PHD

Testing techniques for analogue and mixed-signal integrated circuits

Al-Qutayri, Mahmoud A.

Award date:
1992

Awarding institution:
University of Bath

TESTING TECHNIQUES FOR ANALOGUE AND MIXED-SIGNAL INTEGRATED CIRCUITS

by

MAHMOUD A. AL-QUTAYRI

for the degree of

Doctor of Philosophy

UNIVERSITY OF BATH

1992

Signature ....

To My Parents (Abdullah & Niema)

and

My Wife (Maysoon)

ABSTRACT

The objective of this thesis is the development of a unified testing technique for analogue and mixed-signal integrated circuits.

The main mechanisms of failure in ICs are outlined to illustrate the need for testing and to show the basis of circuit fault models. Then a general overview of the issues involved in testing digital, analogue and mixed-signal circuits and systems is presented.

The dc fault dictionary, digital modelling and logical decomposition approaches that were originally devised for testing discrete analogue circuits are studied, to evaluate their applicability to testing analogue and mixed-signal ICs. Of these approaches the dc fault dictionary and digital modelling have good potential for testing analogue cells after introducing some improvements to the original strategies.

A novel technique called time-domain testing is then presented. The technique is based on exciting an analogue or a mixed-signal circuit-under-test (CUT) with a pseudo-random binary sequence (PRBS) test signal and measuring the transient response generated at the external nodes. This enables the testing of the CUT to be achieved in a unified fashion by eliminating the need for partitioning to separate analogue and digital modules in the case of mixed-signal ICs, and intermediate probing. The technique also has the potential to be implemented on a digital tester to reduce the time and cost of testing.

Both transient voltage at the external CUT node and transient supply current are measured in the time-domain technique to study their effect on fault coverage. The samples values, rate of change, auto and cross correlations, and response digitization analysis methods are then used to analyse the transient response data. Of these methods, the response digitization is the one most attractive due to its computational efficiency and compatibility with a digital tester.

The effectiveness of the time-domain technique in testing analogue and mixed-signal circuits, with catastrophic and soft fault conditions, is studied both by computer simulation, and experimentally using a prototype transient response capturing system. The results indicate that the technique achieves a high fault coverage.

ACKNOWLEDGEMENTS

The author would like to thank and express his sincere appreciation to his advisor Dr Peter R. Shepherd for providing the initial direction of this research and for his expert guidance throughout the course of this work.

The author also wishes to thank Mr John D. Martin for useful discussions on topics related to signal processing.

This research work was conducted under grants GR/G 23937 and GR/D 14488 from the UK Science and Engineering Research Council.

The author would also like to thank his parents and all members of his family for their invaluable support and encouragement.

Many thanks are also due to the authors wife, Maysoon, for being his inspiration, for patiently waiting for him to finish and for helping in typing the original manuscript.

TABLE OF CONTENTS

GLOSSARY OF SYMBOLS

∂	Partial derivative
$\oplus$	Exclusive-OR operation
Δt	Sampling interval
ΔV	Small tolerance voltage
$\delta(t)$	Delta or impulse function
ν , τ	Time shift constants
$\phi_{xx}(\tau)$	Auto correlation function of x(t)
$\phi_{xy}(\tau)$	Cross correlation function of x(t) with y(t)
ABIST	Analogue built-in self test
ADC	Analogue-digital converter
Al	Aluminium
ASIC	Application specific integrated circuit
ASR	Analogue shift register
ATE	Automatic test equipment
Au	Gold
BILBO	Built-in logic block observation
BIST	Built-in self test
CAD	Computer aided design
CERDIP	Ceramic dual-in-line package
CMOS	Complementary metal oxide semiconductor
Cu	Copper
CUT	Circuit under test
CV	Coefficient of variation
D	Percentage of deviation
DAC	Digital-analogue converter
DFT	Design for testability
DID	Diffusion induced dislocations
dn	Number of detection instances

DSP	Digital signal processing
DTL	Diode-transistor logic
EED	Emitter edge dislocations
EOS	Electrical overstress
ESD	Electrostatic discharge
F	Total number of faults introduced to a circuit
F_c	Number of faults simulations that did not converge
F_d	Number of faults detected
HBIST	Hybrid built-in self test
h(t)	Impulse response function
$H(\omega)$	Fourier tranform of h(t)
IC	Integrated circuit
Idd	Dynamic voltage supply current
IDDQ	Static voltage supply current
L	Transistor length
LDF	Logical diagnostic function
LFSR	Linear feedback shift register
LPF	Low-pass filter
LSSD	Level sensitive scan design
M	Total number of samples
MeV	Mega electron volt
MISAR	Multiple input signature analysis register
MUX	Multiplexer
N	Number of bits in PRBS
PED	Plastic encapsulated device
pdcf	Primitive D-cubes of failure
PODEM	Path oriented decision making
PRBS	Pseudo-random binary sequence
PSG	Phosphosilicate glass
R	Period of PRBS
RAM	Random access memory
R_t	Rate of change

