf{L}^{2}_{21}\end{bmatrix}\begin{bmatrix}\mathbf{a}^1\\ \mathbf{b}^1\\ \mathbf{a}^2\\ \mathbf{b}^2\end{bmatrix}=\begin{bmatrix}\mathbf{0}\\ -\mathbf{L}^1_{12}\mathbf{u}\\ -\mathbf{L}^2_{12}\mathbf{u}\\ \mathbf{y}-\mathbf{L}_{22}\mathbf{u}\end{bmatrix}$

Alternative form of (5.2-8) $\Rightarrow \mathbf{T}\begin{bmatrix}\mathbf{b}^1\\ \mathbf{a}^2\\ \mathbf{a}^1\\ \mathbf{b}^2\end{bmatrix}=\left[\begin{array}{cc:cc}-\mathbf{I} & \mathbf{Z}^{12} & \mathbf{Z}^1 & \mathbf{0}\\ \mathbf{L}^{21}_{11} & -\mathbf{I} & \mathbf{0} & \mathbf{L}^{22}_{11}\\ \cdots & \cdots & \cdots & \cdots\\ \mathbf{L}^{11}_{11} & \mathbf{0} & -\mathbf{I} & \mathbf{L}^{12}_{11}\\ \mathbf{L}^{1}_{21} & \mathbf{0} & \mathbf{0} & \mathbf{L}^{2}_{21}\end{array}\right]\begin{bmatrix}\mathbf{b}^1\\ \mathbf{a}^2\\ \mathbf{a}^1\\ \mathbf{b}^2\end{bmatrix}=\begin{bmatrix}\mathbf{0}\\ -\mathbf{L}^2_{12}\mathbf{u}\\ -\mathbf{L}^1_{12}\mathbf{u}\\ \mathbf{y}-\mathbf{L}_{22}\mathbf{u}\end{bmatrix}$

With $\mathbf{T}=\begin{bmatrix}\mathbf{T}_{11} & \mathbf{T}_{12}\\ \mathbf{T}_{21} & \mathbf{T}_{22}\end{bmatrix}$

The following Lemma[75] is used to derive the alternative expression for $\mathbf{Q}$:

If $\mathbf{T}_{22}$ is invertible, $\mathbf{T}$ is invertible iff $\mathbf{Q}=\mathbf{T}_{11}-\mathbf{T}_{12}\mathbf{T}_{22}{}^{-1}\mathbf{T}_{21}$ is invertible.

$$\mathbf{T}_{22}{}^{-1}=\begin{bmatrix}-\mathbf{I} & \mathbf{L}^{12}_{11}\mathbf{L}^{2}_{21}{}^{-1}\\ \mathbf{0} & \mathbf{L}^{2}_{21}{}^{-1}\end{bmatrix},\mathbf{Q}=\begin{bmatrix}-\mathbf{I} & \mathbf{Z}^{12}\\ \mathbf{L}^{21}_{11} & -\mathbf{I}\end{bmatrix}-\begin{bmatrix}\mathbf{Z}^1 & \mathbf{0}\\ \mathbf{0} & \mathbf{L}^{22}_{11}\end{bmatrix}\mathbf{T}_{22}{}^{-1}\begin{bmatrix}\mathbf{L}^{11}_{11} & \mathbf{0}\\ \mathbf{L}^{1}_{21} & \mathbf{0}\end{bmatrix}$$

$$\mathbf{Q}=\begin{bmatrix}-\mathbf{I}-\mathbf{Z}^1[\mathbf{L}^{12}_{11}\mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}^{1}_{21}-\mathbf{L}^{11}_{11}] & \mathbf{Z}^{12}\\ \mathbf{L}^{21}_{11}-\mathbf{L}^{22}_{11}\mathbf{L}^{2}_{21}{}^{-1}\mathbf{L}^{1}_{21} & -\mathbf{I}\end{bmatrix}$$

The argument on p.129 in [75] cannot be used to simplify $\mathbf{Q}$ as $\mathbf{Z}^1$ may has zero rows. Therefore, the necessary condition for the existence of the Pseudo Circuit depends on the invertibility of the alternative expression for $\mathbf{Q}$ if the constituent edges of at least one hierarchical component is not in the same partition. This alternative expression would have led to a reconsideration of the Test Point Selection Procedure if the hierarchical partition rule had not been included in the FDA.

6.6. Hierarchical Decision Algorithms

As the diagnosis procedure works in the circuit graph level which does not distinguish between edges from non-hierarchical and hierarchical components, the decision algorithms are re-phrased in terms of the edges as follows:

- Identification of a fault free component from the test results in a test cycle.

  A testee edge will be tested fault free and be moved to the tester partition in the next test cycle if its test result is a pass and it represents the current and voltage of a non-hierarchical component. All the constituent edges of a hierarchical component will be tested ambiguous if the test result of at least one of its constituent edges is a fail. This is because different faults within a hierarchical component do not result in the same constituent edges being faulty. The hierarchical component will be tested fault free and have all its constituent edges moved to the tester partition in the next test cycle if the test results of all its edges are passes.

- Exact Algorithm:

  1. Test results of all testee edges are passes.

     All the testee edges are tested fault free and moved to the tester partition in the next test cycle.

  2. Test result of one testee edge is a fail and the test results of all the other testee edges are passes.

     If the failed edge is non-hierarchical, all the other passed edges will be fault free. Otherwise, all the testee edges, except all constituent edges of the hierarchical component corresponding to the failed edge, will be tested fault free. Those testee edges tested fault free can then be moved to the tester partition in the next test cycle.

  3. Test results of more than one testee edges are fails.

     If at least one of the failed edges is non-hierarchical, or if all the failed edges are hierarchical and are the constituent edges of more than one hierarchical components, then all the testee edges will be fault free. Otherwise, all the testee edges, except the failed edges which are the constituent edges of a hierarchical component, will be tested fault free. Those testee edges tested fault free can then be moved to the tester partition in the next test cycle.

- Heuristic Algorithm

The actions taken for cases 1 and 2 are the same as the actions taken for the respective cases of the Exact Algorithm. The actions required for case 3 are: If all the failed edges are non-hierarchical, all the testee edges except those failed ones will be fault free. Otherwise, a passed edge will only be tested fault free if it is non-hierarchical, or it is hierarchical and it is not one of the constituent edges of the hierarchical components corresponding to the failed hierarchical edges. Those testee edges tested fault free can then be moved to the tester partition in the next test cycle.

6.7. Hierarchical Fault Diagnosis Algorithm

It can be seen from Chapter 6.2 that a root partition matrix and its associated vector $\mathbf{b}_{hier}$ define a set of tree/cotree edge partitions for the hierarchical component concerned. Thus, multiple optimal trees will be derived for a CUT with one hierarchical component if the hierarchical OTG procedure (Chapters 3.2 and 6.3) is applied to the CUT with each of the tree/cotree edge partitions of the hierarchical component taken in turn. The number of possible optimal trees will be increased drastically when more than one hierarchical component is present in the CUT. Clearly, a priority scheme is necessary to select ordered combinations of a tree/cotree edge partition from each hierarchical component in the CUT for the derivation of prioritized optimal trees. This section addresses such a scheme before the hierarchical fault diagnosis algorithm, which results from combining this priority scheme with the enhancements and modifications described in previous sections, is discussed.

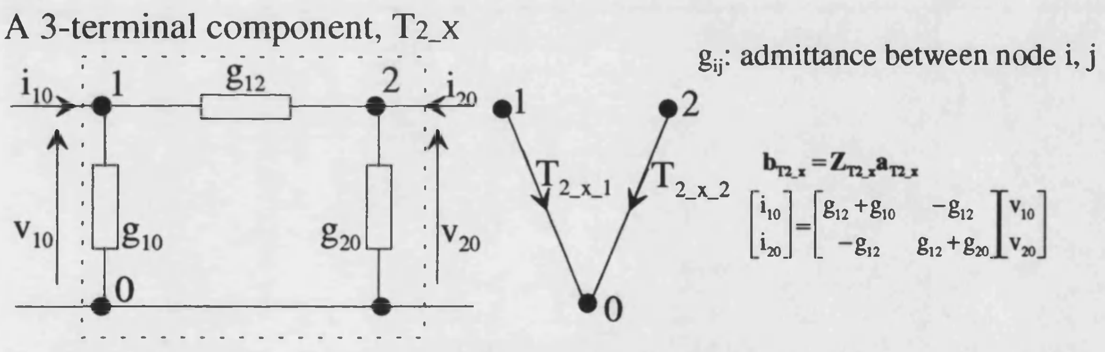
Since it is often desirable, as far as possible, to have the test points as voltages rather than currents, the priority scheme proposed aims to allow the user of the hierarchical FDA to specify preferred hierarchical components in which they can maximise the number of voltage test points without a guarantee that all the test points in the preferred hierarchical components will be voltages. Intuitively speaking, it is very rare that all the possible optimal trees are needed to diagnose faults. At the expense of completeness, the implementation of the scheme is simplified so that it needs only to provide sufficient number of optimal trees prioritized according to the aforementioned criterion.

With reference to Chapter 4, all essential test points selected because of the topology and diagnosability rules are elements of the vector **b**, which is partitioned as $[\mathbf{v}_{tree}\ \ \mathbf{i}_{cotree}]^T$. Therefore, the number of voltage test points on a hierarchical component will be maximised if all of its constituent edges are on the tree. Based on this fact, a partition table is drawn up for the hierarchical component. This table lists all the possible tree/cotree edge partitions for the hierarchical component in descending number of tree edges per partition. In the case of two or more partitions having equal number of tree edges, they are arranged in descending order of the sum of tree edge weights. The entries at the top and bottom of the partition table then have respectively the highest and lowest priority among other entries in being chosen to fill up the hierarchical tree and cotree edge groups in Table 6.3-1 during the component grouping stage of the hierarchical OTG procedure.

When there is more than one hierarchical component in the CUT, the hierarchical components are prioritized in ascending size of their partition tables. In addition, a user can specify preferred hierarchical components in the form of a prioritized list which has higher priority than the list obtained by prioritizing the remaining hierarchical components in ascending size of their partition tables. The presence of a user preference effectively makes the prioritized list two-tier, with the top part of the list determined by the user preference and the bottom part determined by the size of the partition tables. A higher priority component will have a higher possibility of having more voltage test points than that of a component with lower priority. This will become clear shortly with an example.

KCL tells us that there are k-1 independent nodal equations for a CUT with k nodes. This means that the circuit tree has k-1 edges[79][86]. Thus, the number of available tree edges for assigning to the constituent edges of the hierarchical components in the CUT is k-1 less the number of independent voltage sources in the CUT. For the scheme to proceed, the sum of tree edges in all the bottom partitions of the tables must not be greater than the number of tree edges available. The next step is to examine each partition table in this prioritized order, starting from the bottom and working up until the number of tree edges in the partition exceeds the number of tree edges available. This limit is recorded. If there are any remaining tree edges available, the procedure moves to the next partition table and the process is repeated. When all the tree edges are assigned, the remaining hierarchical components have all their edges on the cotree.

It is best to illustrate the scheme with an example: a CUT has a 5 terminal component T_4 (Figure 6.1-1 and Table 6.2-1) and two 3 terminal components T_{2_1} and T_{2_2}. The 3 terminal components are both simple Π networks as shown in Figure 6.7-1 below.


A 3-terminal component, T2_x
g_ij: admittance between node i, j
i10
1
g12
2
i20
v10
g10
g20
v20
0
1
2
T2_x_1
T2_x_2
0

$$\mathbf{b}_{T2\_x} = \mathbf{Z}_{T2\_x}\mathbf{a}_{T2\_x}$$

$$\begin{bmatrix} i_{10} \\ i_{20} \end{bmatrix} = \begin{bmatrix} g_{12}+g_{10} & -g_{12} \\ -g_{12} & g_{12}+g_{20} \end{bmatrix} \begin{bmatrix} v_{10} \\ v_{20} \end{bmatrix}$$

Figure 6.7-1 A simple Π network and its representation as a hierarchical component

For the sake of argument, let us assume that all possible tree/cotree edge partitions for the hierarchical components exist and the number of available tree edges is 6. The partition table for T_4 is already mostly shown in Table 6.2-1 without the root partition (all edges are on the cotree). There are 4 entries in the partition table for each of the two 3 terminal components. They are shown in Figure 6.7-2 for the two scenarios considered in this example. All partition tables show only the tree edges in each partition for simplicity.

With reference to Figure 6.7-2, there is a user preference on T_4 in scenario 1 whereas there is no user preference in scenario 2. Therefore T_4 has the highest and lowest priority in scenarios 1 and 2 respectively. In each scenario, the partitions identified by all the limits set for the partition tables are used to obtain the first optimal tree with the hierarchical OTG procedure.

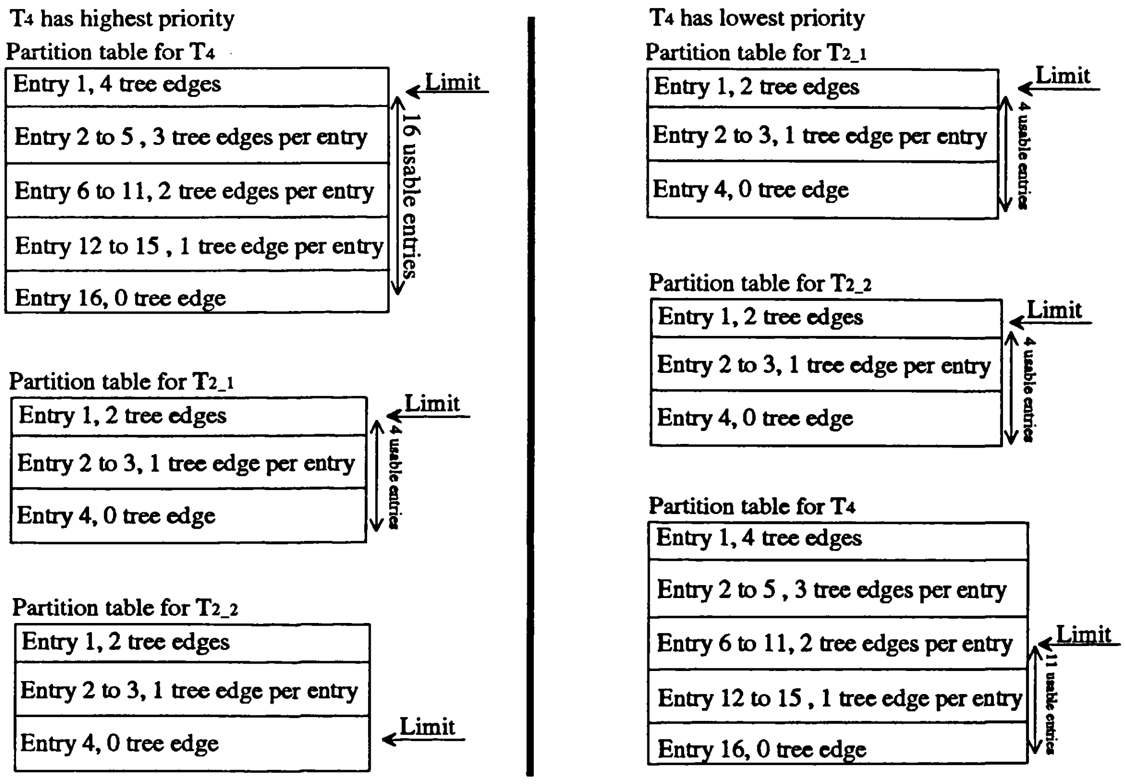


Figure 6.7-2 Scenarios showing how the limit for each partition table is set

In scenario 1, T_4 has the highest priority and has at most 4 voltage test points whereas T_{2_1} has at most 2 voltage test points. However, it may happen that only a test point is chosen on T_4 and 2 test points are chosen on T_{2_1} because of the circuit topology they are in. This is why the priority scheme only provides a higher priority component with a higher possibility of having more voltage test points than that of a component with lower priority.

The limits obtained for all the partition tables are made use of to derive alternative optimal trees with a next tree procedure. This procedure manipulates a set of moving pointers, which are set to point at the same locations recorded by all the limits initially, with two indices, namely, the fixedIndex and movingIndex, which identify two partition tables in the prioritized list. The partitions corresponding to these pointers are then used to derive the next tree after some manipulations are performed.

The procedure begins by assigning the prioritized list index of the partition table with the maximum number of available partitions to the fixedIndex. If the fixedIndex is not the last list index, the movingIndex will be set to the last list index. Otherwise, the movingIndex will be set to the last list index less 1. In both cases the number of hierarchical components is at least 3. To find alternative trees, the pointer for the partition table indexed by the movingIndex is moved down one entry at a time for each new tree needed. If the pointer is already at the bottom entry, it will be reset to its initial position and the movingIndex will be updated according to three different cases before the pointer for the partition table identified by the new movingIndex is moved down one entry to find the next tree. These cases are:

1. fixedIndex equals last list index and movingIndex equals last list index less one.
2. fixedIndex equals 0 (first list index) and movingIndex equals last list index.
3. fixedIndex is between 1 and last list index less 1, and movingIndex equals last list index.

For cases 1 and 2, whenever the movingIndex is updated, it is decreased by one until it equals 1. When the movingIndex is not allowed to decrease further, if its corresponding pointer is again at the bottom entry and one more tree is needed, the pointer will be reset to its initial position and the pointer for the partition table indexed by the fixedIndex will be moved down one entry from its previous position. If more trees are needed, the whole process will be repeated. The actions taken for case 3 are similar to those needed for case 1 apart from the facts that: a) the amount decreased from the movingIndex must not make the new movingIndex equal fixedIndex and b) if the fixedIndex equals 1, the decrement will be 1 each time the movingIndex is updated and will only continue until the movingIndex equals 2. The reason for keeping the pointer for the partition table of the highest priority hierarchical component at its initial position (cases 1 and 3), or keeping it at each of the available entry positions as far as possible (case 2), is to maximise the number of voltage test points on the highest priority hierarchical component.

Having laid the groundwork in the previous sections and paragraphs, it is time to discuss the hierarchical FDA illustrated in Figures 6.7-3 to 6.7-5 in details. A comparison between the basic and hierarchical FDA (Figures 5-1 and 6.7-3) reveals that there are now additional inputs and some changes in the original inputs to the FDA, as well as some changes in the execution of the algorithm.

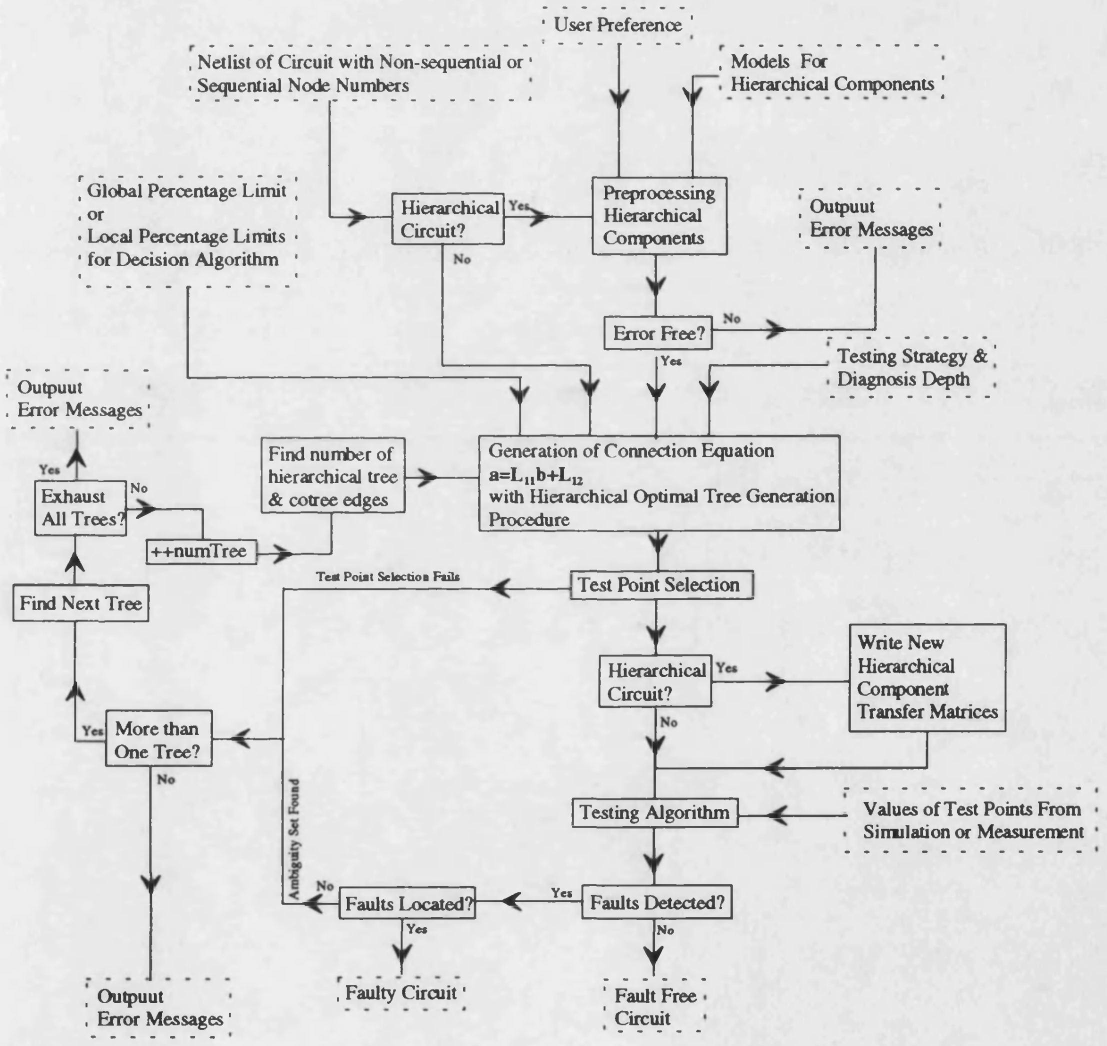

User Preference
Netlist of Circuit with Non-sequential or Sequential Node Numbers
Models For Hierarchical Components
Global Percentage Limit or Local Percentage Limits for Decision Algorithm
Hierarchical Circuit?
Yes
No
Preprocessing Hierarchical Components
Outpuut Error Messages
Error Free?
No
Yes
Testing Strategy & Diagnosis Depth
Outpuut Error Messages
Yes
Exhaust All Trees?
No
Find number of hierarchical tree & cotree edges
Generation of Connection Equation $a=L_{11}b+L_{12}$ with Hierarchical Optimal Tree Generation Procedure
++numTree
Find Next Tree
Test Point Selection Fails
Test Point Selection
Hierarchical Circuit?
Yes
No
Write New Hierarchical Component Transfer Matrices
Yes
More than One Tree?
No
Ambiguity Set Found
Testing Algorithm
Values of Test Points From Simulation or Measurement
No
Faults Located?
Yes
Faults Detected?
Yes
No
Outpuut Error Messages
Faulty Circuit
Fault Free Circuit

Figure 6.7-3 Hierarchical Fault Diagnosis Algorithm

Moreover, the possibility of having incomplete diagnosis results is minimised and the efficiency of the algorithm is enhanced due to the provision of multiple circuit testing trees made possible by the use of the hierarchical approach. This appears as a closed loop which re-enters the connection equation generation stage as a result of incomplete diagnosis due to failure in the test point selection stage or the test outcome that some components become ambiguous (they can be either fault free or faulty). All these changes will be discussed in the following paragraphs.

The test cycle limit input for the testing algorithm is now only needed when the implementation of the hierarchical FDA, the diagnosis program, is run with the option ***openLoop, openLoop+trace*** or ***openLoop+traceAll*** (see Appendix IV). In addition, the input of test point values will be performed more than once if more than one testing tree is involved in the diagnosis. It is also inevitably to have non-sequential node numbers when a CUT is visualised at a higher hierarchical level as some of the circuit nodes in the previous lower hierarchical level will become internal if they are embedded in new hierarchical components found in the higher hierarchical level. Therefore, the hierarchical FDA must be able to accept a circuit netlist with non-sequential node numbers as well as one with sequential node numbers.

The additional inputs to the FDA are: user preference, models for hierarchical components and either global percentage limit or local percentage limits for the decision algorithm. User preference input has already been discussed earlier in this section. The models for hierarchical components are simply their root partition matrices (Chapters 6.1 and 6.2). These matrices are stored in files whose names are coded in the circuit netlist file (Appendix IV) for the diagnosis program to retrieve the matrix files automatically. As for the percentage limit inputs, these have been addressed in Chapter 5.1 and Appendix IV. The local percentage limit inputs can also be used to input percentage threshold values calculated from fault models of hierarchical components before these threshold values can be read directly from files, whose implementation will be a future development for the hierarchical FDA.

The inputs for user preference and models for hierarchical components are needed for the preprocessing of hierarchical components which is shown in Figure 6.7-4. The preprocessing of hierarchical components basically implements the priority scheme up to the point that the first partition of hierarchical tree and cotree edges is available to feed the hierarchical OTG procedure. In addition, it also finds the partition table with the maximum number of available partitions to prepare for the use of the next tree procedure to find alternative trees, should the need arises.