RTL	Resistor-transistor logic
s-a-1	Stuck-at one
s-a-0	Stuck-at zero
Si	Silicon
SiO_2	Silicon dioxide
S_n	Sensitivity factor of node n
SRL	Shift register latch
$S_{xx}(\omega)$	Cross spectral density function
$S_{xy}(\omega)$	Auto spectral density function
T	Bit interval of PRBS
TAP	Test access port
TCH	Transient capture hardware
TTL	Transistor-transistor logic
u	Number of distinct lines
v	Number of distinct transistors
V_n	Nominal voltage of node n
V_f	Faulty voltage of node n
V_t	Threshold voltage
VLSI	Very large scal integration
W	Transistor width
Yf	Response of CUT
Yn	Response of fault-free CUT
$\bar{Y}$n	Average of fault-free CUT response

CHAPTER ONE

INTRODUCTION

Testing is one of the major bottle-necks in the production of integrated circuits (ICs). As the ICs grow in complexity the task of testing them also becomes more complex leading to an increase in the time, and hence cost of production.

For digital ICs, including complex VLSI (Very-Large-Scale-Integration) ones, the task of testing is now manageable. This has been achieved by the use of simple fault-models, devising efficient test pattern generation algorithms to detect the faults synthesized by the fault-models, and the introduction of design-for-testability (DFT) techniques, especially for sequential circuits, during the early stages of an IC design to enhance its testability. The term testability simply refers to the ease with which a circuit can be tested. Software test tools and automatic test equipment (ATE), that implement most of the testing algorithms and testability enhancement techniques for digital ICs, are now widely available. Many of these software tools can perform testability analysis on the circuit to identify the nets that are difficult to test or untestable. This allows the designer to change the design of those nets or introduce DFT techniques to make them testable. Another important CAD (Computer-Aided-Design) tool that is available for a digital circuit designer is an efficient simulator.

The testing algorithms and DFT techniques for digital ICs, and the test tools that implement them have no counterparts for analogue integrated circuits. This is due to the complex characteristics of analogue circuits. The characterisation of analogue circuits depends on continuous descriptive variables, such as magnitude and frequency of an amplifier gain, rather than the discrete variables of logic levels in a digital circuits. This makes the design and test of an analogue circuit more knowledge intensive than its digital counterpart. The requirement to handle the complex, and many times conflicting, parameters of an analogue circuit resulted in the lack of efficient simulation and test CAD tools similar to those for digital circuits. At present, testing an analogue IC involves verifying that the manufactured IC meets the design specifications, such as gain, bandwidth and noise margin. This process is time consuming and costly.

Mixed-signal integrated circuits are ICs that have both analogue and digital circuitries on a single chip. These ICs only became available recently, mid-seventies, following technological advancements in CMOS (Complementary-Metal-Oxide-Semiconductor) and BiCMOS (Bipolar and CMOS) fabrication processes. They result in an improvement in system performance and reduction in cost. However, the presence of analogue and digital circuits on a single device, with usually very few or no accessible internal signals, makes the task of testing such devices a very difficult one. In practice mixed-signal ICs are tested by partitioning the device into separate analogue and digital modules, by including extra probe pads to provide access to internal signals, and applying mode specific tests to each module. The inclusion of probe pads results in an increase in the wafer area required, reduces IC reliability and increases its overall cost. The application of mode specific test would require two testing set ups; one for analogue and one for digital, resulting in an increase in testing time and cost. Partitioning of a mixed-signal IC during testing also means that the interaction between the analogue and digital modules through the interface blocks, such as ADCs (Analogue-Digital Converter) and DACs (Digital-Analogue Converters) cannot be tested.

The main goal of this thesis is to propose and evaluate a unified testing technique, called time-domain testing, for analogue and mixed-signal integrated circuits. The technique eliminates the requirement for partitioning mixed-signal ICs and can be implemented on a digital tester, resulting in reduction of testing time and cost. Both catastrophic and soft faults can be handled by the technique.