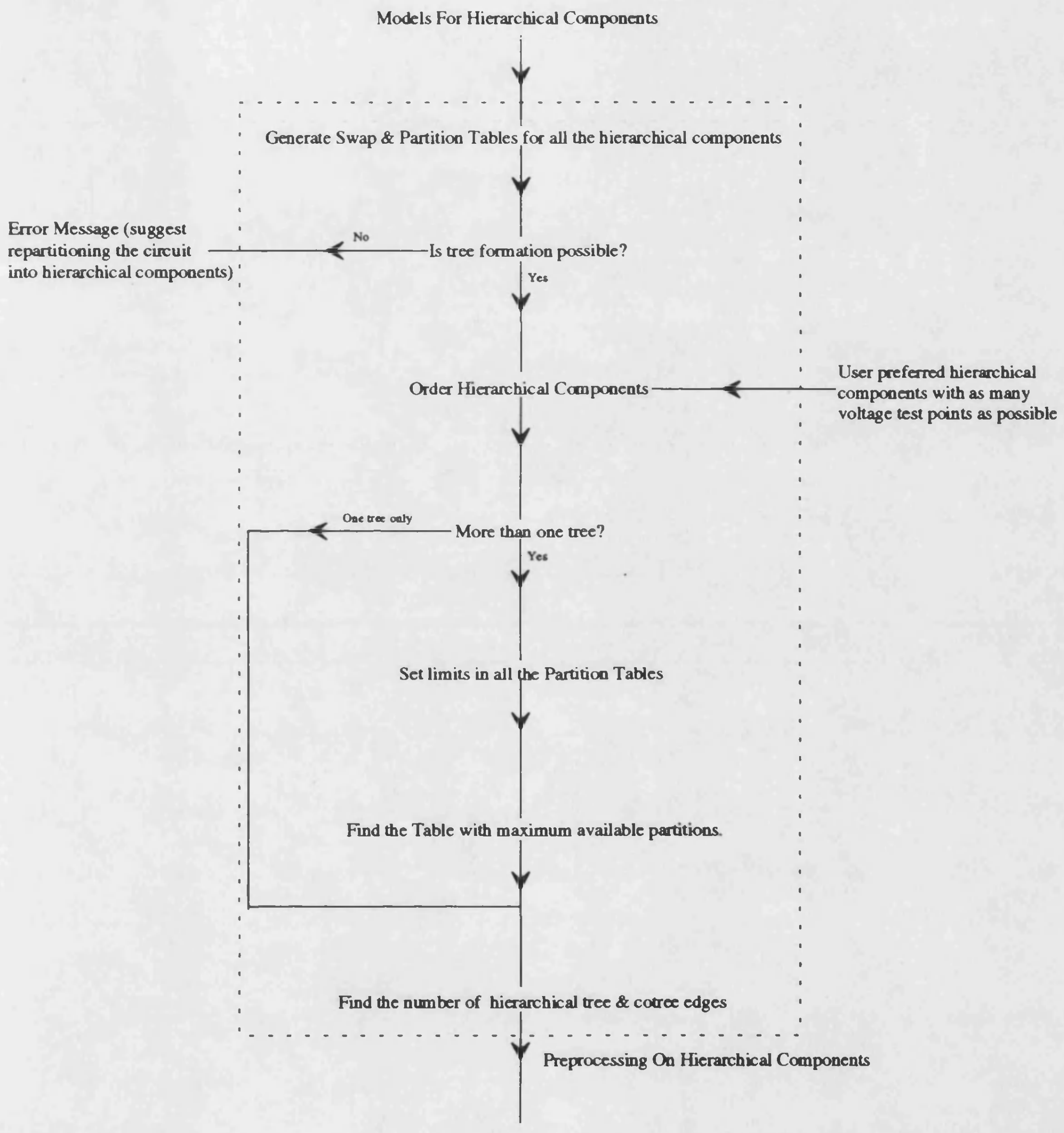

Models For Hierarchical Components
Generate Swap & Partition Tables for all the hierarchical components
Error Message (suggest repartitioning the circuit into hierarchical components)
No
Is tree formation possible?
Yes
Order Hierarchical Components
User preferred hierarchical components with as many voltage test points as possible
One tree only
More than one tree?
Yes
Set limits in all the Partition Tables
Find the Table with maximum available partitions.
Find the number of hierarchical tree & cotree edges
Preprocessing On Hierarchical Components

Figure 6.7-4 Preprocessing of hierarchical components

From the stand point of the implementation of the algorithm, the operations illustrated in Figure 6.7-4 can be divided into two parts: the first part is specific to the hierarchical components and independent of whatever CUT the hierarchical components are in, whereas the second part is CUT specific and has been addressed by the aforementioned discussion on the priority scheme and next tree procedure. As the component transfer matrices of all the possible tree/cotree edge partitions of a hierarchical component is topology independent and component specific, the operations to generate them will only be done once if they are written into files for retrieval later. However, since not all the possible matrices derived from the root partition matrix of the hierarchical component are needed, only the intermediate results to derive them are stored in files and each of these matrices is not derived until it is needed. This is why the derivation of all the hierarchical component transfer matrices is delayed until after the test point selection stage in Figure 6.7-3. The intermediate results needed to derive an alternative matrix is written in a swap file which records the required swapping of elements between the $\mathbf{b}_{\mathbf{hier}}$ and $\mathbf{a}_{\mathbf{hier}}$ vectors of the root partition matrix to yield the new $\mathbf{b}_{\mathbf{hier}}$ and $\mathbf{a}_{\mathbf{hier}}$ vectors of the alternative matrix (Chapter 6.2). In the actual implementation of the herarchical FDA, the partition table of a hierarchical component is derived from a swap table file constructed from the set of swap files corresponding to all the possible tree/cotree edge partitions of the hierarchical component. Details of these files can be found in Appendix IV.

The changes in the execution of the algorithm are mainly in the testing algorithm stage depicted in Figure 6.7-5, in addition the generation of the connection equation is now accomplished with the hierarchical OTG procedure. Three changes are revealed from the comparison between Figures 6.7-5 and 5-2:

- The input for the cycle limit is not necessary for normal operation of the diagnosis algorithm.
- The matrix **Q** instead of **Q'** is used for the testability criterion as the component transfer matrix of a hierarchical component may not be block diagonal because it can have zero row entries.
- With the requirement to impose the hierarchical rule (Chapter 6.5) during component partition, the inter-matrix and inter-vector row exchanges (Chapter 5.4) becomes over-complicated to implement. Instead of trying to obtain an invertible $\mathbf{L}'^{2}_{21}$ matrix when the inverse of the matrix $\mathbf{L}^{2}_{21}$ does not exist, the tableau equation 5.2-6 (Chapter 5.2) has to be

solved although it means inverting a matrix with dimensions comparatively larger than the dimensions of the pseudo circuit matrix **M** (Chapter 5.5).

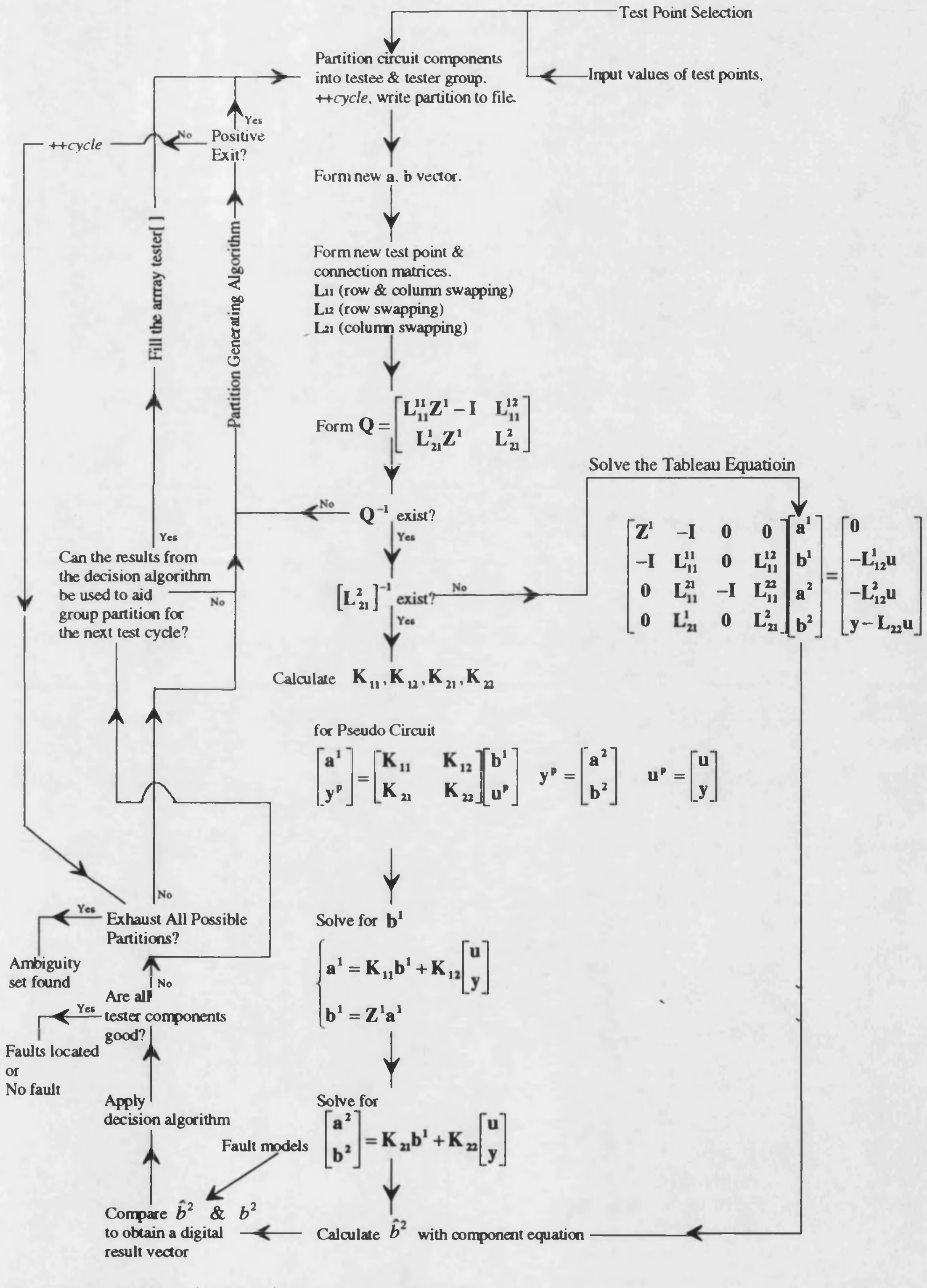

Test Point Selection
Input values of test points.
Partition circuit components into testee & tester group. ++cycle, write partition to file.
Form new a, b vector.
Form new test point & connection matrices.
L11 (row & column swapping)
L12 (row swapping)
L21 (column swapping)
Form $\mathbf{Q} = \begin{bmatrix} \mathbf{L}_{11}^{11}\mathbf{Z}^1 - \mathbf{I} & \mathbf{L}_{11}^{12} \\ \mathbf{L}_{21}^{1}\mathbf{Z}^1 & \mathbf{L}_{21}^{2} \end{bmatrix}$
$\mathbf{Q}^{-1}$ exist?
$[\mathbf{L}_{21}^{2}]^{-1}$ exist?
Yes
No
Solve the Tableau Equatioin
$\begin{bmatrix} \mathbf{Z}^1 & -\mathbf{I} & 0 & 0 \\ -\mathbf{I} & \mathbf{L}_{11}^{11} & 0 & \mathbf{L}_{11}^{12} \\ 0 & \mathbf{L}_{11}^{21} & -\mathbf{I} & \mathbf{L}_{11}^{22} \\ 0 & \mathbf{L}_{21}^{1} & 0 & \mathbf{L}_{21}^{2} \end{bmatrix} \begin{bmatrix} \mathbf{a}^1 \\ \mathbf{b}^1 \\ \mathbf{a}^2 \\ \mathbf{b}^2 \end{bmatrix} = \begin{bmatrix} 0 \\ -\mathbf{L}_{12}^{1}\mathbf{u} \\ -\mathbf{L}_{12}^{2}\mathbf{u} \\ \mathbf{y} - \mathbf{L}_{22}\mathbf{u} \end{bmatrix}$
Calculate $\mathbf{K}_{11}, \mathbf{K}_{12}, \mathbf{K}_{21}, \mathbf{K}_{22}$
for Pseudo Circuit
$\begin{bmatrix} \mathbf{a}^1 \\ \mathbf{y}^p \end{bmatrix} = \begin{bmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{b}^1 \\ \mathbf{u}^p \end{bmatrix} \quad \mathbf{y}^p = \begin{bmatrix} \mathbf{a}^2 \\ \mathbf{b}^2 \end{bmatrix} \quad \mathbf{u}^p = \begin{bmatrix} \mathbf{u} \\ \mathbf{y} \end{bmatrix}$
Solve for $\mathbf{b}^1$
$\begin{cases} \mathbf{a}^1 = \mathbf{K}_{11}\mathbf{b}^1 + \mathbf{K}_{12}\begin{bmatrix} \mathbf{u} \\ \mathbf{y} \end{bmatrix} \\ \mathbf{b}^1 = \mathbf{Z}^1\mathbf{a}^1 \end{cases}$
Solve for $\begin{bmatrix} \mathbf{a}^2 \\ \mathbf{b}^2 \end{bmatrix} = \mathbf{K}_{21}\mathbf{b}^1 + \mathbf{K}_{22}\begin{bmatrix} \mathbf{u} \\ \mathbf{y} \end{bmatrix}$
Calculate $\hat{b}^2$ with component equation
Compare $\hat{b}^2$ & b^2 to obtain a digital result vector
Fault models
Apply decision algorithm
Are all tester components good?
Faults located or No fault
Exhaust All Possible Partitions?
Ambiguity set found
Can the results from the decision algorithm be used to aid group partition for the next test cycle?
Fill the array tester[ ]
Partition Generating Algorithm
Positive Exit?
++cycle

Figure 6.7-5 Testing Algorithm for Hierarchical Fault Diagnosis

6.8. Conclusions

Fault diagnosis of large size circuits with the basic FDA is impractical due to the direct proportionality relationship between circuit complexity and the requirement on computing resources. With hierarchical fault diagnosis, the circuit complexity as seen by the basic FDA is reduced to make fault diagnosis on large size circuits possible. The essence of the hierarchical fault diagnosis is the use of sets of external nodes and voltage/current pairs specific to the analogue circuit building blocks each set abstracts, so that the demand on computing resources necessary to diagnose faults in a large size circuit incorporating such abstractions is reduced to a manageable level. Each circuit block can be at any level of circuit description and can encapsulate other abstractions at lower levels. This encapsulation process goes on until the lowest discrete component level. For this reason, the circuit block is termed a hierarchical component.

A hierarchical component is also called a black box component as the details of the circuit block it abstracts are not necessarily known. What is known are its multiple external nodes and characteristic set of voltage/current pairs which form its graph representation: the constituent graph edges representing the voltage/current pairs are all incident on a common out-node which can be any node in a CUT, whereas all the in-nodes of the edges are the external nodes. These characteristic voltages and currents are elements of the column vectors $\mathbf{b}_{hier}$ and $\mathbf{a}_{hier}$ which are linked by a hierarchical component transfer matrix and partitioned into tree and cotree parts. Knowledge of the tree/cotree partition on either column vector together with the matrix define a set of possible tree/cotree edge partitions for the hierarchical component. Consequently, more than one optimal tree can be derived with these partitions for any circuit containing the hierarchical component. The number of circuit testing trees will build up rapidly when there are numerous hierarchical components in the CUT.

The inclusion of the aforementioned hierarchical approach into the basic FDA necessitates a new priority grouping for circuit components (Chapter 6.3), a re-interpretation of the diagnosability rule in relation to the required minimum number of test points for a given diagnosis depth (Chapter 6.4), the enforcement of a hierarchical rule during component partition at the beginning of each test cycle (Chapter 6.5), a re-interpretation of the decision algorithms in terms of graph edges (Chapter 6.6) and a priority scheme to divide all the hierarchical edges into ordered tree/cotree edge partitions (Chapter 6.7). All these changes have resulted in a hierarchical FDA with improved usability and enhanced efficiency due to the provision of multiple circuit testing trees. However, these improvements and enhancements are gained at the expense of the depth of diagnosability of the CUT. That is, identification of faulty components is limited to the hierarchical level at which the diagnosis is performed. If a hierarchical component is diagnosed as faulty, it will not be possible to distinguish between, or identify faults inside, the faulty component unless a new diagnosis is performed on the faulty component at a lower hierarchy. This gives a degree of flexibility to the diagnosis procedure as it is up to its user to decide whether to devote more computing resources to identify faults within those hierarchical components diagnosed faulty.

7. Effectiveness of the Hierarchical FDA

Having laid out the details of the hierarchical FDA in the previous section, this section benchmarks its effectiveness by applying its ANSI C implementation, the diagnosis program, to diagnose faults in several test circuits. The diagnosis results obtained are then evaluated to see what types of faults simulated can or cannot be diagnosed by the hierarchical FDA, and to investigate the reasons behind the incorrect diagnosis of some types of faults so as to shine light on what improvements are needed on the hierarchical FDA to make it a usable tool for fault diagnosis on practical circuits.

Before these diagnosis results are evaluated, it is worth re-emphasizing how the hierarchical FDA detects faults in comparison with conventional fault detection:

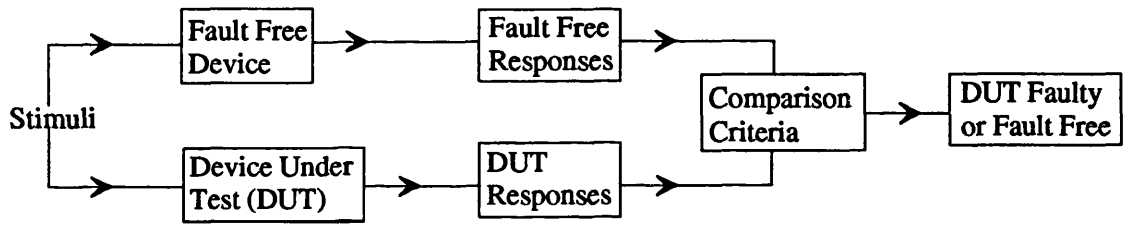
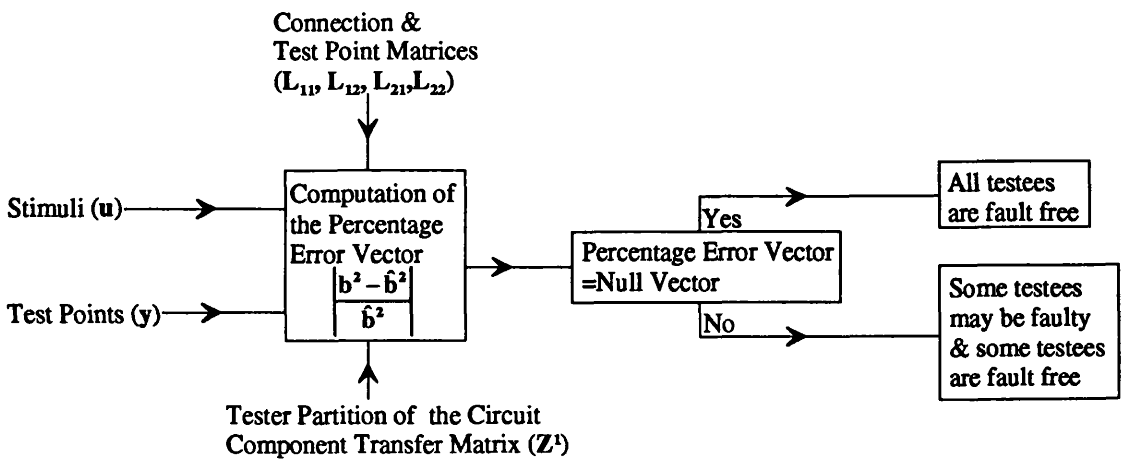


Figure 7-1 Conventional Fault Detection

In Figure 7-1, the responses of the DUT are obtained by measurements on several selected observation points on the device. Fault detection is by comparing the DUT responses with the gold standard, fault free responses against a set of criteria usually derived from the device specifications.



Connection & Test Point Matrices (L_{11}, L_{12}, L_{21},L_{22})
Stimuli (u)
Test Points (y)
Computation of the Percentage Error Vector
$\frac{|b^2-\hat{b}^2|}{\hat{b}^2}$
Tester Partition of the Circuit Component Transfer Matrix (Z^1)
Percentage Error Vector =Null Vector
Yes
No
All testees are fault free
Some testees may be faulty & some testees are fault free

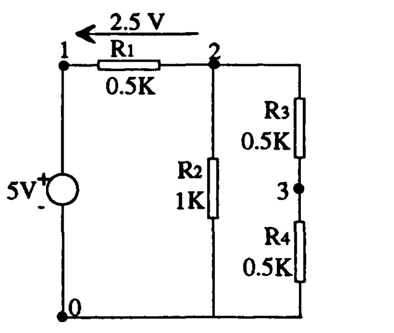
Figure 7-2 Identification of fault free testees by the Hierarchical FDA in a test cycle

In contrast to the conventional manner, Figure 7-2 depicts the gold standard employed by the hierarchical FDA as the condition that the percentage errors (Chapter 5) for all the testee edges must be zero in a test cycle. This condition is in fact true for all possible testable test cycles. However, measurement or simulation errors on the values of the test points will make the testee percentage errors close to instead of exactly zero. Another distinct difference from the conventional approach is that the detection of faults in the DUT is an iterative process involving more than one test cycle as explained in Chapter 5.

In the following evaluations of the diagnosis results of the test circuits, the values of all the test points selected by the diagnosis program are obtained by simulations and are listed in Appendix III.

7.1. *Single Fault Diagnosis*

7.1.1. First Test Circuit



Test Points for 1-diagnosability:
i_{R2}, v_{R4}

Figure 7.1.1-1 First Test Circuit

The simple circuit in Figure 7.1.1-1 was chosen, as the values of the test points under fault free conditions are exact numbers which will not carry rounding up errors when they are used as inputs to the diagnosis program. Applying the diagnosis program to this test circuit under fault free condition has confirmed that there is no inherent numerical errors in the routines to solve the pseudo-circuit and tableau equation as the testee percentage errors are exactly zero in all test cycles. Another feature of the diagnosis program is that the testee percentage errors of a DUT under fault free conditions will be significantly larger than zero instead of exactly zero or close to zero if incorrect values of test points are used as inputs to the program. This feature is used to check that the fault free simulation of the DUT is done correctly before further fault simulations are carried out.

Parameter deviation faults in the test circuit were exhaustively simulated and are diagnosed correctly with the diagnosis program. The diagnosis results are summarized in Table 7.1.1-1.

	Number of Test Cycles Needed	Decision Threshold / %
Fault Free	2	0.1
$R_1 \pm 20\%, R_1 \pm 40\%, R_1 \pm 60\%$ $R_1 \pm 80\%, R_1 + 100\%$	3	10
$R_1 \pm 10\%$	3	5
$R_2 \pm 20\%, R_2 \pm 40\%$	3	10
$R_2 \pm 10\%$	3	5
$R_3 - 10\%$	2	5
$R_3 + 10\%$	3	5
$R_3 \pm 20\%$	3	10
$R_3 \pm 40\%$	2	10

Table 7.1.1-1 Diagnosis results for the parameter deviation faults of the first test circuit

These results have confirmed that the program successfully diagnoses all parameter deviation faults in a flat circuit.

7.1.2. Student Project

Preliminary benchmarking of the hierarchical FDA was carried out by a MEng student working on his third year project[92] in which he applied the hierarchical FDA and transient response technique[93] to diagnose various single faults in the circuits in Figure 7.1.2-2 and drew conclusions on the fault coverage of both testing methods with the diagnosis results obtained. The diagnosis program at that time allowed the tester partition to be manually chosen by the user or generated by the program itself at the beginning of each test cycle, instead of implementing the decision algorithms to automatically re-group the tester partition from the results of a useful previous test cycle, or letting the program generate the tester partition in the case that the previous test cycle does not provide useful results. To perform diagnosis with the program, the project student let the program itself always partition the circuit components, recorded the resulting testee percentage errors and testee/tester partition for each test cycle, and then applied the decision algorithm manually to diagnose the fault after all the results had been collected.

Before using the program to diagnose faults in the circuits in Figure 7.1.2-2, the project student also investigated whether it is necessary to manually change the netlist of a circuit which contains reactive components and is stimulated with an independent DC source, before applying the program to the circuit to diagnose faults. This investigation is necessitated by the fact that the diagnosis program replaces all inductors and capacitors with 1Ω and 100MΩ resistors respectively under DC condition in order to compute the circuit component transfer matrix, instead of reconfiguring the circuit topology to reflect the open and short circuit effects of the respective capacitors and inductors. By avoiding automatic topology reconfiguration, the implementation of the hierarchical FDA has been greatly simplified. On the other hand, the use of equivalent resistance to approximate the short and open circuit effects is intuitively ineffective as the circuit connection equation is no longer a true representation of the circuit topology under DC condition. This intuitive thinking has been justified by the student's investigation in which he applied the program to the fault free circuit in Figure 7.1.2-1.

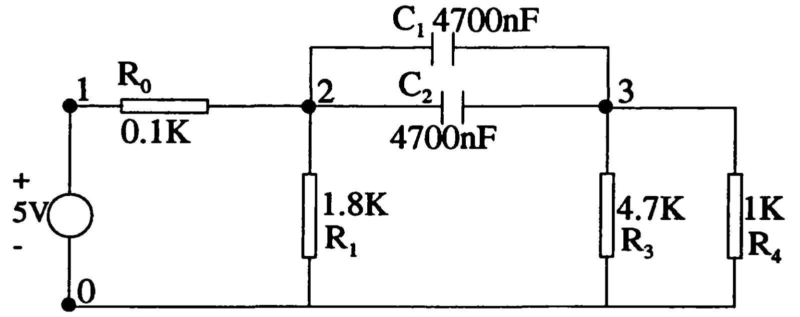
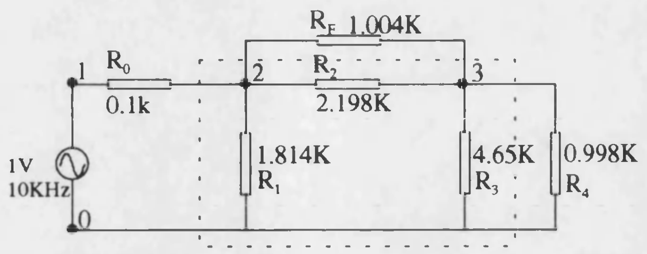


Figure 7.1.2-1 First circuit simulated by the project student

Most of the resulting testee percentage errors for the nine testable test cycles were significantly larger than zero instead of near to zero. When the capacitors in the netlist were replaced with 100MΩ resistors (C_1 and C_2 becomes R_F and R_C respectively) before the program was re-applied to the fault free circuit, the resulting testee percentage errors were all close to zero except in a few cases. The values in these exceptions are between 1% and 4% and in one case is 8.462%.

A circuit block enclosed in dotted line is a hierarchical component

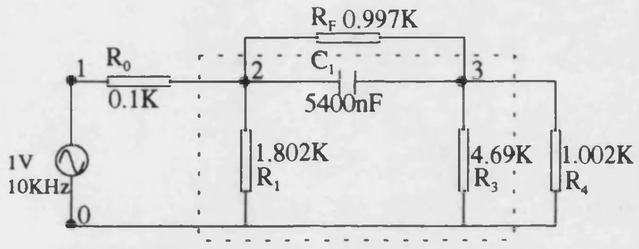


Test Circuit 1

S/C Faults: nodes 2 & 3; nodes 2 & 0

O/C Faults: R_3 removed; R_1 removed

Soft Faults:
R_0=91Ω (R_0-10%), R_2=1997Ω (R_2-10%)
R_4=1105Ω (R_4+10%), R_1=1648Ω (R_1-10%)

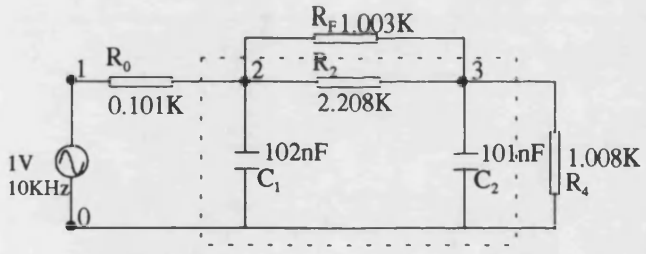


Test Circuit 2

S/C Faults: nodes 2 & 3; nodes 3 & 0

O/C Faults: R_3 removed; R_1 removed

Soft Faults:
R_0=110Ω (R_0+10%), C_1=7530nF (C_1+40%)
R_F=907Ω (R_F-10%), R_1=1951Ω (R_1+10%)

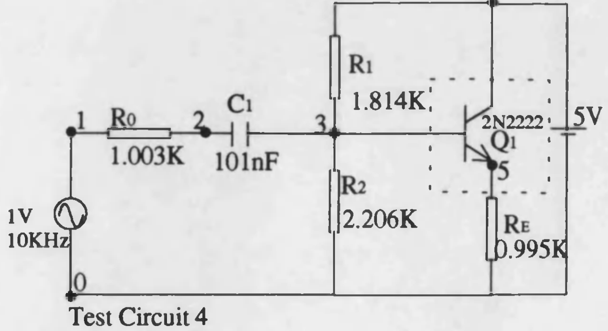


Test Circuit 3

S/C Faults: nodes 2 & 3; nodes 2 & 0

O/C Faults: C_2 removed; C_1 removed

Soft Faults:
C_1=83nF (C_1-20%), C_2=123nF (C_2+20%)
R_4=1105Ω (R_4+10%), R_2=2006Ω (R_2-10%)



Test Circuit 4

S/C Faults: nodes 2 & 3; nodes 2 & 0; nodes 1 & 2

O/C Faults: R_1 removed; R_2 removed; C_1 removed

Soft Faults on R_0, R_2, R_E & C_1

Figure 7.1.2-2
Test circuits constructed and simulated by the project student and the corresponding faults he injected to each circuit

Four test points were chosen in both cases: iR_3, iR_0, vR_3, iC_1 for the former case and iR_4, iR_1, vR_F, iR_F for the latter case. The circuit in both cases are the same to the diagnosis program as the program replaces the capacitors with 100 MΩ resistors under DC condition. However, the differences lie in the values of the common test point (iC_1 and iR_F) and the 3 different test points input to the program, and the tree used in both cases. Different trees are used for the former and latter cases as capacitors are more likely to be on the tree than resistors are (Chapter 3). The simulator, MICROCAP IV, used for the simulation, will give different values of the common test point for the two cases. The reasons for the few above-zero error percentages in the latter case are due to the rounding up errors on the values of the test points and probably the fact that the extreme component values (100MΩ) have exacerbated the rounding up errors during matrix operations performed by the program. As the diagnosis results of the latter case are as expected for a fault free circuit, the simulated values of the test points should be correct in both cases. This leads to the conclusion that the reason for the expected large testee percentage errors in the former case is due to the different circuit topology under DC condition. Thus, the best way to circumvent this problem is to manually remove all the components affected by the DC short and open circuit effects from the circuit netlist before using the program to do diagnosis.

When the project student asked the author why the testee percentage errors for the former case were non-zero, the author actually wrote the netlist file for the circuit in Figure 7.1.2-1 himself for re-running the diagnosis program. Non-zero testee percentage errors were also obtained but the percentage values and the tree the program selected were different from those of the student's. The only difference between the author's and the student's netlist files was the order in which the entries for the capacitors appear in the files. The reason behind this netlist dependency of the tree is apparent when the graph of the circuit is examined: both capacitors have the same edge weights and there was no rule to prioritize equal weight edges within the same component group (Chapter 3). Thus, the rules in Chapter 6.3 have been proposed to circumvent the ambiguity brought about by the allocation of the column indices of the node incidence matrix to equal weight edges with the same component type.

Prior to the evaluation of the diagnosis results on the faults in the test circuits in Figure 7.1.2-2, it is best to let the reader to have an understanding on the manner these faults were introduced into these circuits and how the diagnosis results were obtained with these faults: in order for the same circuits to be tested with the transient response technique[93], these test circuits were physically built on breadboard. The measured values of the components on the physical test circuits were then used to construct the netlist files (Appendix II) for the diagnosis program. Flat and hierarchical netlist files were written for each of the test circuits 1, 2 and 3 (Test circuit 4 has only a hierarchical netlist). In the case of a hierarchical netlist, the hierarchical component (T_1 in the case of the test circuits 1, 2 and 3) is the circuit block enclosed by the dotted line in each test circuit. The component transfer matrices for the hierarchical components in the test circuits 1, 2 and 3 were obtained with the respective measured component values and the formula listed in Figure 6.7-1. For test circuit 4, the methods to derive the component transfer matrix of the BJT has been explained in [92].

Each netlist file would be an input to the diagnosis program regardless of whether the corresponding test circuit is faulty or fault free. The differences between the inputs to the program for faulty and fault free circuits would be the values of the test points which were obtained from simulations of the faulty and fault free circuits respectively.

Before any fault was introduced into a test circuit, the transient response equipment[92] was used to get the auto-correlation on the gold standard data for the test circuit, and a simulation was performed on the test circuit to provide the values for the test point inputs to the diagnosis program. The auto-correlation would be needed for comparison with the cross-correlation obtained from the test circuit after a fault had been introduced, whereas the fault free simulation would enable the verification of all-zero/near-zero testee percentage errors for all testable test cycles so that the program could be used to diagnose faults in the test circuit.

A fault was injected into a test circuit by replacing a component with another one of which its measured value is closest to the faulty component value. This new faulty circuit would then be tested by the transient response equipment[92] to get its characteristic cross-correlation whereas the measured value of the replacing (faulty) component would be used in a re-simulation to yield the test point values for the faulty test circuit. There would be two sets of test point values for the test circuit with flat and hierarchical netlist files although some of the test points in the two sets might be the same.

The project student attempted diagnosis for all the faults listed in Figure 7.1.2-2 with the flat netlist files describing the test circuits 1, 2 and 3 to the diagnosis program. In addition, he tried to diagnose all the short (s/c) and open circuit (o/c) faults listed in the same figure with the hierarchical netlist files as inputs to the program. Entries for the hierarchical component transfer matrices and the simulated test point values were kept to six significant figures for all cases. Below is a brief summary of, and the author's evaluation on, the diagnosis results which appear as tables of testee percentage errors versus testable test cycles in his report[92]:

- Diagnosis results for fault free circuits

  The testee percentage errors in all testable test cycles for the test circuits 1, 2 and 3 in flat netlist description were all less than 0.01%. When the hierarchical netlist format was used, some of the testee percentage errors increased slightly but they were still less than 0.1% except in one test cycle of test circuit 2. In this particular test cycle, the percentage error on the testee R_F was 24.72%. The main reason is that the impedance of the capacitor C_1 becomes so low at the 10 KHz test frequency that nodes 2 and 3 are nearly short circuit. This reason is testified by the fact that the percentage error on R_F falls to less than 0.1% when the capacitance of C_1 is reduced to 5.4 nF.

These results showed that the fault free simulations on the test circuits 1, 2 and 3 were done correctly so that further fault simulation results on the test circuits could be used to test the effectiveness of the diagnosis program. The slight increase in some of the testee percentage errors with the hierarchical netlist description is attributed to the rounding up errors in computing the hierarchical component transfer matrices. These rounding up errors, together with the nearly short circuit connection made by C_1 in test circuit 2, manifest themselves as a large percentage error on R_F. Therefore, no evaluation is made on the diagnosis results of test circuit 2 with the hierarchical netlist format.

As for test circuit 4, its testee percentage errors were all significantly larger than 0 mainly due to the difficulty in establishing the correct component transfer matrix (in the form of an admittance matrix) for the BJT Q_1[92]: the project student used the transistor parameters (v_{be},v_{ce}, i_b and i_c) MICROCAP IV provided to calculate the small signal h parameters (formulae exist to convert the h parameters to the entries in the required admittance matrix) before attempting to derive the admittance matrix with measurements by short-circuiting the base-emitter and collector-emitter terminals. (Alternative measurements required to derive the h parameters of a BJT are detailed in Sedra[94]) Deriving the admittance matrix with measurement data is the correct approach as, although the testee percentage errors resulting from this method are still larger than 0, they are comparatively smaller than those errors obtained from using the typical transistor parameters MICROCAP IV provided. Measurement accuracy plays a crucial part in deriving the correct transistor admittance matrix as it is impossible to have accurate matrix entries while the measurements are made with an oscilloscope without digital readout (only crude measurement is possible as the accuracy is limited by the divisions on the x and y axes on the scope display, and the scales these axes represent). Even if the correct admittance matrix is derived, the default parameters MICROCAP IV used in a simulation to obtain the test point values may have to be adjusted to achieve satisfactory testee percentage errors. On the other hand, it is not known how sensitive the FDA is to a change in measurement accuracy. This aspect is explored in Chapter 7.1.6.

Due to the situation described in the aforementioned paragraph, the project student did not use test circuit 4 to test the diagnosis program and all other test circuits discussed in the rest of Chapter 7 do not include transistors as separate circuit components.

- Diagnosis results for faulty circuits

The diagnosis results of the test circuits 1, 2 and 3 for the faults listed in Figure 7.1.2-2 are summarised in Table 7.1.2-1.

	Flat Netlist Description			Hierarchical Netlist Description	
	Soft Faults	O/C Faults	S/C Faults	O/C Faults	S/C Faults
Test Circuit 1	Diagnosed	No diagnosis on the fault on R_3 but the fault on R_1 is diagnosed	No diagnosis	T_1 is diagnosed faulty when R_3 is removed, when R_1 is removed, T_1 & R_0 are diagnosed ambiguous	No diagnosis
Test Circuit 2	Diagnosed	Incorrect diagnosis	No diagnosis	Diagnosed	No diagnosis
Test Circuit 3	Diagnosed	Diagnosed	No diagnosis	T_1 is diagnosed faulty when C_2 is removed, when C_1 is removed, T_1 & R_0 are diagnosed ambiguous	No diagnosis

Table 7.1.2-1 Summary of diagnosis results for the student project

The results recorded in Table 7.1.2-1 have reinforced one of the conclusions of Chapter 7.1.1 that the diagnosis program can diagnose parameter faults in non-hierarchical circuits. When hard faults are present in a circuit, intuition suggests that these faults will be diagnosable with the FDA only if they occur deep within hierarchical components. If these faults, in particular short circuit faults, occur in a flat circuit or in the boundary between a hierarchical component and the CUT, circuit topology will be changed and therefore the FDA will not always diagnose these faults. The more the circuit topology is affected by the hard faults, the more likely that the diagnosis results of the FDA is misleading instead of just non-diagnosis of the faults.

This intuitive conclusion has been partially proven by the diagnosis results for open circuit faults in the test circuits with flat and hierarchical netlist description: in the flat netlist cases, the change in circuit topology effected by the open circuit faults either misleads the program to diagnose good components as faulty, or results in the program not always diagnosing the faults. Whereas in the hierarchical netlist cases, diagnosis of the open circuit faults are not always guaranteed as the faults occur on the boundary of the hierarchical component and there are only two testable test cycles available for the FDA to do the diagnosis.

There is no diagnosis for all the short circuit faults on the test circuits and one open circuit fault on test circuit 1, as many of the testee percentage errors calculated by the diagnosis program are not numbers which can be represented by the precision of the workstation. These non-numbers, denoted by INF (infinity) and NAN (not a number), are the results of two program design errors which were not noticed when the program was given to the project student. The number INF is a "divide by zero" error which results whenever an element of $\hat{b}^2$ (Figure 7-2) is zero. This has been circumvented by replacing any zero element of $\hat{b}^2$, which can be either a current or voltage quantity, with a predefined tiny value of current or voltage before the division is performed. The cause for the NAN error is less obvious than the INF error: It had been traced back to the function which performs the LU decomposition[87] of a matrix. An essential step of this decompositon is to use the diagonal elements of an intermediate matrix as denominators in divisions. If any of the diagonal elements is zero, the matrix will not be LU decomposable which implies that it is singular. For a near singular matrix, these diagonal elements are very minute numbers close to zero, which cause huge numeric errors in the subsequent matrix inversion. This problem has been addressed by imposing a check on these diagonal elements so that the matrix is considered singular if any diagonal element is less than a predefined tiny quantity.

After reviewing the diagnosis results for the test circuits in this and the previous sections, the ability of the FDA to diagnose single parameter deviation faults in a non-hierarchical circuit has been proven whereas the major advantages of having hierarchical components in a circuit are only partially demonstrated. These advantages are diagnosis of hard faults entirely buried in a hierarchical component and the reduction in circuit complexity, which leads to saving in the computing resources needed to perform the fault diagnosis. The test circuits in this section have shown that the FDA is unable to give reliable diagnosis results on open circuit faults in a flat circuit, and it is also incapable of diagnosing such open circuit faults which are not completely embedded in a hierarchical component. However, the diagnosis results of the FDA in the hierarchical case is more reliable than those in the flat circuit as there are no misleading results. Nevertheless, instead of disproving the intuitive conclusions made previously on the locations of hard faults which the FDA cannot diagnose, these results have reinforced these conclusions. The diagnosis of hard faults within a hierarchical component is explored further with the passive network in Chapter 7.1.3, which together with the filter circuit in Chapter 7.1.4, again demonstrate the advantage of reduced circuit complexity with the hierarchical approach. Although the reduction in circuit complexity of these test circuits is not large (from a 7 edge to a 6 edge graph), these results have also reminded us of the price to pay for the complexity reduction: The fact that the ratio of the number of testable test cycles to the number of possible test cycles decreases from 9/15 in the flat netlist case to 2/4 in the hierarchical netlist case, means that a hierarchical circuit is less testable than its non-hierarchical counterpart. This deterioration in circuit testability depends on the level of hierarchical abstraction adopted for a CUT. Generally speaking, the higher the level of hierarchical abstraction is, the more severe this deterioration in circuit testability will be.

7.1.3. Second Test Circuit

The effectiveness of the FDA in diagnosing both hard and soft faults in a hierarchical circuit is investigated in this section with the test circuit in Figure 7.1.3-1, whose graph representation for its non-hierarchical description is illustrated in Figure 7.1.3-2.

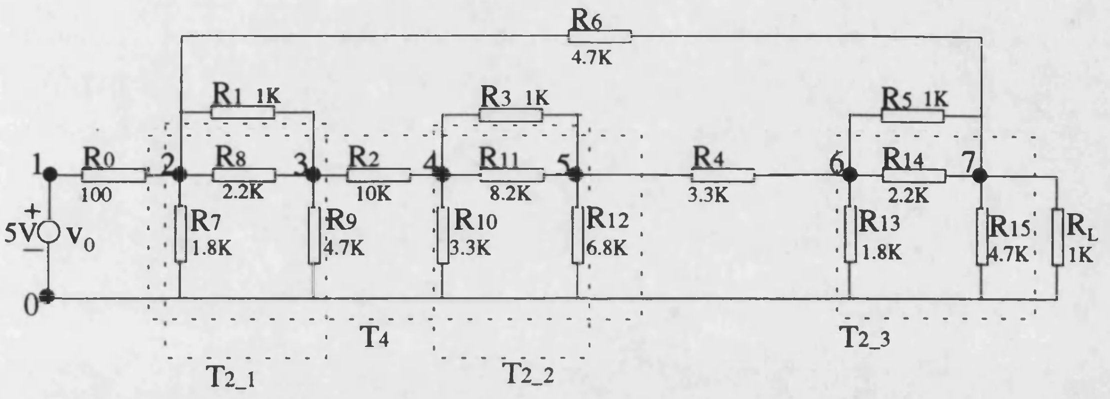


Figure 7.1.3-1 Passive Network

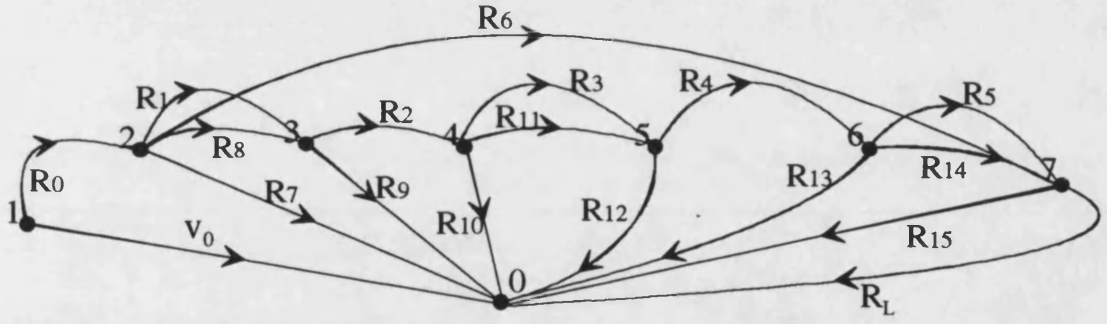


Figure 7.1.3-2 Flat Circuit Graph of the Passive Network

It is clear that there are many ways to lump components of this test circuit into hierarchical components. Two of the possible combinations of hierarchical components, which make use of the formulae for the component transfer matrices listed in Figures 6.1-1 and 6.7-1, are shown by enclosing the blocks of components of the test circuit with a dotted line. These combinations are one 4-edge and 2-edge hierarchical components (T_4 and T_{2_3}), or three 2-edge hierarchical components (T_{2_1}, T_{2_2} and T_{2_3}). The circuit realizations and graph representations corresponding to these combinations are depicted in Figures 7.1.3-3, 7.1.3-4, 7.1.3-5 and 7.1.3-6. Given these different hierarchical descriptions of the same test circuit, the immediate question needing to be answered is: which hierarchical description should be adopted for the investigations of soft and hard faults in the test circuit? A re-interpretation of this question is: which description is more efficient for fault diagnosis?

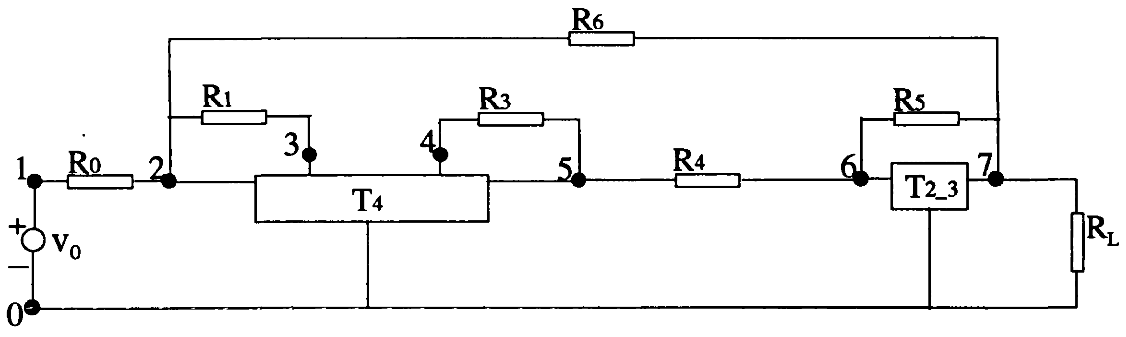


Figure 7.1.3-3 Realization of the Passive Network as Hierarchical Circuit 1

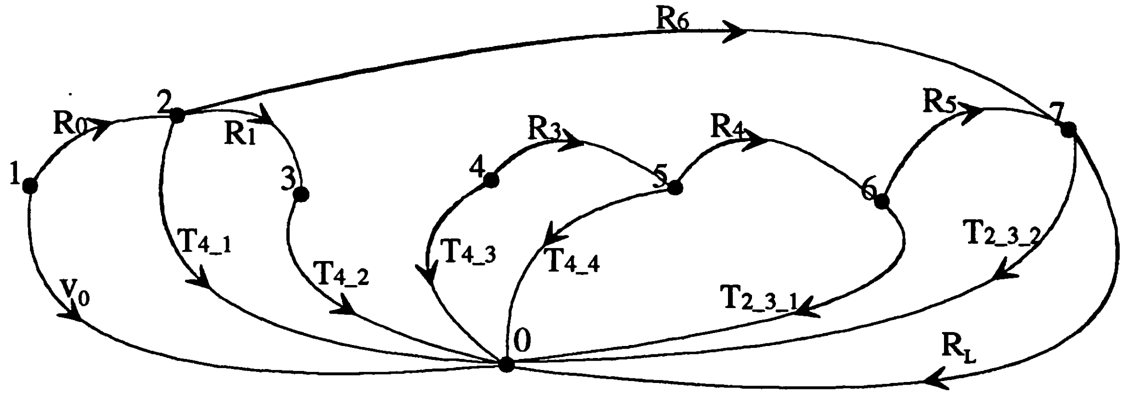


Figure 7.1.3-4 Graph of Hierarchical Circuit 1

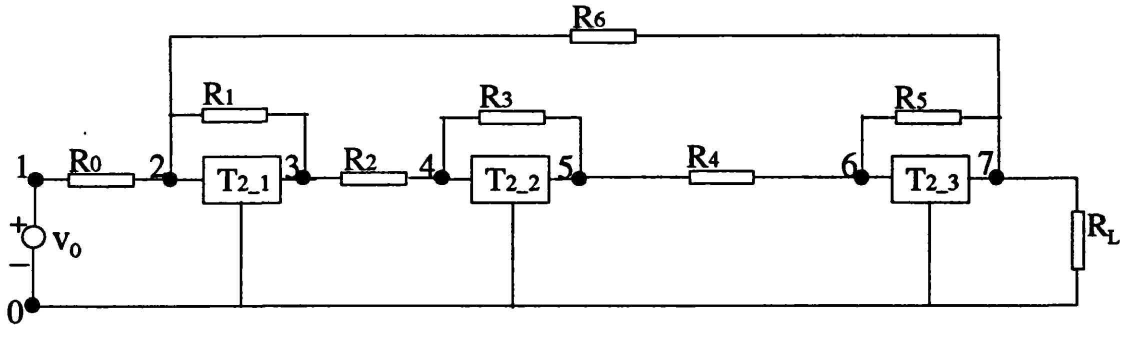


Figure 7.1.3-5 Realization of the Passive Network as Hierarchical Circuit 2

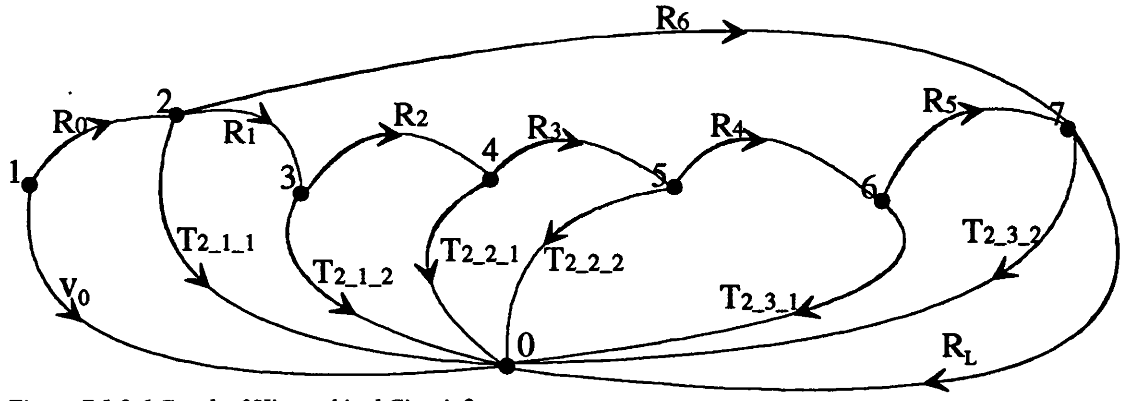


Figure 7.1.3-6 Graph of Hierarchical Circuit 2

It is intuitively true that the less the number of test cycles and test points required for the FDA to yield the diagnosis results of the test circuit in a particular circuit realization, the more efficient such a circuit realization is compared with other realizations of the same circuit in diagnosing faults. In addition, the intuitive criterion, the number of test cycles required, takes priority over the other criterion, the number of required test points, should there be a conflict between them. Thus, these questions have been answered by applying the diagnosis program to the different realizations, including the non-hierarchical realization, of the test circuit under fault free conditions, to see which realization requires the least number of test cycles and test points to diagnose the fault free conditions. The required test point values were obtained with the simulator MICROCAP IV and are listed in Appendix .