All the testing techniques investigated in this thesis, including the time-domain one, are for CMOS based analogue and mixed-signal ICs. CMOS is chosen because it is the dominant technology for these ICs currently in the market. In general, only single faults will be assumed, similar to digital testing, because the number of possible multiple faults in an IC is very high, and it would be very complex and computationally expensive to deal with multiple faults. In addition, extensive studies on digital circuits showed that the single fault assumption is a reasonable one and achieves reliable testing results. The main objective of all the techniques investigated is to detect the presence of a fault and not to identify it or locate it. Therefore, the techniques are considered suitable for production (i.e. Go/No-Go) testing.

Chapter 2 gives a review of the mechanisms of failure that occur in integrated circuits during the manufacturing process and in later use. The failures are broadly classified under electrical, intrinsic and extrinsic categories. The objective of the review is to highlight the multitude of failures that can possibly occur in an IC, their effect on the IC and the difficulty in deriving fault models, that abstract the effect of some of the failures, for the purpose of testing.

Chapter 3 presents an overview of testing digital, analogue and mixed-signal circuits. It outlines the various testing stages that an IC goes through. The various fault models that are used in testing digital circuits are then reviewed. The physical transistor based fault-model that will be used in this thesis is also described. The chapter then reviews the test generation algorithms and design-for-testability techniques reported in the literature and used in practice to test digital ICs. The difficulties associated with testing analogue integrated circuits are then presented, along with a description of the testing strategies available for discrete analogue circuits. Finally, the testing techniques suggested in the literature and the issues involved in testing mixed-signal ICs are discussed.

In Chapter 4 three testing approaches that were originally proposed for discrete analogue circuits are studied in detail. The objectives are to investigate their applicability to testing analogue and later mixed-signal ICs, and any necessary modifications. The three techniques include: dc fault-dictionary, digital modelling and logical decomposition. An important focus of the study is to reduce the number of nodes that need to be accessed and yet achieve an acceptable level of fault-coverage.

The unified time-domain testing technique for analogue and mixed-signal ICs is presented in Chapter 5. Both the voltage at the circuit-under-test (CUT) output terminals and the supply current are measured in this technique. Four methods of analysing the time-domain voltage and current data are also described in the chapter. The objective is to determine which type of measurement achieves highest fault-coverage and which method of analysis is most efficient computationally.

Chapter 6 describes the prototype transient response capturing experimental system designed and implemented to verify the time-domain testing strategy and simulation results in Chapter 5. The experimental results of testing analogue and mixed-signal circuits are also discussed in the chapter.

Finally, the main conclusions of the work in this thesis together with suggestions for future research work in the field of testing analogue and mixed-signal integrated circuits are outlined in Chapter 7.

CHAPTER TWO

MECHANISMS OF FAILURE IN ICs - A REVIEW

The failure mechanisms in integrated circuits (ICs) can broadly be classified into the following three categories [1,2]:

1- Electrical stress (in-circuit) failures

2- Intrinsic failure mechanisms

3- Extrinsic failure mechanisms

2.1 Electrical Stress (in-circuit) Failures

Electrical stress failures are caused by two factors which are attributed to either poor design or improper handling of the IC:

1- Electrical overstress (EOS)

2- Electrostatic discharge (ESD)

2.1.1 Electrical Overstress (EOS)

Electrical overstress failure mode is a result of operating an integrated circuit outside the specifications described by the manufacturer for a short period of time. An example of this is high transient pulses having short duration [3].

The effect of EOS is to increase the current flow through the IC leading to the formation of a hot-spot at a semiconductor junction in the device. As the junction being subjected to EOS gets hotter, more current flows through it resulting in an increase in its temperature. If the temperature of the junction reaches the melting point of silicon, the junction short circuits, aluminium migrates down the molten silicon and the metallization open circuits.

To prevent EOS or at least minimize its effect, manufacturers usually include protection circuitry that can withstand excessive electrical stress. However, the protection circuitry can accommodate electrical stress within its design specifications and like the rest of the IC it is prone to other failure modes.

2.1.2 Electrostatic Discharge (ESD)

The input of a MOS device is principally capacitive. Since the capacitance is extremely small, it represents an extremely high input impedance. Any small electrostatic charge that may be induced on the input leads, for example by human body contact, would have no way of leaking off and induce a rather large voltage.

If the ESD induced voltage is high enough (e.g. more than 2KV) it can crack the gate oxide or melt small volumes of silicon, thereby creating minute explosions on the device surface. The result could be voids, cratering, microcracks and subsequent short circuits and open circuit. Small amounts of ESD can damage the device slightly, but enough to cause subsequent damage, such as a reduction in the drain saturation current, early in the device operating life time [1,2,4].