Realization of Circuit	Number of test points selected	Number of possible component partitions	Number of Test Cycles needed for diagnosis of the fault
Hierarchical Circuit 1, (14, 10)[2]	7	1716	5, (3/5)[1]
Hierarchical Circuit 2, (15, 12)[2]	8	3003	97, (13/97)[1]
Flat Circuit, (18, 18)[2]	11	12376	248, (48/248)[1]

[1]Ratio of the number of testable test cycles to the number of total test cycles performed.
[2]Number of edges in the circuit graph, number of components in the circuit.

Table 7.1.3-1 Comparison on the FDA Efficiency of Different Realizations of the Passive Network

From the comparison shown in Table 7.1.3-1, it is obvious that the test circuit should be realized as hierarchical circuit 1, which employs the hierarchical components T_4 and T_{2_3}, for benchmarking the effectiveness of the FDA in diagnosing soft and hard faults. The fact that there are always untestable test cycles in the total test cycles performed for all different circuit realizations (note[1] of the table), has proved that the original ST Algorithm (Chapter 5.1) will not always converge without the ability to generate an arbitrary component partition which must not repeat the partitions already used for all previous test cycles. Table 7.1.3-1 also demonstrates the reduction in circuit complexity in terms of the number of components and graph edges, and the number of possible component partitions for the hierarchical circuits, as compared to those of the flat circuit. Due to the hierarchical rule explained in Chapter 6.5, the number of possible component partitions equals the number of possible test cycles only when a circuit is non-hierarchical. In the case of a hierarchical circuit, the number of possible partitions is a lot larger than the possible number of test cycles. Both the possible number of test cycles and the total number of testable test cycles can be counted with the diagnosis program running in the ***openLoop*** mode (Appendix IV), but it is impractical to do it except for very simple circuits. Therefore, no attempt has been made to justify again the statement, "a hierarchical circuit is less testable than its non-hierarchical counterpart", made in the previous section with this test circuit.

Having chosen the hierarchical description for the test circuit, parameter deviation faults of +5%, -5%, +50% and -50% on three components in the test circuit were simulated to obtain the test point values required for the diagnosis program. These faults were all correctly diagnosed and their diagnosis results are summarized in Table 7.1.3-2.

Soft Fault on Component, Decision Threshold	Percentage Deviation of Component Values/%	Number of Test Cycles needed for diagnosis of the fault	Faulty Component or Edges
R_0, 1%	+5, -5, +50, -50	7, (4/7)[1]	R_0
R_2, 1%	+5, -5	5, (3/5)[1]	$T_{4\,3}$, $T_{4\,4}$
R_2, 1%	+50, -50	5, (3/5)[1]	$T_{4\,1}$, $T_{4\,2}$, $T_{4\,3}$, $T_{4\,4}$
R_{14}, 0.2%	+5, -5	4, (3/4)[1]	$T_{2\,3\,1}$, $T_{2\,3\,2}$
R_{14}, 1%	+50, -50	4, (3/4)[1]	$T_{2\,3\,1}$, $T_{2\,3\,2}$

[1]Ratio of the number of testable test cycles to the number of total test cycles performed.

Table 7.1.3-2 Diagnosis Results for Soft Faults in the Passive Network

Three salient features can also be observed from the diagnosis results in Table 7.1.3-2. One of these features is typical of the fault diagnosis process of the FDA whereas the other two features are characteristic of a hierarchical component. Concerning the diagnosis process, the different number of test cycles needed to diagnose the faults have indicated that the FDA does not always take the same route (i.e. use the same set of component partitions) to converge for different faults. However, if the soft faults occur on the same component or constituent element of a hierarchical component, the route taken by the FDA to diagnose the faults will be usually the same. Considering the diagnosis results for the soft faults on the constituent element (R_2) of T_4, different deviations from the component value manifest themselves as different sets of faulty constituent edges of T_4. This implies that a single fault on any constituent element of a hierarchical component can manifest itself as any number of constituent edges of the hierarchical component being faulty. When there is more than one fault in the hierarchical component, the manifestation of faults will also appear as any number of constituent edges of the hierarchical component being faulty. Therefore, the FDA can only diagnose whether a hierarchical component is faulty, it cannot distinguish between a single fault and multiple faults inside the hierarchical component. This has confirmed the sacrifice made on the depth of diagnosability (Chapter 6.8) for the use of hierarchical approach. Lastly, it can be told from the different decision threshold values needed to diagnose the faults in Table 7.1.3-2 that the hierarchical component T_4 has a much lower fault tolerance than that of T_{2_3}. The higher the fault tolerance of a hierarchical component, the lower the decision threshold must be set to detect faults within the hierarchical component. A high fault tolerance component such as T_{2_3}, has only small percentage deviations on most of its characteristic voltages and currents. This is why it needs a 0.2% decision threshold to diagnose 5% component deviation faults on R_{14}.

Having proven the ability of the FDA to diagnose soft faults, hard faults in the test circuit were exhaustively simulated to provide the test point values for the diagnosis program. A 1% decision threshold value was used for the subsequently diagnosis and the results are summarized in Table 7.1.3-3. These results have shown that the FDA can successfully diagnose all the hard faults apart from a few short circuit faults. Although all the hard faults occur on the boundary between the hierarchical components and the test circuit (Figures 7.1.3-1 to 7.1.3-4), the topology of the test circuit at the hierarchical level as seen by the FDA is not affected by the open circuit faults. Hence, all open circuit faults are diagnosed. On the other hand, each short circuit fault changes the topology of the test circuit to a varying degree so that the connection equation, which represents the topology at the hierarchical level (Figure 7.1.3-3), no longer truly represent the changed topology. This mismatch between the topology perceived by the FDA and the actual topology effected by the short circuit fault, can sometimes lead to incorrect or misleading diagnosis results as reported in Table 7.1.3-3. Whether the FDA will give correct diagnosis results for a short circuit fault which affects the circuit topology is impossible to predict as this also depends on other inputs to the diagnosis program. For example, when the program was re-run on the test circuit with a short circuit fault on R_{11} using the Heuristic instead of the Exact decision algorithm (Chapter 6.6), the diagnosis results became misleading as R_3 was also diagnosed faulty.

Hard Fault on Component	Diagnosis Results	Number of Test Cycles needed for diagnosis of the fault	Faulty Component or Edges
R_2, O/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_2, S/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_7, O/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_7, S/C	Misleading	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}, R_L, (T_{2\,3\,1}, T_{2\,3\,2})$[2]
R_8, O/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_8, S/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_9, O/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_9, S/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_{10}, O/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_{10}, S/C	Misleading	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}, R_L, (T_{2\,3\,1}, T_{2\,3\,2})$[2]
R_{11}, O/C	Correct	5, (3/5)[1]	$T_{4\,3}, T_{4\,4}$
R_{11}, S/C	Correct	5, (3/5)[1]	$T_{4\,3}, T_{4\,4}$
R_{12}, O/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_{12}, S/C	Correct	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}$
R_{13}, O/C	Correct	4, (3/4)[1]	$T_{2\,3\,1}, T_{2\,3\,2}$
R_{13}, S/C	Correct	4, (3/4)[1]	$T_{2\,3\,1}, T_{2\,3\,2}$
R_{14}, O/C	Correct	4, (3/4)[1]	$T_{2\,3\,1}, T_{2\,3\,2}$
R_{14}, S/C	Misleading	5, (3/5)[1]	$T_{4\,1}, T_{4\,2}, T_{4\,3}, T_{4\,4}, R_L, (T_{2\,3\,1}, T_{2\,3\,2})$[2]
R_{15}, O/C	Correct	4, (3/4)[1]	$T_{2\,3\,1}, T_{2\,3\,2}$
R_{15}, S/C	Incorrect	4, (3/4)[1]	$(T_{2\,3\,1}, T_{2\,3\,2})$[2]

[1]Ratio of the number of testable test cycles to the number of total test cycles performed.
[2]Edges or component which were diagnosed as faulty in the last test cycle but were diagnosed as fault free in previous test cycles.

Table 7.1.3-3 Diagnosis Results for Hard Faults in the Passive Network

Additionally, these diagnosis results have again reinforced the fault manifestation feature of a hierarchical component made previously in the case of soft faults: all hard faults except the faults on R_{11} result in all constituent edges of T_4 being diagnosed faulty.

This section has complemented the insufficient proof provided by the diagnosis results in the previous two sections for the effectiveness of the FDA to diagnose faults in a hierarchical circuit. In view of the diagnosis results for soft faults, the ability of the FDA to diagnose single parameter deviation faults in a hierarchical circuit has been proved beyond doubt. In addition, these diagnosis results have also revealed that different hierarchical components have different fault tolerances and a fault within a hierarchical component can manifest itself as any number of the constituent edges of the hierarchical component being faulty. Fault tolerance affects the value of the decision threshold needed to diagnose faults within a hierarchical component. A low decision threshold is required to detect faults in a hierarchical component with high fault tolerance. The implication of fault manifestation in a hierarchical component is that the FDA can only diagnose the hierarchical component as faulty and will not know how many faults the hierarchical component has. Apart from the revelation of these two characteristics of a hierarchical component, these diagnosis results have confirmed that the FDA does not always use the same set of component partitions to perform diagnosis for different faults.

As for the evaluation of the diagnosis results for hard faults, they have consolidated the intuitive conclusions made in the previous section that the FDA will not always diagnose these faults, in particular short circuit faults, if these faults affect the circuit topology at the hierarchical level. These topology-changing faults are, without exception, not entirely buried in a hierarchical component. However, faults which are not completely embedded in a hierarchical component are not necessarily topology-changing. This is why all the open circuit faults in the test circuit were correctly diagnosed despite the fact that they are not totally enclosed by the hierarchical components. Open circuit faults are, by their very nature, less prone to make changes in the circuit topology than short circuit faults.

The issue of choosing a suitable circuit realization from different available hierarchical realizations of a circuit for fault diagnosis has also been addressed, by considering their diagnosis efficiency in terms of the required number of test cycles for the FDA to converge and the number of selected test points for each realization under fault free conditions. However, structural decomposition of a practical analogue circuit[95] into hierarchical building blocks demands a radically different approach and is out of the scope of this research work. Nevertheless, these different circuit realizations demonstrate again the reduction in circuit complexity with the hierarchical approach in terms of the number of components and graph edges, and the number of possible component partitions. Their diagnosis results together with the diagnosis results for the non-hierarchical realization under fault free conditions have also justified the additional capability of the FDA to generate an arbitrary component partition which does not repeat those partitions already used in previous test cycles. This added capability is crucial for the convergence of the FDA as not every test cycle has a testable component partition (Figure 5-2).

7.1.4. Third Test Circuit

As the test circuits chosen to date are made up of trivial circuits, the test circuit adopted in this section is a band pass filter circuit with many practical applications. One such application is in automotive electronics for the detection of engine knock[96]. Integrated circuit realization of such a circuit is usually in the form of a switched capacitor filter due to the more precise control on capacitor values than that can be achieved with resistor values. However, the FDA cannot diagnose faults in switch capacitor circuits as the conventional circuit graph only conveys voltage/current information instead of the voltage/charge information inherent in such circuits. Therefore, a time continuous, instead of a switched capacitor, implementation of the band pass filter shown in Figure 7.1.4-1[97] is used.

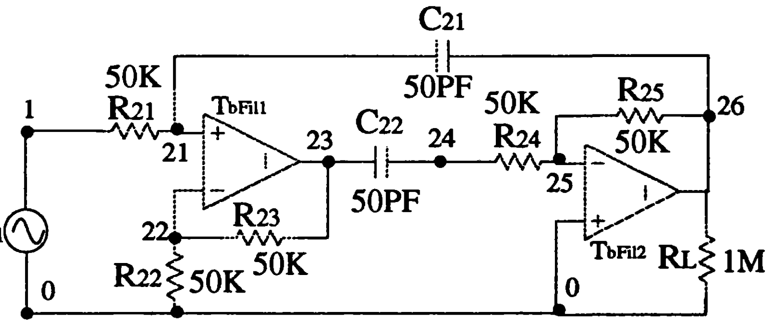


Figure 7.1.4-1 Band Pass Filter Circuit

This filter has been designed with a centre frequency around 35KHz and the attenuation is 0.8 of the signal magnitude at the frequencies 25KHz and 55KHz. The opamps T_{bFil1} and T_{bFil2} are represented as 3 edge and 2 edge hierarchical components respectively in the circuit graph in Figure 7.1.4-2. Their component transfer matrices assume the forms explained in Chapter 6.1 and are rewritten with the notation in Figure 7.1.4-1 for clarity:

$$\begin{pmatrix} i_{TbFil1\_1} \\ i_{TbFil1\_2} \\ v_{TbFil1\_3} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ A_v & -A_v & 0 \end{pmatrix} \begin{pmatrix} v_{TbFil1\_1} \\ v_{TbFil1\_2} \\ i_{TbFil1\_3} \end{pmatrix}, \quad \begin{pmatrix} i_{TbFil2\_1} \\ v_{TbFil2\_2} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ -A_v & 0 \end{pmatrix} \begin{pmatrix} v_{TbFil2\_1} \\ i_{TbFil2\_2} \end{pmatrix}$$

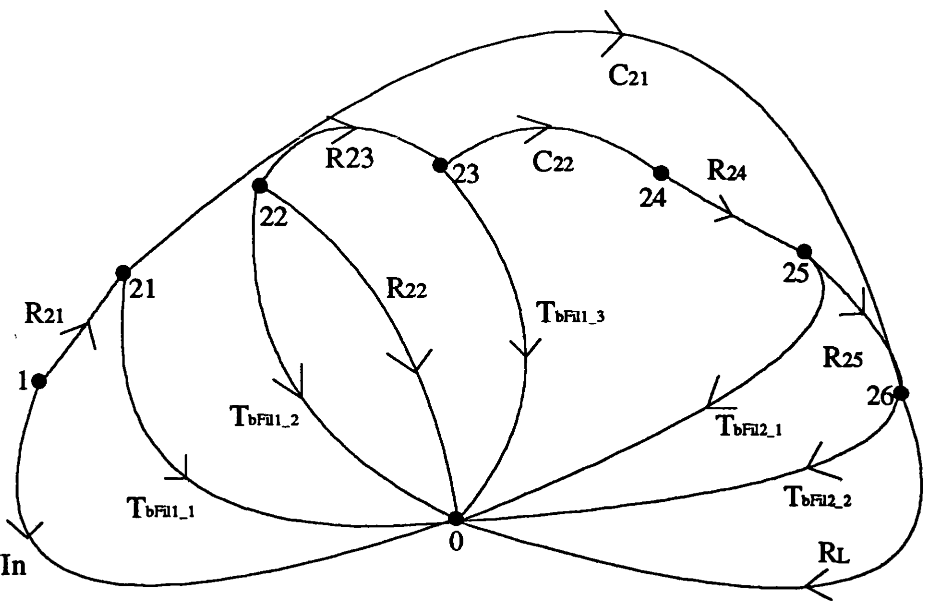


Figure 7.1.4-2 Graph Representation of the band pass filter

These matrices result from the assumptions that both opamps have the same open loop gain, A_v, and they do not draw any leakage current at their input nodes. MICROCAP IV was initially used to obtain the required test point values for the filter circuit under fault free conditions but the simulation results were shown to be inaccurate by non-zero testee percentage errors. This inaccuracy was traced back to the voltage controlled source model MICROCAP IV used and hence, is inherent in this simulator. As an alternative, a simulator routine was written in the symbolic simulation package, MAPLE V, to obtain the values of the required test points which are listed in Appendix III.

Soft faults such as the usual component deviations, deviation on the open loop gain of and the presence of leakage currents in the opamps were simulated and tested with the following global settings:

Magnitude and Frequency of VIn: 1V and 50KHz	A_v: 1000
MAPLE Simulation Accuracy: 10 digits	

Test points selected: i_{TbFil2_2}, i_{R22}, v_{R23}, v_{R21} and v_{R24} Decision Threshold: 1%

The diagnosis results obtained are summarized in Table 7.1.4-1.

	Diagnosis Results	Number of Test Cycles needed	Faulty Component or Edges	Ambiguous Components
No Fault	Fault Free	30	None	None
R_{21}+10%	Fault Not Diagnosed[1]	Exhaust All	None	R_{21}
R_{22}+10%	Fault Diagnosed	20	R_{22}	None
R_{23}+10%	Fault Not Diagnosed[1]	Exhaust All	None	R_{23}
R_{24}+10%	Fault Diagnosed	19	R_{24}	None
R_{25}+10%	Fault Diagnosed	19	R_{25}	None
R_L+10%	Fault Diagnosed	19	R_L	None
C_{21}+10%	Fault Diagnosed	69	C_{21}	None
C_{22}+10%	Fault Diagnosed	20	C_{22}	None
Gain of T_{bFil1} +10%	Fault Diagnosed	31	T_{bFil1_3}	None
Gain of T_{bFil2} +10%	Fault Diagnosed	52	T_{bFil2_2}	None
Leakage current at the v^+ input of the opamp T_{bFil1}	Fault Not Diagnosed[2]	Exhaust All	None	R_{21} , T_{bFil1}
Leakage current at the v^- input of the opamp T_{bFil1}	Fault Not Diagnosed[2]	Exhaust All	None	R_{23} , T_{bFil1}
Leakage current at the v^- input of the opamp T_{bFil2}	Fault Diagnosed[3]	77	T_{bFil2_1}	None

[1]Fault would have been diagnosed if backtracking had been implemented in the diagnosis program
[2]Fault would not have been diagnosed even if backtracking had been implemented in the diagnosis program as the testee partitions including all the ambiguous components are untestable.
[3]A decision threshold of 0.05% was used

Table 7.1.4-1 Diagnosis Results for Soft Faults in the Filter Circuit

Table 7.1.4-1 has revealed that all but the diagnosis results for the component value deviation faults on R_{21} and R_{23} are correct. In both the non-diagnosis cases the FDA has exhausted all 143 possible test cycles (out of 1287 possible components partitions) without converging to a final tester/testee partition in which all testers have been proved fault free by previous test cycles. This is why R_{21} and R_{23} are tested ambiguous. However, there must be some components already tested fault free by all the testable test cycles. If the number of these fault free components is more than or equal to the size of the tester group, these fault free components will be used to backtrack on the test results for all the previous testable test cycles so that the test results of a previous testable test cycle will become reliable if all its testers are in the group of components tested fault free by all the testable test cycles. This backtracking capability will enable the deviation faults on R_{21} and R_{23} to be diagnosed if it is implemented into the FDA. Despite the lack of this ability, the FDA still diagnoses the gain deviation faults in the two opamps.

As for the leakage input current faults at the input nodes, v^+ and v^- of T_{bFil1} and v^- of T_{bFil2} , these were simulated by adding a 1Mohm resistor at each of the input nodes of the opamps in turn in the MAPLE simulator routine. The fault in T_{bFil2} is diagnosed with its input edge T_{bFil2_1} being faulty whereas the faults in T_{bFil1} are not diagnosed. In the former case the decision threshold has to be reduced to 0.05% to diagnose the fault because of the tiny value of the leakage current ([voltage at node 25]x10^{-6}). In the latter case two ambiguous components are diagnosed. One of the ambiguous components is the opamp T_{bFil1} and the other is R_{21} or R_{23} for the leakage current fault at nodes 21 and 22 respectively. These faults will not have been diagnosed even if backtracking is implemented in the diagnosis program as the testee partition including all the ambiguous components are untestable. Thus, the non-diagnosis of the leakage current faults in the opamp T_{bFil1} is due to the testability of the circuit rather than the inefficiency of the FDA.

The different number of required test cycles to diagnose the faults in Table 7.1.4-1 have again shown that the FDA will not always use the same set of component partitions to converge to diagnosis results for different faults. Concerning the issue of reduction in circuit complexity, the test circuit at the transistor level of description has 35 components and 61 graph edges with the CMOS opamp in [95] having 13 transistors which are represented as two edge hierarchical components. These large number of components and graph edges are reduced to 11 components and 14 graph edges at the opamp level of description. This means that the dimensions of the L_{11} matrix is reduced from 60x60 to 13x13. Thus, the saving in computing resources in doing the fault diagnosis at the hierarchical level is substantial.

This section has used a practical band pass filter circuit at the opamp level of description to test the efficiency of the FDA in diagnosing single parameter deviation faults which include opamp faults such as gain deviation and the presence of leakage currents, as well as the conventional component value deviation faults. It can be told from the diagnosis results that the FDA is efficient in diagnosing most soft faults in the test circuit provided the precision of the test points is high and there is no tolerance in all the good components. In the cases in which the FDA results in diagnosing a set of ambiguous components, this ambiguity set will usually be resolved if the backtracking enhancement is implemented into the FDA. However, even with the backtracking enhancement implemented, there will be cases of an unresolved ambiguity set due to the testability of the test circuit. These diagnosis results have again demonstrated that the FDA will not always use the same set of component partitions to converge to diagnosis results for different faults.

Moreover, the use of the opamp level of description for the test circuit is another rigid demonstration of reduced circuit complexity, which is a direct consequence of the use of the hierarchical approach. In addition, a practical circuit will not always result in its circuit component transfer matrix being block diagonal, hence the criterion for a testable component partition is the existence of the inverse of Q instead of $Q^{`}$ (Chapter 5.2) for a hierarchical circuit. This fact has been shown by examining the diagnosis results, which indicate that the existence of the inverse of $Q^{`}$ does not always guarantee the invertibility of Q, for the faults in the test circuit.

7.1.5. Parameter Deviations around the Decision Threshold

In the real world, there are always tolerances on the values of components. In-tolerance components are not considered as faulty but out-of-tolerance components are. It is therefore importance that the FDA diagnoses in-tolerance and out-of-tolerance components as fault free and faulty respectively with the decision threshold set to the value of the accepted tolerance.

This section uses test circuits two and three in the previous sections to investigate the efficiency of the FDA in diagnosing parameter deviation faults around the decision threshold of 1%. The diagnosis results are summarized in Tables 7.1.5-1 and 7.1.5-2.

Soft Faults	Diagnosis Results	Number of Test Cycles needed	Faulty Component or Edges
R_0+1.1%	Fault Diagnosed	7	R_0
R_0-0.1%	Fault Free	5	None
R_2+1.1%	Fault Free	5	None
R_2-0.9%	Fault Free	5	None
[1]R_{14}+0.21%	Fault Free	5	None
[1]R_{14}-0.19%	Fault Free	5	None
R_4+1.1%	Fault Diagnosed	3	R_4
R_4-0.9%	Fault Free	5	None

[1]A decision threshold of 0.2% is used instead of 1% because of the high fault tolerance of the hierarchical component T_{2_3}.

Table 7.1.5-1 Diagnosis Results for Parameter Deviations around the Decision Threshold for the Passive Network

Soft Faults	Diagnosis Results	Number of Test Cycles needed	Faulty Component or Edges
R_{25}+1.5%	Fault Diagnosed	30	R_{25}
R_{25}+1.1%	Fault Diagnosed	30	R_{25}
R_{25}-0.1%	Fault Free	30	None
C_{21}+1.1%	Fault Diagnosed	69	C_{21}
C_{21}-0.9%	Fault Free	30	None
Gain of T_{bFil1} +1.1%	Fault Diagnosed	31	T_{bFil1_3}
Gain of T_{bFil1} -0.9%	Misleading	30	R_{25}

Table 7.1.5-2 Diagnosis Results for Parameter Deviations around the Decision Threshold for the Band Pass Filter

It is clear from the diagnosis results in the above tables that the FDA will diagnose faults with parameter deviations just larger than the decision threshold if the fault is in a non-hierarchical component such as R_0, R_4, R_{25} and C_{21}. However, the FDA may not diagnose faults with parameter deviations just larger than the decision threshold if the fault is in a hierarchical component. This is because the diagnosability of the fault depends on how it manifests itself on the constituent edges of the hierarchical component. In the case of T_4, a 1.1% deviation on the component value of R_2 results in the maximum percentage error of its edges being just below the 1% threshold. In the case of T_{2_3}, a 0.21% deviation on the component value of R_{14} results in the maximum percentage error of its edges being nearly 10 times smaller than the 0.2%

threshold due to the high fault tolerance of T_{2_3}. These are the reasons why these two faults are not diagnosed.

In addition, the FDA diagnoses the test circuits as fault free for all but one case of the soft faults which have parameter deviations below the decision threshold. There are two scenarios for the FDA to give fault free diagnosis results. In scenario 1, the edges affected by the below threshold deviation fault are in the testee partition in the last test cycle with all testers being proved good by previous test cycles. As the error percentages on these affected edges are all less than the decision threshold, the FDA does not diagnose any fault in the test circuit. Examples are the below threshold deviation faults in R_{25} and C_{21}. The rest of the below threshold deviation faults are examples of scenario 2, which has the edges affected by the below threshold deviation fault in the tester partition in the last test cycle. The percentage errors of the testee partition in this last reliable test cycle would be non-zero due to the presence of these affected edges in the tester partition. If these non-zero percentage errors are all less than the decision threshold, the FDA will again not diagnose any fault in the test circuit.

The change in gain of the opamp T_{bFil1} will only affect its output voltage which is represented as the output edge T_{bFil1_3}. Therefore, the gain change fault will manifest itself as an above zero percentage error for T_{bFil1_3} whenever it is in a testee partition. The faulty edge is in the testee partition in test cycle 17 and was tested good after this test cycle. Then, this faulty edge is in the tester partition of test cycles 26 and 30. In test cycle 26, the testee percentage errors deviates slightly from zero: iR_{21}: 0.0009% vC_{21}: 0.0012% iR_{24}: 0.0012% vR_{22}:0.0018% iR_{23}: 0.0000%. These small deviations are typical as the voltages and currents of the circuit components are not sensitive to a slight change in the gain of the op-amp. However, the deviations in test cycle 30 on the testees vR_{25}, VC_{21} and iR_{21} are 1.4709%, 0.9028%,0.0574% respectively. These values are unreasonably large given the insensitivity of the circuit parameters to the slight change in the gain of the opamp. These are not due to simulation precision as the same result is obtained from the MAPLE simulation with an increased accuracy of 15 digits on the test point values. As a result, R_{25} is mis-diagnosed as faulty.

This section has aimed to see if the FDA gives satisfactory diagnosis results for parameter deviation faults around the accepted tolerance value. Satisfactory diagnosis results mean that just out-of-tolerance components are diagnosed as faulty and just in-tolerance components are diagnosed as fault free. This statement has been found true for non-hierarchical components but it is not always true for hierarchical components. However, if the decision threshold is set to a value smaller than the accepted tolerance value and the FDA is modified so that it uses the accepted tolerance value instead of the decision threshold to decide which testee edges are faulty in the last reliable test cycle, the non-diagnosis on the two just out-of-tolerance components, T_4 (R_2+1.1%)and T_{2_3} (R_{14}+0.21%), and the misleading diagnosis caused by the gain-0.9% fault on T_{bFil1}, will be avoided. Of course, different components have different values of accepted tolerances in practical circuits. Thus, the future practical version of the FDA should allow the user to input local decision thresholds and accepted tolerance values. The use of local decision thresholds has already been implemented in the current version of the diagnosis program.

7.1.6. Effect of Accuracy of Test Point Values

All the diagnosis results reported in the previous sections have been based on high accuracy test point values which were simulated with either MICROCAP IV or MAPLE V with the precision set at 9 and 10 decimal points respectively. It is hard to have the test points values close to this precision in practical measurements. Therefore, the aim of this section is to see whether the FDA still gives reliable diagnosis results with reduced accuracy test point values. To achieve this aim, the diagnosis program was applied to test circuits 2 and 3 with three soft faults which have been correctly diagnosed with high accuracy test point values. The diagnosis results for the soft faults in test circuit 2 were obtained with the test points values rounded up to 5 and 3 significant figures, and are summarized in Tables 7.1.6-1 and 7.1.6-2 respectively. Table 7.1.6-3 summarizes the diagnosis results for the soft faults in test circuit 3 obtained with the test point values rounded up to 5 decimal points.

Soft Faults	Diagnosis Results	Number of Test Cycles needed	Faulty Component or Edges	Are the diagnosis results the same as the high accuracy case?
R_2+5%	Correct	5	T_{4_3}, T_{4_4}	Yes
R_{14}+5%	Correct	4	$T_{2\_3\_1}$, $T_{2\_3\_2}$	Yes
R_0+5%	Correct	7	R_0	Yes

Table 7.1.6-1 Diagnosis Results for Soft Faults at Reduced Simulation Accuracy (5 significant figures) for the Passive Network

Soft Faults	Diagnosis Results	Number of Test Cycles needed	Faulty Component or Edges	Are the diagnosis results the same as the high accuracy case?
R_2+5%	Correct	5	T_{4_3}, T_{4_4}	Yes
R_{14}+5%	Misleading	5	R_L, $T_{2\_3\_1}$[1]	No
R_0+5%	Misleading	5	R_L, $T_{2\_3\_1}$[1], T_{4_2}, T_{4_3}	No

[1]This component is diagnosed faulty in the last test cycle but is diagnosed good in previous test cycles

Table 7.1.6-2 Diagnosis Results for Soft Faults at Reduced Simulation Accuracy (3 significant figures) for the Passive Network

Soft Faults	Diagnosis Results	Number of Test Cycles needed	Faulty Component or Edges
R_{25}+10%	Misleading	19	R_{25}, R_L
R_{24}+10%	Correct	19	R_{24}
C_{21}+10%	Misleading	52	R_L

Table 7.1.6-3 Diagnosis Results for Soft Faults at Reduced Simulation Accuracy (5 digits) for the Band Pass Filter

It is clear from the diagnosis results in the above tables that the reliability of the diagnosis results of the FDA is extremely sensitive to precision in the test point values. Considering the diagnosis results for the soft faults in the passive network, the FDA gives reliable diagnosis results for a test point precision of 5 significant figures but the results turn unreliable when the accuracy of the test point values reduces further to 3 significant figures. Whereas the diagnosis results for the soft faults in the band pass filter become unreliable even with the precision of the test point values at 5 decimal places. These results altogether have also indicated that the minimum precision of the test point values required for the FDA to yield reliable diagnosis results is circuit specific. It is also noticed from the diagnosis results for the band pass filter circuit that resistors with large resistance value in general are more sensitive to a change in precision in the test point values as even small variations in current cause a large change in the voltage across them (e.g. R_L). This is why R_L is mis-diagnosed as faulty.

A possible approach to reduce the dependency of reliable diagnosis results of the FDA on high precision test point values is to apply some sort of normalization on the testee percentage errors before they are compared with the decision threshold to yield the pass/fail test results in a test cycle.

7.2. *Double Fault Diagnosis*

In the previous section, fault diagnosis with the FDA under a non-ideal but practical situation, moderate precision test point values, has been briefly investigated. Another non-ideal but also practical situation is to diagnose out-of-tolerance components as faulty in the presence of in-tolerance components. This section looks at this practical situation by narrowing down the investigation to the diagnosis of an out-of-tolerance component in the presence of an in-tolerance component. In addition, the out-of-tolerance and in-tolerance components can be either inside a hierarchical component or distinct components at the same hierarchical level. The passive network and band pass filter circuits have again been used as test circuits for this study and the diagnosis results for these double faults are summarized in Tables 7.2-1, 7.2-2 and 7.2-3.

Soft Faults	Decision Threshold/%	Diagnosis Depth	Diagnosis Results	Number of Test Cycles needed	Faulty Component or Edges
R_2+10%, R_8+1%	5	1	Correct	5	T_{4_3}
R_2+10%, R_8+1%	5	2	Correct	8	T_{4_3}
R_2+10%, R_8+1%	0.5	1	Correct	5	T_{4_2}, T_{4_3}, T_{4_4}
R_2+10%, R_8+1%	0.5	2	Correct	27	T_{4_2}, T_{4_3}, T_{4_4}

Table 7.2-1 Diagnosis Results for Double Faults within a Hierarchical Component for the Passive Network

The diagnosis results in Table 7.2-1 above have provided solid proof for the implication of fault manifestation in a hierarchical component. As has already been suggested in the conclusions of Chapter 7.1.3, the FDA cannot distinguish multiple faults within a hierarchical component and it can only diagnose the hierarchical component as faulty. In the example shown by the diagnosis results in Table 7.2-1, parameter deviation faults in the component values of R_2 and R_8 result in T_4 being diagnosed faulty. The differences in the diagnosis results are more edges of T_4 being diagnosed faulty as the decision threshold reduces from 5% to 0.5%.

Soft Faults	Decision Threshold/%	Diagnosis Depth	Diagnosis Results	Number of Test Cycles needed	Faulty Component or Edges
R_4+10%, R_6+1%	5	1	Fault Free	5	None
R_4+10%, R_6+1%	5	2	Misleading	7	R_5, R_4
R_4+10%, R_6+1%	0.5	2	Correct	6	R_4, R_6

Table 7.2-2 Diagnosis Results for Double Faults at the Hierarchical Level for the Passive Network

Soft Faults: R_{24}+10%, C_{22}+1%					
Decision Threshold/%	Diagnosis Depth	Diagnosis Results	Number of Test Cycles needed	Faulty Component or Edges	Ambiguous Components
5	1	Misleading	30	R_{25}, R_{21}[1], C_{21}[1]	None
5	2	Misleading	63	R_{24}, T_{bFil2_1}	None
0.5	2	[2]Incomplete	Exhaust All	None	R_{24},C_{22}, T_{bFil2}

[1]This component is diagnosed faulty in the last test cycle but are diagnosed good in previous test cycles.
[2]Backtracking is not possible as all the testee partitions containing the ambiguity set are not testable.
Table 7.2-3 Diagnosis Results for Double Faults at the Hierarchical Level for the Band Pass Filter

The diagnosis results in Tables 7.2-2 and 7.2-3 have shown that the FDA will not be able to diagnose distinct double faults if the decision threshold is set at the accepted tolerance value of 5%. These results are expected as the FDA cannot diagnose faults with parameter deviations smaller than the decision threshold. In addition, the diagnosis results in this case become misleading due to this "tolerance masking" effect on the FDA. However, intuition suggests that faults with parameter deviations larger than the decision threshold are diagnosable with the FDA. This has been shown by the correct diagnosis of double faults in the passive network with a decision threshold of 0.5%. To meet the requirement for the practical situation to diagnose the in-tolerance component R_6 as fault free and the out-of-tolerance component R_4 as faulty, the modification on the FDA suggested in the conclusions of Chapter 7.1.5 is needed.

The non-diagnosis of the distinct double faults in the band pass filter with a decision threshold of 0.5% is due to the testability of the circuit, which is also the cause for the non-diagnosis of the leakage current faults in the opamps, instead of the fault masking effect on the FDA.

This section has confirmed that double parameter deviation faults in a hierarchical component are not distinguishable but they are detectable by the FDA as a single faulty hierarchical component. It is thus sensible to assume that multiple faults in a hierarchical component are also detectable by the FDA as a single faulty hierarchical component. On the other hand, distinct double parameter deviation faults at the hierarchical level will not be diagnosable by the FDA if the decision threshold is larger than any one of the parameter deviations. However, when the threshold value is less than both the parameter deviations, the double faults have been shown to be diagnosable by the FDA. Whether or not double faults will be diagnosable by the FDA provided the threshold value is less than both the parameter deviations needs further investigation with more test circuits. For practical fault diagnosis, the diagnosis results of the FDA will need to be unaffected by the tolerance masking effect (i.e. the effect of parameter deviations smaller than the decision threshold), as it is not certain that information on minimum parameter deviations can always be made available from tolerance modelling of hierarchical components for choosing an appropriate decision threshold value. Even if this information is available, the FDA will not converge because all practical components have tolerances and hence no fault free testee component will be found for component re-partition at the end of a test cycle.

7.3. Conclusions

This section has, through the evaluation of the diagnosis results yielded by the FDA for different faults in several test circuits, benchmarked its diagnosis effectiveness and identified room for further improvements on the FDA. In addition, these evaluations have highlighted some typical features of the FDA and two salient characteristics of a hierarchical component, as well as demonstrating the advantages on adopting the hierarchical approach. Moreover, the issue of choosing an efficient circuit realization from different available hierarchical realizations of a CUT for fault diagnosis is also discussed.

The benchmarking exercise has been started by using the first test circuit to confirm that the FDA does not have any inherent numerical errors (Chapter 7.1.1) in its pseudo-circuit and tableau equation solvers. Then, under the assumption of prevailing ideal situations, which are high precision test point values and zero tolerance in all fault free components, single parameter deviation and catastrophic faults have been proved to be, on the whole, diagnosable with the FDA. There is cast iron proof from the diagnosis results of the first test circuit and the test circuits of the student project, that single soft faults in non-hierarchical circuits are diagnosable with the FDA. The diagnosis results of the student project, have also reminded us of the incapability of the FDA to diagnose open circuit faults in a flat circuit. For hierarchical circuits, the diagnosis results of the passive network and band pass filter circuits have shown that the FDA is able to diagnose most single soft faults except in a few cases, which are either diagnosable with the backtracking enhancement (Chapter 7.1.4) or non-diagnosable because of the testability of the CUT. Testability of a circuit, in terms of the ratio of the number of testable test cycles to the number of possible test cycles, has been suggested by the diagnosis results of the student project, to deteriorate with the level of hierarchical circuit abstraction. However, this deterioration of circuit testability as the circuit abstraction changes from flat to hierarchical is the inevitable consequence of the reduced circuit complexity, which has been convincingly demonstrated by the band pass filter circuit. Apart from the reduction in circuit complexity, which leads to saving in computing resources and is one of the major advantages of adopting the hierarchical approach, the other main advantage this approach offers is the diagnosis of hard faults entirely embedded in hierarchical components. This point has been shown by the negative proof the diagnosis results of the passive network circuit provided: all the hard faults in the second test circuit are not completely buried in the two hierarchical components. If these faults, in particular the short circuit faults, affect the circuit topology at the hierarchical level, they will not always be diagnosable with the FDA. However, hard faults which are not completely embedded in a hierarchical component are not necessarily topology-changing. This is why all the open circuit faults in the passive network circuit are diagnosable despite the fact that they are not totally enclosed by the hierarchical components. Open circuit faults are, by their very nature, less liable to make changes in the circuit topology than short circuit faults.

It is essential to have satisfactory diagnosis results with the FDA in practical testing of integrated circuits. Satisfactory diagnosis results mean that in-tolerance and out-of-tolerance components are diagnosed as fault free and faulty respectively. In order to let the FDA distinguish between out-of-tolerance and in-tolerance components, common sense suggests that the decision threshold is set to the accepted tolerance value. However, the diagnosis results for single soft faults with parameter deviations around the decision threshold in the second and third test circuits have proved that this intuitive statement is only true for faults not in a hierarchical component. To enable the FDA to give satisfactory diagnosis results for single parameter deviation faults in a hierarchical component around the accepted tolerance value, the decision threshold must be set to a value smaller than the accepted tolerance value and the FDA must be modified according to the suggestion made in the conclusions of Chapter 7.1.5.

Having examined the effectiveness of the FDA under the ideal situations, this section has re-focused its investigation on the effectiveness of the FDA under non-ideal situations in which precision of the test point values in practical measurements is only moderate and there are always tolerances in fault free components. The effect of the former non-ideal situation on the diagnosis results of the FDA has been investigated by re-running the diagnosis program on a few single soft faults in the second and third test circuits with the accuracy of the test point values reduced to various degrees. This investigation has found that the reliability of the diagnosis results of the FDA is extremely sensitive to the precision in the test point values, and the minimum precision of the test point values required for the FDA to yield reliable diagnosis results is also circuit specific. This dependency of reliable diagnosis results of the FDA on high precision test point values can be reduced by applying some kind of normalization on the testee percentage errors before they are compared with the decision threshold to yield the pass/fail test results in a test cycle.

As for the latter non-ideal situation, its implication on the FDA is to diagnose out-of - tolerance components as faulty in the presence of in-tolerance components. The effect of this practical situation on the diagnosis results of the FDA has been looked at by narrowing down the investigation to the diagnosis of an out-of-tolerance component in the presence of an in-tolerance component. Double parameter deviation faults, which correspond to the in-tolerance and out-of-tolerance components, in the second and third test circuits were simulated and their diagnosis results have led to two conclusions according to whether the double soft faults are in a hierarchical component, or are distinct faults at the same hierarchical level. In the former case, the double soft faults manifest themselves as a faulty hierarchical component and therefore they are detectable but not distinguishable by the FDA. Whereas in the latter case, the FDA will not be able to diagnose the double soft faults if the decision threshold is larger than any one of the parameter deviations associated with the faults. However, it has been shown in one case that the double soft faults are diagnosable by the FDA if the decision threshold is less than both the parameter deviations. Although this has implied that double soft faults are diagnosable by the FDA if the decision threshold is less than both the parameter deviations, this implication requires justification by the diagnosis results of more test circuits. Moreover, the FDA will need to be made insensitive to the tolerance masking effect (i.e. the effect of parameter deviations smaller than the decision threshold), as well as to changes in the precision of test point values, for it to be useful in practical fault diagnosis, as it is not certain that information on minimum parameter deviations can always be made available from tolerance modelling of hierarchical components for the selection of an appropriate decision threshold value. Even if this information is available, the FDA will not converge because all practical components have tolerances and hence no good testee component will be found for the re-partition of the circuit components at the end of a test cycle.

Apart from assessing the effectiveness of the FDA, the issue of choosing a suitable circuit realization from different available hierarchical realizations of a circuit for fault diagnosis has also been raised. The suitability of the circuit realization chosen is based on considering the diagnosis efficiency of all the available circuit realizations in terms of the required number of test cycles for the FDA to converge and the number of selected test points for each realization under fault free conditions.

The evaluations on the diagnosis results reported in this section have also identified some typical features of the FDA and revealed two salient characteristics of a hierarchical component. These typical features are: The FDA will not always use the same set of component partitions to converge to diagnosis results for different faults and there are always untestable test cycles in each diagnosis of a fault with the FDA. This last feature has justified the crucial capability of the FDA to enable its convergence by arbitrarily generating a component partition which does not repeat those partitions already used in previous test cycles. In addition, the fact that a practical circuit, such as the band pass filter in Chapter 7.1.4, will not always result in its circuit component transfer matrix being block diagonal, which consequence has been observed in the diagnosis results of the filter as the existence of the inverse of **Q**\` does not always guarantee the invertibility of **Q**, has proved the need for the FDA to use the existence of the inverse of **Q** instead of **Q**\` as the criterion for a testable component partition of a hierarchical circuit. As for the salient characteristics of a hierarchical component, these are fault tolerance which affects the value of the decision threshold, and fault manifestation which means a faulty hierarchical component can manifest the fault as any number of its constituent edges being faulty. The former characteristic implies that a high fault tolerance hierarchical component will need a low decision threshold for the diagnosis of any fault in it, whereas the implication of the latter characteristic is that multiple faults within a hierarchical component are detectable but not distinguishable by the FDA.

8. Conclusions and Further Work

This thesis has proposed fault diagnosis algorithms (FDAs), which are based on the combined formulation of the Self Test (ST) Algorithm[75][76] and Component Connection Model (CCM)[77], for linear analogue circuits after explaining test-related issues and reviewing the state-of-the-art and development of testing for both analogue and digital logic circuits. Before introducing this combined formulation in Chapter 5, it is necessary to let the reader have an understanding on how circuit topology is represented as a directed linear graph[79] whose many possible partitions of tree and cotree edges are described by matrix equations termed connection equations. This is the aim of Chapter 2 which has also demonstrated the derivation of the connection equation of an example circuit with two alternative methods and a set of tree edge selection rules.

Chapter 3 has addressed the problem of choosing an appropriate tree/cotree edge partition from the many available combinations so that the corresponding connection equation has high sparsity matrices optimized for efficient computation. These optimized connection matrices and their corresponding circuit tree are derived with the Optimal Tree Generation (OTG) Procedure[78] which is the subject of Chapter 3.

The Test Point Selection Scheme[78] which is based on the Topology and Diagnosability Rules together with either iterative test point compaction or the Supplementary Rule, is discussed in Chapter 4 which has also shown how the test point matrices in the measurement equation are constructed from the selected test points.

Chapter 5 has built on the topics discussed in the previous chapters to propose a Basic FDA for non-hierarchical circuits. The understanding of this algorithm requires the discussion of various related issues encompassing the ST Algorithm with the CCM formulation, the necessary condition for the existence of the Pseudo Circuit[75], the partition of circuit components, inter-matrix and inter-vector row exchanges, and decision algorithms. In addition, an analytical expression for the output of the Pseudo Circuit in terms of its input, the connection and test point matrices, and the tester partition of the circuit component transfer matrix has been derived.

Fault diagnosis of large size circuits with the basic FDA is impractical due to the direct proportionality relationship between circuit complexity and the requirement on computing resources. Chapter 6 has integrated the hierarchical approach[73][74] with the basic FDA to circumvent this problem. With the adoption of the hierarchical approach, the circuit complexity as seen by the basic FDA is reduced to make fault diagnosis on large size circuits possible. The essence of the hierarchical fault diagnosis is the use of sets of external nodes and voltage/current pairs specific to the analogue circuit building blocks each set abstracts, so that the demand on computing resources necessary to diagnose faults in a large size circuit incorporating such abstractions is reduced to a manageable level. Each circuit block can be at any level of circuit description and can encapsulate other abstractions at lower levels. This encapsulation process goes on until the lowest discrete component level. For this reason, the circuit block is termed a hierarchical component. Chapter 6 has also discussed the changes needed to be made on the Basic FDA because of the inclusion of the hierarchical approach. Besides the efficient use of computing resources, the hierarchical FDA has also had the advantages of improved usability and enhanced efficiency due to the provision of multiple circuit testing trees.

Chapter 7 has, through the evaluation of the diagnosis results yielded by the Hierarchical FDA for different faults in several test circuits, proved the effectiveness of the Hierarchical FDA under ideal conditions, which are high precision test point values and zero tolerance in all fault free components. This evaluation has also revealed another major advantage of the Hierarchical FDA: the diagnosis of hard fault entirely embedded in hierarchical components. In addition, the algorithm will become more effective if the backtracking enhancement (Chapter 7.1.4) and the suggestion made in the conclusions of Chapter 7.1.5 are implemented. However, the FDA will need to be made insensitive to the tolerance masking effect (i.e. the effect of parameter deviations smaller than the decision threshold), as well as to changes in the precision of test point values, for it to be useful in practical fault diagnosis. The tolerance masking problem may be solvable with a combination of multiple test frequencies and statistical techniques making use of the correlation between the tolerances and process deviations of parameters. This is currently being investigated by Mr Eberhardt at Robert Bosch GmbH.

Apart from these two research areas identified in Chapter 7, the natural continuation to the author's work is to extend the approach in this thesis to deal with non-linear analogue circuits as well as mixed analogue and digital circuits. The approaches in [26], [27] and [28] are useful in these proposed further works. Another investigation area suggested by the MEng project student[92] is to combine the Hierarchical FDA with the Transient Response Technique (TRT)[23][35][93]. That is, the Hierarchical FDA is used to provide a set of optimal test points for a CUT and then the CUT is tested with the TRT on these test points. The last area for further work is to adapt the Hierarchical FDA for testing of high frequency circuits which are described by signal flow graph, where the nodes represent the incident and reflected power waves and the edges represent the transmission and reflection coefficients. This signal flow graph is similar to the circuit graph in a way that the edges in both graphs represent the parameters measured for a specific test. Therefore the Hierarchical FDA should be applicable to high frequency circuits provided a translation interface from the signal flow graph to the circuit graph description is available. Of course, fault models for high frequency effects such as mismatched transmission lines are also needed for the decision algorithm.

Although the Hierarchical FDA in its current state of development cannot be used as a practical test tool, it can be used as a design tool by providing designers with an optimal set of test points to improve the observability of a CUT at a user specified hierarchical level. This is indeed the way the C-language implementation of the Hierarchical FDA is being evaluated in Bosch at present.

The work done to date in this research work has achieved the aim set out at the beginning of the collaboration between Bosch and the University of Bath in that the diagnosis procedure clarifies reachable diagnosis levels by providing a set of test points for maximum fault coverage at the prescribed hierarchical level to improve circuit observability. Furthermore, it is possible that the Hierarchical FDA will be included in the environment of a commercial CAD package developed by OPMAXX Inc. in USA as a diagnostic tool. This likely commercial exploitation of the author's work is still in the discussion stage between Bosch and OPMAXX Inc. Besides this likely exploitation, Bosch has the intention to promote AMITY[71] and VIRTUS[72] with an interface to the test generation support provided by the author's work and [90] (Chapter 1.5), as a commercial CAD tool set in future.

APPENDICES

Appendix I: References

1 Peter R. Shepherd, "Integrated Circuit Design, Fabrication and Test", Macmillan Press Ltd, 1996.

2 E. A. Amerasekera, "Failure mechanisms in semiconductor devices", John Wiley & Sons, 1987.

3 R. L. Wadsack, "Fault Modeling and Logic Simulation of CMOS and MOS Integrated Circuits", Bell Syst. Tech. Jour., Vol. 57, pp. 1449-1474, 1978.

4 K. C. Y. Mei, "Bridging and Stuck-at Faults", IEEE Trans. Computers, Vol. C-23, pp. 720-727, 1974.

5 Editor: Ruey-Wen Liu, Introduction in "Selected Papers On Analog Fault Diagnosis", IEEE Press, 1987.

6 V. D. Agrawal, "When to Use Random Testing", IEEE Trans. Computers, Vol. C-27, pp. 1054-1055, 1978.

7 R. D. Eldred, "Test routines based on symbolic logical statements", J. ACM, 6(1), pp.33-36, January 1959.

8 J. P. Roth, "Diagnosis of automata failures: A calculus and a method", IBM Journal of Research and Development, Vol. 10, No. 4, pp.278-291, 1966.

9 J. P. Roth, W. G. Bouricius, P. R. Schneider, "Programmed Algorithms to Compute Tests and to Detect and Distinguish Between Failures in Logic Circuits", IEEE Trans. Electronic Computers, Vol. EC-16, pp567-580, 1967.

10 F. F. Sellers, M. Y. Hsiao, L. W. Bearnson, "Analysing Errors with Boolean Difference", IEEE Trans. Computers, Vol. C-17, pp. 676-683, 1968.

11 J. Jacob, N. N. Biswas, "GTBD Faults and Lower Bounds on Multiple Fault Coverage of Single Fault Test Sets", Proc. Int. Test Conf., Washington D. C., pp. 514-519, 1987.

12 P. Goel, "An implicit enumeration algorithm to generate tests for combinational logic circuits", IEEE Transactions on Computers, Vol. C-30, No. 3, pp.215-222, 1981.

13 H. Fujiwara, T. Shimono, " On the acceleration of test generation algorithms", IEEE Transactions on Computers, Vol. C-32, No. 12, pp. 1137-1144, 1983.

14 M. H. Schulz, E. Trischla, T. H. Sarfert, "SOCRATES: a highly efficient automatic test pattern generation system", IEEE Trans on Computer-Aided Design, Vol. 7, No.1, pp. 126-137, 1988.

15 V. D. Agrawal, S. C. Seth, "Test generation for VLSI chips", IEEE Computer Society Order Number 786, 1988.

16 "SCOPE^{TM} Logic Products, Application & Data Manual", pp. 542-547, Texas Instruments, 1994.

17 Mani Soma, "Challenges in Analogue and MIxed-Signal Fault Models", IEEE Circuit and Devices Magazine, USA, Vol. 12, No. 1, pp. 16-19, January 1996.

18 Brian A. A. Antao, "Trends in CAD of Analog ICs", IEEE Circuits and Devices Magazine, USA, Vol. 12, No. 5, pp. 31-41, Sept. 1996.

19 S. K. Sunter, "Cost/benefit analysis of the P1149.4 mixed-signal test bus", IEE Proc.-Circuits, Devices, Syst., Vol. 143, No. 6, December 1996.

20 Vishwani D. Agrawal, "Testing in a Mixed-Signal World", Proc. 9th Annual IEEE Int. ASIC Conf. and Exhibit., pp.241-244, Rochester, NY, USA, Sept. 1996.

21 N. C. Lee, "A Hierarchical Analog Test Bus Framework for Testing Mixed Signal Integrated Circuits and Printed Circuit Boards", Journal of Electronic Testing: Theory and Applications, Vol. 4, pp.361-368, November, 1993.

22 B. R. Wilkins, B. S. Suparjo, "A Structure for Interconnect Testing on Mixed-Signal Boards", Journal of Electronic Testing: Theory and Applications, Vol. 4, pp. 369-374, November 1993.

23 M. A. Al-Qutayri, P. R. Shepherd, A. Bertin, "Implementation of the Transient Response Measurement of Mixed Signal Circuits", Proceedings, 3rd European Test Conference, ECT 93, pp. 66-73, April 1993.

24 Y. K. Malaiya, R. Rajsuman, "Bridging faults and IDDQ testing", IEEE Computer Society Press, 1992.

25 I. M. Bell, S. J. Spinks, J. Machado da Silva, "Supply current test of analogue and mixed signal circuits", IEE Proc.-Circuits Devices Systems, Vol. 143, No. 6, pp. 399-407, December 1996.

26 David Reisig, Raymond DeCarlo, "A method of Analogue-Digital Multiple Fault Diagnosis", International Journal of Circuits, Theory & Applications, Vol. 15, pp.1-22, 1987.

27 Y. H. Chang, H. T. Sheu, "Unified relaxation -pseudocircuit approach for analogue-digital fault diagnosis", IEE Proc.-Circuits Devices Syst., Vol. 142, No. 4, pp. 236-246, August 1995.

28 H. T. Sheu, Y. H. Chang, "The Relaxation Pseudocircuit Method for Analogue Fault Diagnosis", International Journal of Circuit Theory and Applications, Vol. 24, pp. 201-221, 1996.

29 Feng Lin, Zheng Hui Lin, T. William Lin, "A Uniform Approach to Mixed-Signal Circuit Test", International Journal of Circuit Theory and Applications, Vol. 25, pp. 81-93, 1997.

30 Mani Soma, "Automatic test generation algorithms for analogue circuits", IEE Proc.-Circuits Devices Syst., Vol. 143, No. 6, December 1996.

31 A. Walker, P. K. Lala, "Fault Diagnosis In Analog Circuits Using Element Modulation", IEEE Design & Test of Computer, pp. 19-29, March 1992.

32 P. M. Lin and Y. S. Elcherif, "Analogue Circuits Fault Dictionary-New Approaches & Implementation", International Journal of Circuit Theory & Applications, Vol. 13, pp. 149-172, 1985.

33 John W. Bandler, Aly E. Salama, "Fault Diagnosis of Analog Circuits", Proceedings of the IEEE, Vol. 73, No. 8, pp. 1279-1325, August 1985.

34 P. Duhamel & J. C. Rault, "Automatic Test Generation Techniques for Analogue Circuits and Systerms": A review, IEEE Trans. On Circuits ad Systems, Vol. CAS-26, No.7, pp. 411-439, July 1979.

35 M. A. Al-Qutayri, P. R. Shepherd, "On the Testing of Mixed-mode Integrated Circuits", Journal of Semicustom ICs, Vol. 7, No.4, pp. 32-39, Elsevier Science Publishers Ltd., England, 1990.

36 N. Navid and A. N. Wilson, "A Theory and an Algorithm for Analog Circuit Fault Diagnosis", IEEE Trans. On Circuits and Systems, Vol. CAS-26, No.7, pp. 440-457, July 1979.

37 N. Sen and R. Saeks, "Fault Diagnosis for Linear systems via Multifrequency Measurements", IEEE Trans. On Circuits and Systems, Vol. CAS-26, No.7, pp. 457-465, July 1979.

38 T. N. Trick, W. Mayeda and A. A. Sakla, "Calculation of Parameter Values from Node Voltage Measurements", IEEE Trans. On Circuits and Systems, Vol. CAS-26, No.7, pp. 466-474 , July 1979.

39 Zheng F. Huang, Chen-Shang Liu, "Node-Fault Diagnosis and a Design of Testability", IEEE Trans. on Circuits & Systems, Vol. CAS-30, pp. 257-265, May 1983.

40 A. E. Salama, J. A. Starzyk, "A Unified Decomposition Approach for Fault Location in Large Analog Circuit", IEEE Trans. on Circuits & Systems, Vol. CAS-31, pp. 609-622, July 1984.

41 N. B. Hamida, B. Kaminska, "Analogue circuit testing based on sensitivity computation and new circuit modeling", Proc. IEEE Int. Test Conf., pp. 652-661, Baltimore, MD, USA, 1993.

42 K. Saab, D. Marche, B. Kaminska, N. B. Hamida, G. Quesnel, "LIMSoft: Automated tool for sensitivity analysis used for test vector generation", Proc. 1st IEEE International Mixed Signal Testing Workshop, Grenoble, France, 1995.

43 N. B. Hamida, K. Saab, D. Marche, B. Kaminska, , G. Quesnel, "LIMSoft: Automated tool for design and test integration", Proc. 2nd IEEE International Mixed Signal Testing Workshop, pp. 56-71, Quebec City, Canada, 1996.

44 B. W. Jervis, Y. Brandt, Y. Maidon, "Circuit multi-fault diagnosis and prediction error estimation using a committee of bayesian neural networks", IEE Colloquium on Testing Mixed Signal Circuits and Systems, pp. 7/1-7/7, Savoy Place, London, 23rd October 1997.

45 A. Materka, M. Strzelecki, "Parametric Testing of Mixed-Signal Circuits by ANN Processing of Transient Responses", Journal of Electronic Testing: Theory and Applications, Vol. 9, No.1/2, pp. 187-202, August/October 1996.

46 Y. Maidon, B. W. Jervis, N. Dutton, S. Lesage, "Diagnosis of multifaults in analogue circuits using multilayer perceptrons", IEE Proc.-Circuits Devices Syst. Vol. 144, No. 3, pp. 149-154, June 1997.

47 Kurosh. Madani, Amar Bengharbi, Véronique Amarger, "Neural fault diagnosis techniques for non linear analogue circuits", Proc. of the SPIE Int. Soc. for Optical Engineering (USA) Applications and Science of Artificial Neural Network III Orlando FL USA, pp. 491-502, April 1997.

48 Robert Spina, Shambhu Upadhyaya, "Linear Circuit Fault Diagnosis Using Neuromorphic Analyzers", IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, Vol. 44, No. 3, pp. 188-196, March 1997.

49 Heiner Herbst, Doris Schmitt-Landsiedel, Matthias Schöbinger, "From Roadmaps to Reality: The Challenges of Designing Tomorrow's Chips", IETE (Instn. Electron. & Telecommun. Eng.) Technical Review (India), Vol. 13, No. 6, pp. 345-349, November-December, 1996.

50 Kamran Eshraghian, "Invited Talk: Challenges in Future Technologies", Proc. 14 IEEE VLSI Test Symposium, p. xxviii, Princeton, NJ, USA, 28 April-1 May, 1996.

51 A. Chatterjee, Naveena Nagi, "Design for Testability and Built-In Self-Test of Mixed Signal Circuits: A Tutorial", Proc. 10th International Conference on VLSI Design, pp. 388-392, January 1997.

52 K. Baker, A. M. Richardson, A. P. Dorey, "Mixed signal test-techniques, applications and demands", IEE Proc.-Circuits Devices Syst., Vol. 143, No. 6, December 1996.

53 C. L. Wey, S. Krishnan, "Built-in self-test structures for analog circuit fault diagnosis with current test data", IEEE Transactions on Instrumentation and Measurement, Vol. 41, No. 4, pp. 535-539, August 1992.

54 C. L. Wey, "Built-in self-test structure for analog circuit fault diagnosis", IEEE Transactions on Instrumentation and Measurement, Vol. 39, No.3, pp. 517-521, June 1990.

55 C. L. Wey, "Alternative built-in self-test structures for analogue circuit fault diagnosis", Electronic Letters, Vol. 27, No. 18, pp. 1627-1628, August 1991.

56 A. H. Bratt, R. J. Harvey, A. P. Dorey, A. M. D. Richardson, "Design-For-Test Structure to Facilitate Test Vector Application with Low Performance Loss in Non-Test Mode", Electronic Letters, Vol. 29, No. 16, pp. 1438-1440, August 1993.

57 Diego Vázquez, José L. Huertas, Adoración Rueda, "Reducing the Impact of DFT on the Performance of Analog Integrated Circuits: Improved Sw-Opamp Design", Proc. 14th VLSI Test Symposium, pp. 42-47, Princeton, NJ, USA, 28 April-1 May 1996.

58 Diego Vázquez, Adoración Rueda, José L. Huertas, "Fully Differential Sw-Opamp for Testing Analogue Embedded Modules", Proc. 2nd IEEE International Mixed Signal Testing Workshop, pp. 204-209, Quebec City, Canada, May, 1996.

59 Mani Soma, "A design for testability methodology for active analogue filters", Proc. IEEE International Test Conference, pp.183-192, September 1990.

60 Mani Soma, V. Kolarik, "A design for test technique for switched capacitor filters", Proc. VLSI Test Symposium, pp. 42-47, April 1994.

61 M. F. Toner, G. W. Roberts, "A BIST SNR, gain tracking and frequency response test of a sigma-delta ADC", IEEE Transactions on Circuits and Systems II: Analogue and Digital Signal Processing, Vol. 42, pp. 1-15, January 1995.

62 K. Arabi, B. Kaminska, J. Rzeszut, "A new built-in self-test approach for digital-to-analog and analog-to-digital converters", Proc. International Conference on Computer-Aided Design, pp. 491-494, 1994.

63 E. Teraoka, et. al., "A built-in self-test for ADC and DAC in a single-chip speech CODEC", Proc. International Test Conference, pp. 791-796, 1993.

64 M. J. Ohletz, "Hybrid built-in self test (HBIST) for mixed analog/digital integrated circuits", Proc. European Test Conference, pp. 307-316, 1991.

65 M. Lubaszewski, S. Mir, L. Pulz, "A multifunctional test structure for analog BIST", Proc. 2nd IEEE International Mixed-Signal Test Workshop, pp. 239-244, Quebec City, Canada, May 1996.

66 N. Nagi, A. Chatterjee, J. A. Abraham, "A signature analyzer for analog and mixed signal circuits", Proc. International Conference on Computer Design, pp. 284-287, October 1994.

67 M. Renovell, F. Azais, Y. Bertrand, "Analog signature analyzer for analog circuits: BIST implementation", Proc. 2nd IEEE International Mixed-Signal Test Workshop, pp. 233-238, Quebec City, Canada, May 1996.

68 Karim Arabi, Bozena Kaminska, Stephen Sunter, "Testing Integrated Operational Amplifiers Based on Oscillation-Test Method", Proc. 2nd IEEE International Mixed-Signal Test Workshop, pp. 227-232, Quebec City, Canada, May 1996.

69 P. M. Dias, J. E. Franca, N. Paulino, "Oscillation Test Methodology for a Digitally - Programmable Switched-Current Biquad", Proc. 2nd IEEE International Mixed-Signal Test Workshop, pp. 221-226, Quebec City, Canada, May 1996.

70 Bernard Courtois, " Some Trends in CAD, Test and Fabrication of Circuits and Systems", Proc. Third IEEE International Workshop on the Economics of Design, Test and Manufacturing, pp. 1-8, Austin, Texas, USA, May 16-17, 1994.

71 European Community funded project, ESPRIT 21.261, "AMITY: Analogue/Mixed-Signal Sub-Micron Design Test Bench System".

72 German government funded project, SSE/P13, "VIRTUS: Virtuelle, Analoge und Mixed-Signal Testentwicklung und Verifikation".

73 C. K. Ho, P. R. Shepherd, W. Tenten & F. Eberhardt, "Hierarchical Approach to Analogue Fault Diagnosis", Proc. 3rd IEEE International Mixed-Signal Test Workshop, pp. 25-30, Seattle, USA, June 1997.

74 C. K. Ho, P. R. Shepherd, F. Eberhardt & W. Tenten, "Improvements to Circuit Diagnosis Through Hierarchical Modelling", IEE Colloquium for Testing Mixed Signal Circuits & Systems, 23rd October 1997, Savoy Place, London.

75 Chin-long Wey, "Design of Testability for Analogue Fault Diagnosis", International Journal of Circuit Theory and Applications, Vol. 15, pp. 123-142 (1987).

76 C. Wu, K. Nakajima, C. Wey & R.Saeks, "Analogue Fault Diagnosis with Failure Bounds", IEEE Transaction on Circuits & Systems, Vol. CAS-29, No. 5, pp. 277-284, May 1982.

77 R. A. DeCarlo, R. Saeks, "Interconnected Dynamical Systems", Marcel Dekker, New York, 1981.

78 C. K. Ho, P. R. Shepherd, W. Tenten & R. Kainer, "Improvements in Analogue Fault Diagnosis Techniques", Proc. 2nd IEEE International Mixed-Signal Test Workshop, pp. 81-97, Quebec City, Canada, May 1996.

79 W. Mayeda, "Graph Theory", John Wiley & Sons, 1972.

80 J. B. Grimbleby, "Computer-aided Analysis and Design of Electronic Networks", Pitman, 1990.

81 William H. Hayt, "Engineering Circuit Analysis", Fifth Edition, McGraw-Hill, 1993.

82 William T. Weeks, Alberto Joe Jimenez, "Algorithms for ASTAP-A Network -Analysis Program, IEEE Transactions on Circuit Theory, vol. CT-20, No.6, November 1973.

83 S. Louis Hakimi, Kazuo Nakajima, "On a theory of t-Fault Diagnosable Analogue Systems", IEEE Transactions on Circuits and Systems, vol. CAS-31, No. 11, November 1984.

84 C. Wey, R. Saeks, "On the Implementation of an Analogue ATPG: The Linear Case", IEEE Trans on Instrumentation and Measurement, Vol. 1M-34, No. 3, P. 442-449, Sept. 1985.

85 E. Flecha, R. DeCarlo, "The Non-linear Analogue Fault Diagnosis Scheme of Wu, Nakajima, Wey and Saeks in the Tableau Context", IEEE Transaction in Circuits & Systems, CAS-31, P. 828-830 (1984).

86 Erwin Kreyszig, "Advanced Engineering Mathematics", fifth edition, p.325, 1983, John Wiley & Sons.

87 William H. Press, "Numerical Recipes in C", Second Edition, 1995, Cambridge University Press.

88 C. Wey, R. Saeks, "On the Implementation of an Analog ATPG: The Nonlinear Case", IEEE Transactions on Instrumentation and Measurement, Vol. 37, No.2, June 1988.

89 M. A. Al-Qutayri, P. R. Shepherd, "Go/no-go testing of analogue macros", IEE Proceedings-G, Vol. 139, No.4, August 1992.

90 Friedemann Edberhardt, PhD Transfer Report on "On the Approach of Analogue and Mixed-Signal Fault Simulation For use in a Fault-Diagnostic Environment" (PhD title is likely to change to Analogue and Mixed Signal Fault Simulation for Use in a Fault Diagnostic Environment"), School of Electronic & Electrical Engineering, University of Bath, 1997.

91 G. Clayton, B. Newby, "Operational Amplifiers", 3rd Edition, Newnes Butterworth-Heinemann, 1992.

92 Graeme J. Anderson, BEng/MEng Project Report on "Practical Application of Hierarchical Fault Diagnosis", May 1997, School of Electronic & Electrical Engineering, University of Bath.

93 S. A. Evans, M. A. Al-Qutayri, P. R. Shepherd, "A Novel Technique for Testing Mixed-Signal ICs", Proceedings, 2nd European Test Conference-ECT 91, pp.301-306, April, 1991.

94 Adel S. Sedra, Kenneth C. Smith, "Microelectronic Circuit", Fourth Edition, Oxford University Press, 1998.

95 F.Eberhardt, W. Tenten, C. K. Ho & P. R. Shepherd, "A Structural Approach to Hierarchical Tolerance Modelling of Analogue CMOS Integrated Circuit", accepted for presentation in the 4th IEEE International Mixed Signal and Test Workshop, Hague, Holland, June, 1998.

96 M. Robson, G. Russell, "An Application of Analogue BIST to an Automotive Circuit", Colloquium on Testing Mixed Signal Circuits and Systems, Savoy Place, London, 23 October, 1997.

97 Editor: Wai-Kai Chen, "The Circuits and Filters Handbook", p.2388, Figure 77.5, CRC Handbook, IEEE Presss.

Appendix II: Input Netlists of Test Circuits

Note: the C language representation of 10^x is written as 1ex, i.e. 5000 is written as 5e3 or 50e2.

First Test Circuit

5
4

R_1	1	2	500
R_2	2	0	1000
R_3	2	3	500
R_4	3	0	500
V_0	1	0	5

Second Test Circuit (Passive Network)

Flat Netlist
18
8

V_0	1	0	5
R_0	1	2	100
R_1	2	3	1000
R_2	3	4	10000
R_3	4	5	1000
R_4	5	6	3.3e3
R_5	6	7	1000
R_6	2	7	4.7e3
R_7	2	0	1800
R_8	2	3	2.2e3
R_9	3	0	4700
R_{10}	4	0	3300
R_{11}	4	5	8200
R_{12}	5	0	6800
R_{13}	6	0	1800
R_{14}	6	7	2200
R_{15}	7	0	4700
R_L	7	0	1000

Netlist for Hierarchical Circuit 1

10				
8				
V_0	1	0	5	
R_0	1	2	100	
R_1	2	3	1000	
R_3	4	5	1000	
R_4	5	6	3.3e3	
R_5	6	7	1000	
R_6	2	7	4.7e3	
T_4	5	1	2	0
2				
0				
3				
4				
5				
T_{2_3}	3	1	2	0
6				
0				
7				
R_L	7	0	1000	

Netlist for Hierarchical Circuit 2

12				
8				
V_0	1	0	5	
R_0	1	2	100	
R_1	2	3	1000	
R_2	3	4	10000	
R_3	4	5	1000	
R_4	5	6	3.3e3	
R_5	6	7	1000	
R_6	2	7	4.7e3	
T_{2_1}	3	1	2	0
2				
0				
3				
T_{2_2}	3	1	2	0
4				
0				
5				
T_{2_3}	3	1	2	0
6				
0				
7				
R_L	7	0	1000	

Third Test Circuit (Band Pass Filter)

11						
8						
VIn	16	0	50e3	0	1	0
R_{21}	16	21	50e3			
T_{bFil1}	3	1	1	0		
21						
22						
23						
C_{21}	21	26	50e-12			
C_{22}	23	24	50e-12			
R_{24}	24	25	50e3			
R_{25}	25	26	50e3			
T_{bFil2}	3	1	1	0		
0						
25						
26						
R_{22}	22	0	50e3			
R_{23}	22	23	50e3			
R_L	26	0	1e6			

Appendix III: Test Point Values obtained from Simulations of Test Circuits

Simulated Test Point Values for the First Test Circuit

Single soft faults and fault free conditions

Fault Condition	iR_2/mA	vR_4/mA
Fault Free	2.5	1.25
R_1+10%	2.380952	1.190476
R_1-10%	2.631579	1.315789
R_1+20%	2.272727	1.136364
R_1-20%	2.777778	1.388889
R_1+40%	2.083333	1.041667
R_1-40%	3.125	1.5625
R_1+60%	1.923077	0.961538
R_1-60%	3.571429	1.785714
R_1+80%	1.785714	0.892857
R_1-80%	4.166667	2.083333
R_1+100%	1.666667	0.833333
R_2+10%	2.325581	1.27907
R_2-10%	2.702703	1.216216
R_2+20%	2.173913	1.304348
R_2-20%	2.941176	1.176471
R_2+40%	1.923077	1.346154
R_2-40%	3.571429	1.071429
R_3+10%	2.53012	1.204819
R_3-10%	2.467532	1.298701
R_3+20%	2.55814	1.162791
R_3-20%	2.432432	1.351351
R_3+40%	2.608696	1.086957
R_3-40%	2.352941	1.470588

Simulated Test Point Values for the Second Test Circuit

Flat circuit under fault free conditions

iR_1/mA	iR_5/mA	iR_3/mA	iR_4/mA	iR_L/mA
0.768728	-0.131481	0.105132	0.038474	0.544975

iR_7/mA	iR_{10}/mA	vR_{13}/V	vR_9/V	vR_6/V	vR_3/V
2.527878	0.195633	0.413493721	3.781451654	4.005205508	0.105132002

Hierarchical circuit 1 under fault free conditions

vT_{4_2}/V	vT_{4_3}/V	iT_{4_1}/mA	$iT_{2\_3\_2}$/mA	iT_{4_4}/mA	vR_6/V	vR_4/V
3.781452	0.64559	2.877301	0.175715	0.066658	4.005206	0.126964

$vT_{4\_2}=v_3$, $vT_{4\_3}=v_4$, $iT_{4\_1}=iR_0-iR_1-iR_6$, $iT_{2\_3\_2}=iR_6+iR_5-iR_L$, $iT_{4\_4}=iR_3-iR_4$

Hierarchical circuit 2 under fault free conditions

$vT_{2\_3\_1}$/V	$vT_{2\_1\_2}$/V	iR_4/mA	$iT_{2\_1\_1}$/mA	$iT_{2\_3\_2}$/mA	$iT_{2\_2\_1}$/mA	vR_6/V	vR_3/V
0.413494	3.781452	0.038474	2.877301	0.175715	0.208454	4.005206	0.105132

$vT_{2\_3\_1}=v_6$, $vT_{2\_1\_2}=v_3$, $iT_{2\_1\_1}=iR_0-iR_1-iR_6$, $iT_{2\_3\_2}=iR_6+iR_5-iR_L$, $iT_{2\_2\_1}=iR_L-iR_3$

Single soft faults in hierarchical circuit 1

Soft Faults	vT_{4_2}/V	vT_{4_3}/V	iT_{4_1}/mA	$iT_{2\_3\_2}$/mA	iT_{4_4}/mA	vR_6/V	vR_4/V
R_0+5%	3.764518	0.642699	2.864415	0.174929	0.06636	3.98727	0.126395
R_0-5%	3.798538	0.648507	2.890301	0.17651	0.066959	4.023303	0.127538
R_0+50%	3.618676	0.6178	2.753444	0.168152	0.063789	3.832799	0.121499
R_0-50%	3.959561	0.675997	3.012823	0.183992	0.069798	4.193853	0.132944
R_2+5%	3.789594	0.625055	2.874553	0.176098	0.065072	4.007523	0.114613
R_2-5%	3.772622	0.667857	2.880278	0.175302	0.068378	4.002693	0.140357
R_2+50%	3.841759	0.493502	2.856958	0.178546	0.05491	4.022366	0.035488
R_2-50%	3.639098	1.004591	2.925316	0.169036	0.09439	3.964698	0.342893
R_{14}+5%	3.781441	0.645296	2.877312	0.17362	0.066587	4.004707	0.127544
R_{14}-5%	3.781463	0.645907	2.877286	0.177979	0.066735	4.005743	0.126337
R_{14}+50%	3.781374	0.643385	2.877393	0.159979	0.066123	4.001465	0.131321
R_{14}-50%	3.781626	0.650528	2.877092	0.210973	0.067857	4.013586	0.117203

Parameter deviations around the decision threshold in hierarchical circuit 1

Test Points	R_0+1.1%	R_0-0.1%	R_2+1.1%	R_2-0.9%	R_{14}+0.21%
vT_{4_2}/V	3.777713194	3.781791874	3.783299088	3.77991546	3.781451202
vT_{4_3}/V	0.644951388	0.645647723	0.640930603	0.64946376	0.645576856
iT_{4_1}/mA	2.874455	2.877559	2.876677	2.877818	2.8773
$iT_{2\_3\_2}$/mA	0.175542	0.175732	0.175803	0.175644	0.175625
iT_{4_4}/mA	0.066592	0.066664	0.066298	0.066957	0.066655
vR_6/V	4.001245837	4.005565859	4.005731205	4.004768375	4.005183816
vR_4/V	0.126838395	0.126975338	0.124161644	0.129294084	0.126989179

Test Points	R_{14}-0.19%	R_4+1.1%	R_4-0.9%
vT_{4_2}/V	3.781452065	3.781473273	3.781433782
vT_{4_3}/V	0.645601237	0.64595616	0.645286629
iT_{4_1}/mA	2.877299	2.877291	2.877307
$iT_{2\_3\_2}$/mA	0.175799	0.175736	0.175699
iT_{4_4}/mA	0.066661	0.066747	0.066585
vR_6/V	4.00522519	4.005292982	4.005133191
vR_4/V	0.126940991	0.127631417	0.12641208

Single hard faults in hierarchical circuit 1

Hard Faults	$vT_{4\ 2}$/V	$vT_{4\ 3}$/V	$iT_{4\ 1}$/mA	$iT_{2\ 3\ 2}$/mA	$iT_{4\ 4}$/mA	vR_6/V	vR_4/V
R_2 O/C	3.990695	0.1179	2.806721	0.185535	0.025895	4.064747	-0.190426
R_2 S/C	2.891062	2.89106	3.177632	0.133933	0.240115	3.75184	1.477548
R_7 O/C	3.982809	0.679966	0.368076	0.185073	0.070208	4.218478	0.133725
R_7 S/C	0.000041552	7.094E-06	49.999482	0.000002	0.000001	4.4011E-05	1.395E-6
R_8 O/C	3.517939	0.607751	2.531924	0.176646	0.063755	4.014393	0.10389
R_8 S/C	4.529555	0.753011	3.857809	0.173075	0.074899	3.979124	0.192469
R_9 O/C	4.370339	0.732057	2.680655	0.177153	0.07361	4.061608	0.174732
R_9 S/C	5.970662E-06	0.090354	4.140022	0.166491	0.022019	3.643025	-0.179769
R_{10} O/C	3.814002	1.155112	2.867069	0.167382	0.10612	3.972619	0.43183
R_{10} S/C	3.740209	0.443495	2.890262	0.186276	0.016658	4.046494	-0.259317
R_{11} O/C	3.781777	0.65091	2.877182	0.175839	0.078686	4.005754	0.122609
R_{11} S/C	3.778256	0.593349	2.878458	0.174511	-0.051431	3.999818	0.169721
R_{12} O/C	3.788531	0.754793	2.875187	0.172427	-0.008107	3.991851	0.246317
R_{12} S/C	3.755565	0.24627	2.885026	0.187745	0.340048	4.054037	-0.30947
R_{13} O/C	3.788359	0.722939	2.877266	0.1458	0.085342	3.878846	-0.024354
R_{13} S/C	3.770287	0.520575	2.877354	0.224068	0.03646	4.209432	0.371529
R_{14} O/C	3.78117	0.637627	2.877635	0.118872	0.064724	3.991694	0.142701
R_{14} S/C	3.782371	0.671599	2.876204	0.361393	0.072974	4.049341	7.55593E-05
R_{15} O/C	3.783405	0.659423	2.877849	0.070193	0.069982	3.938276	0.10017
R_{15} S/C	3.765878	0.535298	2.872924	1.017054	0.040154	4.538839	0.340589

Double soft faults in hierarchical circuit 1

Test Points	R_4+10%, R_6+1%	R_2+10%, R_8+1%
$vT_{4\ 2}$/V	3.782154698	3.795219142
$vT_{4\ 3}$/V	0.648032686	0.605806106
$iT_{4\ 1}$/mA	2.877673	2.869525
$iT_{2\ 3\ 2}$/mA	0.174307	0.176458
$iT_{4\ 4}$/mA	0.067237	0.063585
vR_6/V	4.010987035	4.009729889
vR_4/V	0.134602144	0.103033031

Simulated Test Point Values for the Third Test Circuit

Single soft faults and fault free conditions

Test Points		R_{21}+10%	R_{22}+10%	R_{23}+10%	R_{24}+10%
iT_{bFil2_2}/A	Real	3.124843186e-5	3.308226362e-5	3.449774139e-5	3.238666811e-5
	Imag.	2.844744310e-6	0.6135950410e-5	4.341987182e-6	5.295698336e-6
iR_{22}/A	Real	1.396614576e-6	0.2601274725e-5	1.978350715e-6	2.911441618e-6
	Imag.	-1.296977575e-5	-1.285745361e-5	-1.384345138e-5	-1.37002367e-5
vR_{23}/V	Real	-6.983072880e-2	-0.1300637363	-0.1088092894	-0.1455720809
	Imag.	0.6484887875	0.6428726804	0.7613898260	0.685011835
vR_{21}/V	Real	0.9300296097	0.8566567562	0.9008747374	0.8541367749
	Imag.	0.6497857651	0.7085099812	0.6936261314	0.6863818587
vR_{24}/V	Real	0.6828326069	0.7595184411	0.7848109842	0.8016727016
	Imag.	-0.4271413829	-0.3826014959	-0.4538566120	-0.4416939984

Test Points		R_{25}+10%	R_L+10%	C_{21}+10%	C_{22}+10%
iT_{bFil2_2}/A	Real	3.368094172e-5	3.372302931e-5	0.00003293939479	3.368784940e-5
	Imag.	4.277097035e-6	0.5325899124e-5	0.4026172970e-5	3.557813002e-6
iR_{22}/A	Real	1.551372979e-6	0.2429680024e-5	0.1396613576e-5	2.080344640e-6
	Imag.	-1.365801312e-5	-1.401084177e-5	-1.296977575e-5	-1.339684153e-5
vR_{23}/V	Real	-0.07756864895	-0.1214840012	-0.06983072880	-0.1040172319
	Imag.	0.682900656	0.7005420885	0.6484887875	0.6698420765
vR_{21}/V	Real	0.9222762138	0.8782730308	0.9300296097	0.8957747336
	Imag.	0.6842664573	0.7019431727	0.6497857651	0.6711817607
vR_{24}/V	Real	0.7220953062	0.7727868511	0.6828326069	0.7511002422
	Imag.	-0.4459110008	-0.4167250898	-0.4271413829	-0.4698234073

Test Points		T_{bFil1} gain+10%	T_{bFil2} gain+10%	T_{bFil1_v+}, leakage current=1e-6*v_{21}	T_{bFil1_v-}, leakage current=1e-6*v_{22}
iT_{bFil2_2}/A	Real	3.379592051e-5	3.379451083e-5	0.00003336481920	3.341062999e-5
	Imag.	5.284632982e-6	0.5287103184e-5	0.6419988936e-5	0.575624761e-5
iR_{22}/A	Real	2.428434864e-6	0.2428377710e-5	0.2897264906e-5	0.2664649212e-5
	Imag.	-1.401282027e-5	-1.401086096e-5	-1.382501216e-5	-1.408614648e-5
vR_{23}/V	Real	-0.1214217432	-0.121488855	-0.1448632452	-0.1265708376
	Imag	0.7006410134	0.7005430480	0.6912506080	0.669091958
vR_{21}/V	Real	0.878357490	0.8783382767	0.8548470283	0.8665077361
	Imag.	0.7019149064	0.7019441341	0.6926331092	0.7056807233
vR_{24}/V	Real	0.7728353696	0.7727831991	0.7816101155	0.7657733467
	Imag.	-0.4168610289	-0.4167694308	-0.3869377577	-0.397988783

Test Points		T_{bFil2_v-}, leakage current=1e-6*v_{25}	Fault Free
iT_{bFil2_2}/A	Real	3.379246774e-5	0.00003379321248
	Imag.	5.288994032e-6	0.5288052874e-5
iR_{22}/A	Real	2.430143780e-6	0.2429680024e-5
	Imag.	-1.401099715e-5	-0.00001401084177
vR_{23}/V	Real	-0.1215071889	-0.1214840012
	Imag.	0.7005498575	0.7005420885
vR_{21}/V	Real	0.8782497967	0.8782730308
	Imag.	0.7019509572	0.7019431727
vR_{24}/V	Real	0.7728121151	0.7727868511
	Imag.	-0.4167084985	-0.4167250898

Parameter deviations around the decision threshold

Test Points		R_{25}+1.5%	R_{25}+1.1%	R_{25}-0.1%	C_{21}+1.1%
iT_{bFil2_2}/A	Real	3.378363742e-5	3.378645835e-5	0.00003379375267	3.370048339e-5
	Imag.	5.131278915e-6	0.5172916489e-5	0.5298565388e-5	0.5132679474e-5
iR_{22}/A	Real	2.291522084e-6	0.2328146102e-5	0.2438969650e-5	0.2300174398e-5
	Imag.	-1.396315117e-5	-1.397606644e-5	-1.401394836e-5	-1.389436873e-5
vR_{23}/V	Real	-0.1145761042	-0.1164073050	-0.1219484824	-0.1150087198
	Imag.	0.6981575585	0.6988033220	0.7006974180	0.6947184365
vR_{21}/V	Real	0.8851947436	0.8833598803	0.8778076206	0.8847612627
	Imag.	0.6995538736	0.7002009286	0.7020988128	0.6961078734
vR_{24}/V	Real	0.7651924072	0.7672188679	0.7732926334	0.7621914085
	Imag.	-0.4216143737	-0.4203287146	-0.416392515	-0.4185664834

Test Points		C_{21}-0.9%	T_{bFil1} gain+1.1%	T_{bFil1} gain-0.9%	T_{bFil1} gain-0.9% (15 digit precision)
iT_{bFil2_2}/A	Real	3.386860497e-5	3.379353659e-5	0.00003379294194	3.37929419424438e-5
	Imag.	5.418497900e-6	0.5287643648e-5	0.5288394438e-5	0.528839444790044e-5
iR_{22}/A	Real	2.538864790e-6	0.2429531036e-5	0.2429804372e-5	0.242980437624966e-5
	Imag.	-1.410630520e-5	-1.401107855e-5	-1.401064413e-5	-1.40106441333368e-5
vR_{23}/V	Real	-0.1269432395	-0.1214765519	-0.1214902186	-0.121490218812483
	Imag.	0.7053152600	0.7005539275	0.7005322063	0.700532206666838
vR_{21}/V	Real	0.8728028740	0.8782831385	0.8782645943	0.878264594065797
	Imag.	0.7067258905	0.7019397908	0.7019459952	0.701945995176765
vR_{24}/V	Real	0.7815870617	0.7727926591	0.7727820031	0.772782003386726
	Imag.	-0.4150683124	-0.4167413572	-0.4167115121	-0.416711511945312

Double faults

Test Points (Diagnosis Depth=1)		R_{24}+10%, C_{22}+1%
iT_{bFil2_2}/A	Real	0.00003238312399
	Imag.	0.5127510544e-5
iR_{22}/A	Real	0.2875198562e-5
	Imag.	-0.00001364204433
vR_{23}/V	Real	-0.1437599281
	Imag.	0.6821022165
vR_{21}/V	Real	0.8559525520
	Imag.	0.6834664209
vR_{24}/V	Real	0.7995847149
	Imag.	-0.4474498412

Test Points (Diagnosis Depth=2)		R_{24}+10%, C_{22}+1%
vC_{22}/V	Real	-0.5127910277
	Imag.	-0.9163482256
vR_{25}/V	Real	0.7268951953
	Imag.	-0.4067725828
iT_{bFil2_2}/A	Real	0.00003238312399
	Imag.	0.5127510544e-5
iR_{22}/A	Real	0.2875198562e-5
	Imag.	-0.00001364204433
vR_{23}/V	Real	-0.1437599281
	Imag.	0.6821022165
vR_{21}/V	Real	0.8559525520
	Imag.	0.6834664209

Appendix IV: User Manual for the Diagnosis Program

User Manual

For

Version 1 of the Diagnosis Program

Written by

C. K. Ho, Research Officer
School of Electronic & Electrical Engineering
University of Bath
Claverton Down, Bath BA2 7AY, UK
August, 1997

CONTENTS

1. Introduction

Welcome to version 1 of the diagnosis program which implements an improved Fault Diagnosis Algorithm (FDA) originated from Wey[1]. The original FDA has been modified and extended to deal with linear hierarchical components whose terminal voltages and currents can be described by matrix equations. Its practical usefulness is somewhat limited as fault masking effects due to tolerances & multiple faults are not considered in the mathematical analysis behind the program.

The program is best used to diagnose a single fault in a circuit as the FDA converges more rapidly in the single fault case than in the multiple fault case to give the diagnosis result. Although the program allows a user to do multiple fault diagnosis, it will not always succeed in diagnosing the faults because of the fault masking effects. The program will not always diagnose a faulty hierarchical component if the fault is due to more than one of the constituent elements of the hierarchical component being faulty. This is because of the fault masking effects within the hierarchical component. In addition, if a hierarchical component has a short circuit fault on the boundary between the other components in the CUT and itself, which changes the topology of the CUT, the diagnosis program may diagnose some good components as faulty as well as diagnosing the faulty hierarchical component.

As the program has been developed with a SUN SPARC5 workstation which runs under SunOS Release 4.1.3_U1 with 32Mbyte RAM, the hardware required for running the diagnosis program should be compatible with the aforementioned hardware platform. The limit on the size of the circuit the program can tackle is of course dependent of the available RAM memory of the workstation.

2. Starting the Diagnosis Program

The diagnosis program is normally started by typing

diagnosis1 inputNetlist

where ***inputNetlist*** is the name of the file containing the netlist of the circuit under test in the format specified in Section 4. It can also be run with an extra command line option below,

diagnosis1 inputNetlist option

with ***option*** being one of the following keywords:

trace, traceAll, openLoop, openLoop+trace, openLoop+traceAll

Starting the program with one of the above keywords will display the internal matrices & intermediate results of the program in different details and/or disable tester & testee re-partition from the results of a previous test cycle. This is explained as follows:

trace- Its effect is to enable screen display of the internal matrices & intermediate results during program execution.

traceAll-Its effect is the same as that of ***trace*** but the intermediate matrices from solving the Pseudo Circuit equation and the intermediate results for test point selection and searching of a testee partition are also displayed.

openLoop-Its effect is to disable tester & testee re-partitioning from the results of a previous test cycle.

openLoop+trace-It has the combined effects of ***trace*** and ***openLoop***.

openLoop+traceAll-It has the combined effects of ***traceAll*** and ***openLoop***.

3. User Interaction with the Diagnosis Program

This section presents you with the questions the diagnosis program asks a user to answer before the diagnosis procedure is applied to the Circuit Under Test (CUT), and explains these questions in order to help you to answer them correctly. These questions are listed in the same order as they will be displayed when the diagnosis program is executing.

Text printed in bold & italic type are the questions. If parts of a question are in italics, these parts will depend on the CUT or the state of program execution. If parts of a question are underlined and in italics, these parts will depend on the state of program execution only when the run option is openLoop, openLoop+trace or openLoop+traceAll. The string *** is used to indicate where your answer to the question will be entered.

1. ***List of hierarchical components & their corresponding indices:***

0	*T1_*
1	*T2_*
:	:
:	:
numHierComponent-1	:

The first column is the indices the diagnosis program gives to the hierarchical components. These indices are given according to descending order of the size (the number of constituent graph edges) of their corresponding hierarchical components. If two or more hierarchical components have the same size, their indices will depend on their order of appearance in the netlist of the CUT. For example, T1 will appear before T2 in the netlist if they have the same size. The entries in the second column without the _ are the names of the hierarchical components in the netlist file. The symbol _ is used with the name of a hierarchical component to describe its edges. For example, T1_1 is the first edge of T1, T1_2 is the second edge of T1, etc.

If test points selected are on the hierarchical components
How many hierarchical components do you want to have as
many voltage test points as possible: ***

Your could enter 0, 1, 2,...,or numHierComponent-1. When you enter 0, you have no preference for the hierarchical components regarding the types of test points (current or voltage test point) on the hierarchical components, should some of the test points selected by the diagnosis program be on the hierarchical components. If you enter any number between 1 and numHierComponent-1, you express a preference on some of the hierarchical components so that if some of the test points selected are on the preferred hierarchical components, there will be as many voltage test points as possible.

If you enter a number other than 0, you will be asked the following questions:

Please type the index of each hierarchical component you wanted to have
as many voltage test points as possible in descending order of preference
as you are prompted
or press `r` to redisplay the list of hierarchical components & their indices

index[0]: ***

index[1]: ***
:
:

2. ***Press `g` to instruct the diagnosis program to use a global percentage limit for the decision algorithm, Press any other key to instruct the program to use an individual percentage limit for each voltage/current pair:*** ***

   If you press 'g' and then the "RETURN" key, you will be asked the following question:

   Please enter a global percentage limit for the decision algorithm: ***

   If you press any key other than the 'g' key and then press the "RETURN" key, you will be asked the following questions:

Please enter a percentage limit for the voltage/current of edge R0: ***
Please enter a percentage limit for the voltage/current of edge R2: ***

:

:

:

The percentage limit or limits you enter will be used in the decision algorithm to decide whether a test on an edge voltage or current is a pass or fail. This test is to compare the input percentage limit corresponding to the edge $\mathbf{b}^2[i]$, with

$$\mathbf{b}^2[i] - \hat{\mathbf{b}}^2[i] \Big/ \hat{\mathbf{b}}^2[i]$$

where

$\mathbf{b}^2[i]$ is the ith element of the testee partition of the output variable vector in the connection equation of the CUT as calculated from the Pseudo Circuit or Tableau equation

$\hat{\mathbf{b}}^2[i]$ is the nominal value of $\mathbf{b}^2[i]$, which is the product of its corresponding row in the testee partition of the circuit component transfer matrix, $\mathbf{Z}^2$, and the testee partition of the input variable vector in the connection equation of the CUT as calculated from the Pseudo Circuit or Tableau equation.

The test is a pass if $\mathbf{b}^2[i] - \hat{\mathbf{b}}^2[i] \Big/ \hat{\mathbf{b}}^2[i] \leq$ specific percentage limit.

The value of the global percentage limit to use depends on the CUT and is based on experience. If zero or too large a value is used for the limit, the program will probably give the user misleading diagnosis results. In the former case most of the test results will be failed because $\mathbf{b}^2[i] - \hat{\mathbf{b}}^2[i] \Big/ \hat{\mathbf{b}}^2[i]$ is not always exactly 0 for a fault free CUT due to truncation errors in the values of the test points. In the latter case a failed test result will be considered as a pass as the limit is set too high.

If the diagnosis program is run once on the CUT with the test point values for the fault free case prior to running it on the same CUT with the test point values for the faulty case, the program display in each test cycle will give an indication on the minimum global percentage limit to use for testing the CUT. For example, the percentage errors for most testee edges in each test cycle are at most 0.0006 in the sample program display in Section 7 (page 24-27), the minimum global percentage limit to use will be 0.0006. A sensible range of values to use is 0.001 to 0.1. As for the maximum global percentage limit, this can be gauged by examining the error percentages in the display produced by running the diagnosis program on the faulty CUT. If the value of the global percentage limit you used to run the program is larger than any one of the error percentages in all the test cycles in the program display, you have to re-run the diagnosis program with a reduced global percentage limit to get reliable diagnosis results.

3. ***Choice of Testing Strategies:***
   Please enter 1 for using Node Voltages as test points
   2 for using Both Node Voltages & branch currents as test points
   Your Choice: ***

   Your choice is either 1 or 2. However, if you choose 1, the diagnosis program will most likely stop as it is very unlikely to have all voltage test points selected for most circuits.

4. ***Choice of Diagnosis Depth:***
   Number of maximum faults must be between 1 & 7
   Please enter no. of maximum faults: ***

   The upper bound on the number of maximum faults, 7 in this case, depends on the number of components less the number of independent sources in the CUT.

5. ***Test Point Inputs from Measurements or Simulations***
   for tree number 1
   Press `m` for manual input or `f` for input from a file: ***

The format of the file storing measurement or simulation results is described in Section 6. If you type `m` for manual input, you will be asked the following questions:

Input mode for complex quantities
Press `p` for polar or `r` for cartesian mode: ***

The following questions will need to be answered for all the test points selected.

Please enter the *voltage* ***for*** *T1_2*

If you type `r`, you have to give the real & imaginary parts of the test point.

Please enter real part: ***

Please enter imaginary part: ***

If you type `p`, you have to give the modulus & argument (in radian)of the test point.

Please enter the modulus: ***

Please enter the argument: ***

6. ***Maximum number of test cycles must not exceed*** *1716*
   This number is reduced substantially by the fact that the constituent elements of a hierarchical component must be partitioned in the same testee or tester group
   Please enter maximum number of test cycles: ***

   You will only be asked the above question if the run option is openLoop, openLoop+trace or openLoop+traceAll.

7. ***Test Cycle is*** *0* ***for tree number*** *1*
   Press `y` to manually input the tester partition or `n` to have the program choose the component partitions: ***

   The diagnosis program provides more than one optimal tree for testing a hierarchical circuit. If the diagnosis results of the CUT are incomplete with the first optimal tree which is labelled "tree number 1", the diagnosis program will continue the diagnosis on the CUT with the next available optimal tree until complete diagnosis results are obtained or all the available optimal trees are exhausted.

You will only be asked the above question before the beginning of the first test cycle for each circuit testing tree if you do not provide the diagnosis program with a run option or the run option is either trace or traceAll. If the run option is openLoop, openLoop+trace or openLoop+traceAll, you will be asked the above question every time before a test cycle commences.

If you type `y`, you will be asked the following questions:

```
You have chosen to enter the tester partition manually
Please enter the indices of the elements from the list below
as you are prompted to do

If you have hierarchical elements in the tester group
Please make sure all constituent elements of a hierarchical component are in the tester group

1                    R0
2                    R1
:                    :
:                    :
7                    T1_1
8                    T1_2
9                    T1_3
10                   T1_4
11                   T2_1
12                   T2_2
13                   RL
```

The first column is the graph edge indices the diagnosis program gives to the components in the CUT. These indices are given according to the order the components appear in the netlist file of the CUT. Hence, these indices may not be consecutive because of the presence of independent sources in the netlist. The second column is the names of the components in the netlist. The hierarchical components T1 & T2 have 4 & 2 edges respectively.

```
Start inputting indices for tester partition
Enter the index or `r` to redisplay the list of indices & names when prompted
tester[0]= ***

tester[1]= ***
:
```

:
tester[numTester-1]= ***

8. ***Press `y` to continue or `n` to quit the program***
***

You will only be asked the above question at the end of every test cycle if the run option is openLoop, openLoop+trace or openLoop+traceAll.

If there is no diagnosis result after all the test cycles of the first circuit testing tree are exhausted, questions 5 to 8 will be displayed again from circuit testing tree 2 onwards until the FDA converges to give the diagnosis result. If the FDA does not converge to give the diagnosis result after all the test cycles of all the available circuit testing trees are exhausted, the diagnosis program will display all those components which have been diagnosed good by the FDA.

4. File Format for the Netlist of the Circuit Under Test

The entry in the first line of the netlist is the number of circuit components in the CUT. Each component can be either two-terminal type or hierarchical type. A hierarchical component is a multi-port or multi-terminal component which can be a BJT, a MOST, an Op-Amp, passive components, a controlled source & its controlling element, or a combination of these. Thus the count for circuit components is less than or equal to the count for the constituent elements of the circuit components.

The entry in the second line is the number of nodes in the CUT. From the third line onwards are the entries for the components in the CUT. In the following discussion, <enter> refers to the carriage return character. The entry for a hierarchical component consists of more than one line whereas the entry for a passive component or source occupies only one line. The first line of entry for all components begins with a capital letter which indicates the type of the component with the acronym in the following table:

V	I	C	R	G	L	P	T
Voltage Source	Current Source	Capacitor	Resistor	Conductor	Inductor	Multi-port Component	Multi-terminal Component

The formats for the line entries of different components are explained below with the Backus Naur Form (BNF)[2]:

$\{\ \}_{1+}$: repeat the enclosed items one or more times, $[]_{R+}$: repeat the enclosed items R times.

|: to separate choices, $\{$letter|digit$\}_{1+}$: name for a component.

- The line of entries for a **passive component** or an **independent dc source** consists of four data fields separated by the <tab> character in the following format:

  [V|I|C|R|G|L]$_{1+}$\{letter|digit\}$_{1+}$<tab>in_node<tab>out_node<tab>value<enter>

The first data field is for component name, the second and third data fields are for in node and out node of the component respectively (a node is an in node if current flows from the node into its connecting edge. Otherwise, it is an out node.), and the fourth data field is for the value of the component. If a node is a dummy node, the last character in its field will be a 'D'.

- The line of entries for an **independent ac source** consists of seven data fields separated by the <tab> character in the following format:

  [V|I]$_{1+}${letter|digit}$_{1+}$<tab>in_node<tab>out_node<tab>signal frequency in Hz<tab>dc offset<tab>magnitude of ac signal<tab>phase of the ac signal in radian<enter>

- The line of entries for a **controlled source** consists of six data fields separated by the <tab> character in the following format:

  [V|I]$_{1+}$[v|i]$_{1+}${letter|digit}$_{1+}$<tab>in_node<tab>out_node<tab>in_node_ctrl_ele<tab> out_node_ctrl_ele<tab>proportional constant<enter>

  The second letter in the first data field is either a 'v' or 'i' to indicate either a voltage or current controlled source respectively. In the fourth and fifth data fields are the in node and out node of the controlling element respectively. If the dependent source is current controlled, either the in_node_ctrl_ele or out_node_ctrl_ele entry will be a dummy node. If the in_node_ctrl_ele entry is a dummy node, this dummy node will also be on the out_node entry field of the controlling element of the current controlled source. Similar argument applies for the case that the out_node_ctrl_ele entry is a dummy node. In the last data field is the proportional constant which multiplies the voltage or current of the controlling element to give the value of the controlled source.

  Although a controlled source and its controlling element can be included in the netlist as separate entities by using the above format, the use of the diagnosis program on an input netlist consisting of such a format is somewhat handicapped (see Section 9). It is best to include the controlled source and its controlling element as a hierarchical component in the netlist to utilise the full power of the diagnosis program.

- The first line of entries for a **multi-port component** consists of four data fields separated by the <tab> character in the following format:

  P{letter|digit}$_{1+}$<tab>Number of Ports<tab>Function Number<tab> ParameterSetNo<enter>

  Function number is a number assigned to a function which is specific for a component type. This number is used together with the entry in the first data field to form the name of the file which stores the component transfer matrix. For example, if the first and third data fields are P1 and funNum respectively, the diagnosis program assumes that the file P1_funNum contains the component transfer matrix for P1.

  ParameterSetNo is also a number assigned to a file which contains the relevant parameters for a component function. Components of the same type have the same function number which operates on different parameter sets pointed to by the number ParameterSetNo. The entry in this data field is not used by the current version of the diagnosis program. This data field is provided for future development of the program.

  The rest of the line entries for a multi-port component have the same formats which provide the diagnosis program with the in nodes and out nodes of the ports of the multi-port component. The format of such a line entry is :

  in_node<tab>out_node<enter>

- The first line of entries for a **multi-terminal component** consists of five data fields separated by the <tab> character in the following format:

  T{letter|digit}$_{1+}$<tab>Number of terminals<tab>FunctionNumber<tab>ParameterSetNo <tab> common_node<enter>

  The last data field is the entry for a common terminal which can be any node in the circuit and needs not be on the multi-terminal component. The graph representation of the multi-terminal component is a star like topology with the common terminal being the out node of every constituent edge. If the common terminal happens to be on the multi-terminal component, the diagnosis program will take this fact into account and reduce the number of terminals, and hence the number of constituent edges accordingly. The rest of the line entries are to provide the diagnosis program with the node numbers of the terminals on the multi-terminal component. The format of the line entry is:

  in_node<enter>

The diagnosis program can deal with a netlist file with non-sequential node numbers by mapping the non-sequential node numbers to sequential ones and creating a new netlist for the CUT with these mapped sequential node numbers. The name of the new netlist file is the name of the old netlist file appended with "**.newNetlist**". All subsequent operations of the program will then be on the new netlist file but any outputs relating to the node numbers will still refer to the non-sequential node numbers in the old netlist. Some of the output files generated (see Section 7) will use the name of the new netlist file instead of that of the old netlist file as part of the names. In doing the mapping of non-sequential to sequential node numbers, the diagnosis program maps the minimum non-sequential node number to node number 0, which is assumed to be the reference node of the CUT.

5. File Format for the component transfer matrix of a hierarchical component

The user needs to supply the diagnosis program with the component transfer matrices of all the hierarchical components in the CUT. If a hierarchical component has entries hierCompoName and funNum in the component name and function number fields in the netlist file respectively, the diagnosis program will read the user supplied component transfer matrix for the hierarchical component from the file ***hierCompoName_funNum***, which has the specified tabulate format below:

$$
\begin{array}{l}
\text{i/v<tab>}Z_{00}\text{<tab>}Z_{01}\text{<tab>} \ldots\ldots\ldots\ldots\ldots \text{<tab>}Z_{0K} \\
\text{i/v<tab>}Z_{10}\text{<tab>}Z_{11}\text{<tab>} \ldots\ldots\ldots\ldots\ldots \text{<tab>}Z_{1K} \\
: \quad : \quad : \quad : \quad : \quad : \quad \ldots\ldots\ldots\ldots\ldots \quad : \quad : \\
: \quad : \quad : \quad : \quad : \quad : \quad \ldots\ldots\ldots\ldots\ldots \quad : \quad : \\
\text{i/v<tab>}Z_{K0}\text{<tab>}Z_{K1}\text{<tab>} \ldots\ldots\ldots\ldots\ldots \text{<tab>}Z_{KK}
\end{array}
$$

In the above case the hierarchical component has 1+k edges. A line of entries in the file consists of an indicator, which is either 'i' (cotree edge) or 'v' (tree edge), and a row of entries from the component transfer matrix. This is in fact an abstraction of a line from the matrix component equation **b=Za**. Each entry in a line is separated by the 'tab'character. A line entry is terminated with a 'carriage return' character. The indicator indicates the edge type of the **b** (output variable) vector element corresponding to the row of entries from the component transfer matrix. As the component transfer matrix is complex with its element in the form a+bj, each single entry from the matrix is thus entered as **ajb**. Below is an example for the file contents of a 2-edge hierarchical component with the component transfer matrix

$$
\begin{bmatrix} 1.010101e^{-3} & -4.545455e^{-4} \\ -4.545455e^{-4} & 6.673114e^{-4} \end{bmatrix} \quad : \quad
\begin{array}{lllll}
\text{i} & \text{<tab>} & 1.010101\text{e}-3\text{j}0 & \text{<tab>} & -4.545455\text{e}-4\text{j}0 \\
\text{i} & \text{<tab>} & -4.545455\text{e}-4\text{j}0 & \text{<tab>} & 6.673114\text{e}-4\text{j}0
\end{array}
$$

6. File Format for the Values of Test Points

The input of test point values from a file is left for future development of the diagnosis program.

7. Outputs of the Diagnosis Program

The diagnosis program outputs to both screen and files. The screen outputs consist of your interactions with the program, the test result for each test cycle and the final diagnosis result. A sample of the screen output which was generated from running the diagnosis program on a fault free circuit is attached at the end of this section.

The outputs to files are the intermediate matrices and results from the diagnosis program, and other information needed for the internal operations of the program. Most of the files are deleted after the program has completed unless the run option ***trace***, ***traceAll***, ***openLoop+trace*** or ***openLoop+traceAll*** is specified.

7.1. Output files which are always kept by the program

- The diagnosis program outputs all error messages to the screen and most of these error messages are also written to the file errorlog which is not deleted after program completion. This file is overwritten whenever the diagnosis program is executed.
- There are two types of files which are created and required by the diagnosis program to derive a set of circuit testing trees from the constraint imposed from the number of circuit nodes & the number of independent voltage sources, and from your preference on the hierarchical components. These files are the swap & history files with file extensions '.sx' (x is a number and its minimum is 1) and '.hst' respectively. There are a set of swap files & a history file associated with each hierarchical component in the CUT. Without these file extensions, the names of the swap files & history file for a hierarchical component are both the same and are formed by the concatenation of the name of the hierarchical component, the symbol '_' and the netlist entry in the

function number field of the hierarchical component (example follows shortly). The set of swap files for the hierarchical component are derived from its component transfer matrices. Since the component transfer matrix of the hierarchical component is always the same irrespective of which CUT the hierarchical component is in (unless a different model of the hierarchical component is used and hence a different netlist entry in the function number field), the diagnosis program only needs to create the swap files for the hierarchical component once and in addition, it outputs a history file which is used to let it know that it does not have to create the swap files for the hierarchical component again when the same hierarchical component is in another CUT. A set of tree & cotree edges partition for the constituent edges of a hierarchical component can be derived from its component transfer matrix. For example, a 2-edge hierarchical component T1 with a 2x2 component transfer matrix is recorded in the file T1_3. Let us suppose that the component transfer matrix specifies that all the edges of T1 are cotree edges. If the matrix has an inverse and its diagonal entries are non-zero, the other tree & cotree edges partitions for T1 will be: all of the edges of T1 are tree edges, edge 1 of T1 is tree edge & edge 2 of T1 is cotree edge (swap edge 1 from cotree to tree edge), and edge 1 of T1 is cotree edge & edge 2 of T1 is tree edge (swap edge 2 from cotree to tree edge). The diagnosis program will output two swap files for T1, T1_3.s1 & T1_3.s2, which correspond to the last two partitions having the type of edges 1 & 2 swapped from cotree to tree. In addition in writing these two swap files, the program will output a history file, T1_3.hst, which contains the number of swap files for T1 in its first line and a '1' in its second line to indicate that the partition having all the edges of T1 swapped from cotree to tree edges exists. When the diagnosis program is run again on another CUT containing T1, the program will check for the existence of the history file of T1 (T1_3.hst) to decide whether it needs to generate the swap files.

7.2. Output files which are deleted when there is no run option specified or the run option is openLoop

- The diagnosis program outputs two types of files for each hierarchical component in the CUT. These files depend on circuit topology and are constructed from the swap files of a hierarchical component and the topology information. They have the extensions '.stable' and '.ptable' which stand for swap and partition tables respectively. Their names without the extensions are the same as the names of their respective swap files without the extension. For a tree & cotree edge partition of the hierarchical component, its corresponding swap file records the indices to the hierarchical edges which must have their types swapped to the edge types opposite to their original types specified by the hierarchical component transfer matrix. The swap table file consists of all the swap files arranged in descending order of the sum of tree edges in the respective tree & cotree edge partitions of the swap files. In the case of two or more swap files having equal sum of tree edges, they are arranged in descending order of the sum of tree edge weights. Once the swap table file is written, the diagnosis program constructs the partition table and outputs it to the partition table file. Each line in the partition table records the graph indices of all the hierarchical tree edges in the tree & cotree edge partition which is recorded in the corresponding line of the swap table.

- The component transfer matrix supplied by the user for a particular hierarchical component is transformed to a new matrix which corresponds to the tree and cotree edge partition of the hierarchical component being assigned for the circuit testing tree in use. This transformation is done for all the hierarchical components in the CUT from the information recorded in the swap tables and the resulting matrices are output to files for use by the program at a later stage. The names of the files to store all the transformed hierarchical component transfer matrices are inputNetlist.x, where x is a number starting from 0. For example, a CUT has hierarchical components T1, T2, T3, with the hierarchical components arranged in descending order of their number of constituent edges (T3>T2>T1). The diagnosis program will store the transformed matrices in inputNetlist.0, inputNetlist.1 and inputNetlist.2 for the hierarchical components T3, T2 and T1 respectively. If T2 and T1 have the same size, and T1 appear before T2 in inputNetlist, inputNetlist.1 and inputNetlist.2 will store the transformed matrices for T1 and T2 respectively.
- The file inputNetlist.Amat is used to store the node incidence matrix and a column swapped version of the node incidence matrix for the circuit testing tree in use.
- The file inputNetlist.Dmat is used to store the fundamental matrix derived from the node incidence matrix.
- The file inputNetlist.Lmat is used to store the connection equation derived from the fundamental matrix.
- The file inputNetlist.node is used to store the matrices which map cotree & tree edge voltages to node voltages.

- The file inputNetlist.tp is used to store the measurement equation derived from the test points and connection equation.
- The file inputNetlist.tpt is used to store all the possible testee partitions for the circuit testing tree in use by storing the graph indices corresponding to the testee partitions. However, not every testee partition in inputNetlist.tpt can be used for testing because of the rule that all the constituent edges of a hierarchical component must be partitioned in the same testee or tester group.
- The file counts.log is used to store the counts for the number of non-testable test cycles and the number of test cycles which use the solution of the Tableau equation to do testing.
- The file testee.log is used to store the testee partitions of all the test cycles completed for the circuit testing tree.

7.3. Sample of output to screen

```
mix% diagnosis1 31compodes6


**************************************************
*                                                *
* diagnosis1    written by Chung Kin Ho,   July, 97   *
*                                                *
* Copyright                                      *
* School of Electrical & Electronic Engineering  *
* University of Bath, Bath, UK                   *
*                                                *
* This program is based on the research project  *
* partly funded by the UK EPSRC, Grant GR/J90596 *
*                                                *
**************************************************



Program starts on Tue Aug 12 17:41:00 1997



There are 14 graph edges to represent 10 circuit components
Number of hierarchical components is 2
Press 'r' to redisplay the following list

List of hierarchical components & their corresponding indices:
0     Tfour1_
1     TPI1_

If test points selected are on the hierarchical components
How many hierarchical components do you want to have as
many voltage test points as possible: 0


Pre-test processing for reading of netlist & hierarchical components completes on Tue Aug 12
17:41:02 1997



Pre-test processing time: 2 sec

Please enter a global percentage limit for the decision algorithm: 0.1


Choice of Testing Strategies:
Please enter 1 for using Node Voltages as test points
          2 for using both Node Voltages & branch currents as test points
Your Choice: 2


Choice of Diagnosis Depth:
Number of maximum faults must be between 1 & 7
Please enter no. of maximum faults: 1
```

```
No. of voltage/current pairs in testee group is at least 6

No. of test points needed is thus set to at least 6

Selecting Test Points.....

Test Point Selection has completed successfully

Test node voltages selected:
v3
v4
v2
v7
v6
v5


7 test points selected:
vTfour1_2
vTfour1_3
iTfour1_1
iTPI1_2
iTfour1_4
vR6
vR4

Test Point Inputs from Measurements or Simulations
for tree number 1
Press 'm' for manual input or 'f' for input from a file: m

Input mode for complex quantities
Press 'p' for polar or 'r' for cartesian mode: r

Please enter the voltage for Tfour1_2
Please enter real part: 3.781451654

Please enter imaginary part: 0

Please enter the voltage for Tfour1_3
Please enter real part: 0.645589638

Please enter imaginary part: 0

Please enter the current for Tfour1_1
Please enter real part: 2.877301e-3

Please enter imaginary part: 0

Please enter the current for TPI1_2
Please enter real part: 0.175715e-3

Please enter imaginary part: 0

Please enter the current for Tfour1_4
Please enter real part: 0.066658e-3

Please enter imaginary part: 0
Please enter the voltage for R6
Please enter real part: 4.005205508

Please enter imaginary part: 0

Please enter the voltage for R4
Please enter real part: 0.126963915

Please enter imaginary part: 0

testing frequency=0.000000
The program will perform at most 1716 test cycles
This number is reduced substantially by the fact that the constituent elements of
a hierarchical component must be partitioned in the same testee or tester group

Test Cycle is 0 for tree number 1
Press 'y' to manually input the tester partition or 'n' to
have the program choose the component partitions: n


vector a for test cycle 1, tree number 1:
7, iTfour1_1
8, iTfour1_2
9, iTfour1_3
10, iTfour1_4
11, iTPI1_1
12, iTPI1_2
1, vR0
2, vR1
3, vR3
4, vR4
5, vR5
6, vR6
13, vRL

Matrix is singular,ZLUdecomposition() failed!

inverseOfZmat() failed!

Q has no inverse!!
Matrix is singular,LUdecomposition3() failed!

inverseOfMat3() failed!

Q' has no inverse!!
For tree number 1, partition chosen for test cycle 1 is not testable

vector a for test cycle 2, tree number 1:
7, iTfour1_1
8, iTfour1_2
9, iTfour1_3
10, iTfour1_4
6, vR6
13, vRL
1, vR0
2, vR1
3, vR3
```

```
4, vR4
5, vR5
11, iTPI1_1
12, iTPI1_2

Q has an inverse!!
Q' has an inverse!!
Check the existence of inverse for L21_2
Matrix is singular,LUdecomposition3() failed!

inverseOfMat3() failed!

L21_2 has no inverse, Solve the Tableau Equation!!

There are 64 possible circuit trees
Number of circuit tree chosen is 1
Test cycle is 2
for tree number 1
vector a2, b2, difference_vector(b2-Z2*a2) & (b2-Z2*a2)/Z2*a2:
vR0=0.449819+j0,    iR0=0.0044982+j0,    1.06745e-08\__0,    0.0002%

vR1=0.768729+j0,    iR1=0.000768728+j0,  1.7559e-09\__3.141592654,    0.0002%

vR3=0.105132+j0,    iR3=0.000105132+j0,  3.63408e-10\__3.141592654,    0.0003%

vR4=0.126964+j0,    iR4=3.8474e-05+j0,   9.35861e-11\__0,    0.0002%

vR5=-0.131482+j0,   iR5=-0.000131481+j0,  1.37418e-09\__0,    0.0010%

iTPI1_1=0.000169955+j0,    vTPI1_1=0.413493+j0,  4.8144e-07\__0,    0.0001%

iTPI1_2=0.000175715+j0,    vTPI1_2=0.544976+j0,  3.03564e-06\__0,    0.0006%


For up to test cycle 2, tree number 1,
components tested to be good are:
Non-hierarchical components    Hierarchical components
R0
R1
R3
R4
R5
                  TPI1_1
                  TPI1_2

All testers are good and thus the test results for the next test cycle are reliable
unless next test cycle is not testable

Testers for test cycle 3, tree number 1 are:

Non-hierarchical components    Hierarchical components
                  TPI1_1
                  TPI1_2
R5
R4
R3
R1
```

```
vector a for test cycle 3, tree number 1:
11, iTPI1_1
12, iTPI1_2
5, vR5
4, vR4
3, vR3
2, vR1
1, vR0
13, vRL
6, vR6
7, iTfour1_1
8, iTfour1_2
9, iTfour1_3
10, iTfour1_4

Matrix is singular,ZLUdecomposition() failed!

inverseOfZmat() failed!

Q has no inverse!!
Matrix is singular,LUdecomposition3() failed!

inverseOfMat3() failed!

Q' has no inverse!!
For tree number 1, partition chosen for test cycle 3 is not testable

vector a for test cycle 4, tree number 1:
7, iTfour1_1
8, iTfour1_2
9, iTfour1_3
10, iTfour1_4
5, vR5
13, vRL
1, vR0
2, vR1
3, vR3
4, vR4
6, vR6
11, iTPI1_1
12, iTPI1_2

Q has an inverse!!
Q' has an inverse!!
Check the existence of inverse for L21_2
Matrix is singular,LUdecomposition3() failed!

inverseOfMat3() failed!

L21_2 has no inverse, Solve the Tableau Equation!!

There are 64 possible circuit trees
Number of circuit tree chosen is 1
Test cycle is 4
for tree number 1
vector a2, b2, difference_vector(b2-Z2*a2) & (b2-Z2*a2)/Z2*a2:
vR0=0.449819+j0 ,     iR0=0.0044982+j0 ,     1.20487e-08\__0 ,     0.0003%
```

```
vR1=0.768729+j0,    iR1=0.000768728+j0,  1.7559e-09\__3.141592654,    0.0002%

vR3=0.105132+j0,    iR3=0.000105132+j0,  3.63408e-10\__3.141592654,   0.0003%

vR4=0.126964+j0,    iR4=3.8474e-05+j0,  9.35861e-11\__0,    0.0002%

vR6=4.00521+j0,    iR6=0.000852173+j0,  1.37418e-09\__0,    0.0002%

iTPI1_1=0.000169956+j0,    vTPI1_1=0.413493+j0,  1.48032e-06\__3.141592654,
0.0004%

iTPI1_2=0.000175715+j0,    vTPI1_2=0.544976+j0,  1.69937e-06\__0,    0.0003%


For up to test cycle 4, tree number 1,
components tested to be good are:
Non-hierarchical components    Hierarchical components
R5
R0
R1
R3
R4
R6
                    TPI1_1
                    TPI1_2

All testers are good and thus the test results for the next test cycle are reliable
unless next test cycle is not testable

Testers for test cycle 5, tree number 1 are:

Non-hierarchical components    Hierarchical components
R0
R1
R3
R4
R5
R6

vector a for test cycle 5, tree number 1:
1, vR0
2, vR1
3, vR3
4, vR4
5, vR5
6, vR6
7, iTfour1_1
8, iTfour1_2
9, iTfour1_3
10, iTfour1_4
13, vRL
11, iTPI1_1
12, iTPI1_2

Q has an inverse!!
Q' has an inverse!!
Check the existence of inverse for L21_2
Matrix is singular,LUdecomposition3() failed!
```

```
inverseOfMat3() failed!

L21_2 has no inverse, Solve the Tableau Equation!!

There are 64 possible circuit trees
Number of circuit tree chosen is 1
Test cycle is 5
for tree number 1
vector a2, b2, difference_vector(b2-Z2*a2) & (b2-Z2*a2)/Z2*a2:
iTfour1_1=0.0028773+j0,    vTfour1_1=4.55018+j0,  1.66673e-06\__3.141592654,
0.0000%

iTfour1_2=0.000768728+j0,    vTfour1_2=3.78145+j0,  1.19264e-06\__3.141592654,
0.0000%

iTfour1_3=-0.000105132+j0,    vTfour1_3=0.64559+j0,  4.53165e-07\__3.141592654,
0.0001%

iTfour1_4=6.6658e-05+j0,    vTfour1_4=0.540458+j0, 2.51559e-07\__0,    0.0000%

vRL=0.544974+j0,    iRL=0.000544976+j0,   1.3429e-09\__0,    0.0002%

iTPI1_1=0.000169955+j0,    vTPI1_1=0.413494+j0,  1.37591e-06\__0,    0.0003%

iTPI1_2=0.000175715+j0,    vTPI1_2=0.544974+j0,  2.20359e-06\__0,    0.0004%

no faulty component found, the circuit is fault free!!
Program completes on Tue Aug 12 17:43:00 1997


Execution time: 2 min 0 sec
```

8. Error Messages and System Setting

When errors occur within the program, an error message together with the names of the failed internal function calls will be output to the screen. This information should be reported to the author if you think the error is due to bugs in the diagnosis program.

If a component entry line in the input netlist file begins with a capital B, M or O, the diagnosis program will display an error message about the input format for a BJT, a MOST or an Op-Amp respectively. This error message should be neglected and a transistor or an Op-Amp should be input as a multi-terminal or multi-port component.

The only requirement on system setting is that no alias should be made to the UNIX command **'rm'** as the diagnosis program uses a system call to **rm** to delete intermediate files. For example, the effect of the command, **alias rm 'rm -i'**, is that you have to confirm deletion of every intermediate file the program is going to delete after program completion.

9. Limitation

Although the inclusion of a controlled source and its controlling element not as a hierarchical component in the input netlist file is allowed, the diagnosis program can only handle them as two separate components in the run options ***openLoop***, ***openLoop+trace*** and ***openLoop+traceAll***. This is because the automatic repartitioning of the tester group from previous test cycle result would have been a lot more complicated to code if a controlled source & its controlling element had been treated as separate components. In addition to this limitation, the controlling element must be non-hierarchical.

1 C. Wey, "Design of Testability for Analogue Fault Diagnosis", Int. J. of Circuit Theory and Applications, Vol. 15, pp.123-142, 1987.

2 Al Kelly, Ira Pohl, "A book on C", P. 58-59, The Benjamin/Cummings Publishing Company, Inc., 1984.