The ESD problem can be greatly reduced by correct handling of the device and providing protection circuitry at all inputs pins. There are many such circuits in use, each with varying degrees of effectiveness. Figure 2.1 shows an example of such circuitry, it consists of a combination of a polysilicon resistor and clamp diodes. The resistor limits the current (and also the speed of operation) and the diodes prevent voltages excursions beyond the power and ground rails. Though ESD protection circuits are effective, they tend to occupy valuable die area, slow device speed of operation and are uneconomical to provide in all chips. Therefore, most ICs are considered to be ESD sensitive if the necessary handling and environmental precautions are not taken.

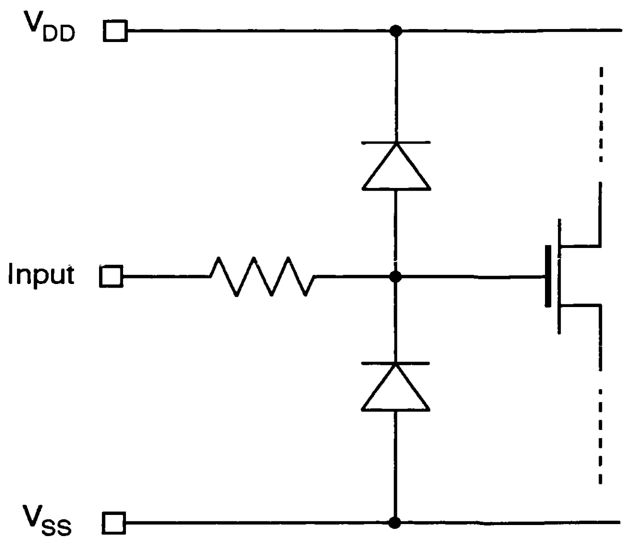


Figure 2.1: ESD Protection Circuitry

2.2 Intrinsic Failure Mechanisms

Failure modes that fall under the intrinsic mechanisms category include:

1- Gate Oxide Breakdown

2- Ionic Contamination

3- Surface Charge Spreading

4- Charge Effects

5- Piping

6- Dislocations

2.2.1 Gate Oxide Breakdown

Gate oxide breakdown is one of the predominant failure modes in MOS integrated circuits. There are two main reasons for oxide breakdown [1,5,7]:

1- Due to EOS and ESD

2- Time-dependent defects which occur during operation within rated conditions of voltage, temperature and power dissipation.

EOS and ESD have been discussed in section 2.1, where it was shown that these modes of failure can be avoided by correct handling procedures and using protection circuits.

Time-dependent breakdown can take place at weaknesses in the oxide layer, which are due to poor processing such as etching steps or an uneven oxide growth [7,8]. A polysilicon gate process has fewer gate oxide defects than a metal gate process, because the thermally grown oxide is immediately covered with a polysilicon layer before any etching steps are performed, provided no buried substrate polysilicon contact is required. However, pin holes in the photoresist during etching can still lead to local weaknesses in the gate oxide. Time-dependent breakdown can also involve the movement of contaminating alkali ions (e.g. sodium) in the gate oxide. Where alkali ion contamination is involved the failure mode has a large activation energy, but otherwise breakdown is only weakly affected by temperature. In both cases the failure rate is strongly affected by applied voltage.

Since gate oxide breakdown depends strongly on applied voltage, an effective screening procedure is to overstress the device, typically by applying a voltage which exceeds the maximum specified level by 50% [1].

2.2.2 Ionic Contamination

The contamination of oxide with mobile alkali ions, such as Na^+, Li^+ and K^+, is one of the serious failure mechanisms in ICs. In general this mechanism results in mobile positive charges trapped within the gate oxide. Improvements in the manufacturing process have greatly reduced this problem. However, process variations and the introduction of new sources of material can still lead to ionic contamination. The main processes that introduce contamination to the oxide are [1,9]:

1- The environment: water vapour and dust particles are contaminants.

2- Human contact: sweat, saliva and minute skin particles.

3- Processing materials: etchants, scribes and furnaces.

4- Packaging: materials, adhesives and lead frame.

The presence of positive charges inside the dielectric oxide and at the silicon-oxide interface affect the characteristics of both MOS and bipolar devices. The effect on MOS devices is illustrated in Figure 2.2 [5]. In NMOS devices (Figure 2.2a) with a positive bias applied on the gate the ions move towards the silicon surface and induce an extra negative charge in the channel. The result is a decrease in the threshold voltage V_{th} of the NMOS transistor. Positive charges are also attracted to the gate oxide of an NMOS devices from surrounding areas by the favourable direction of the gate electric field.

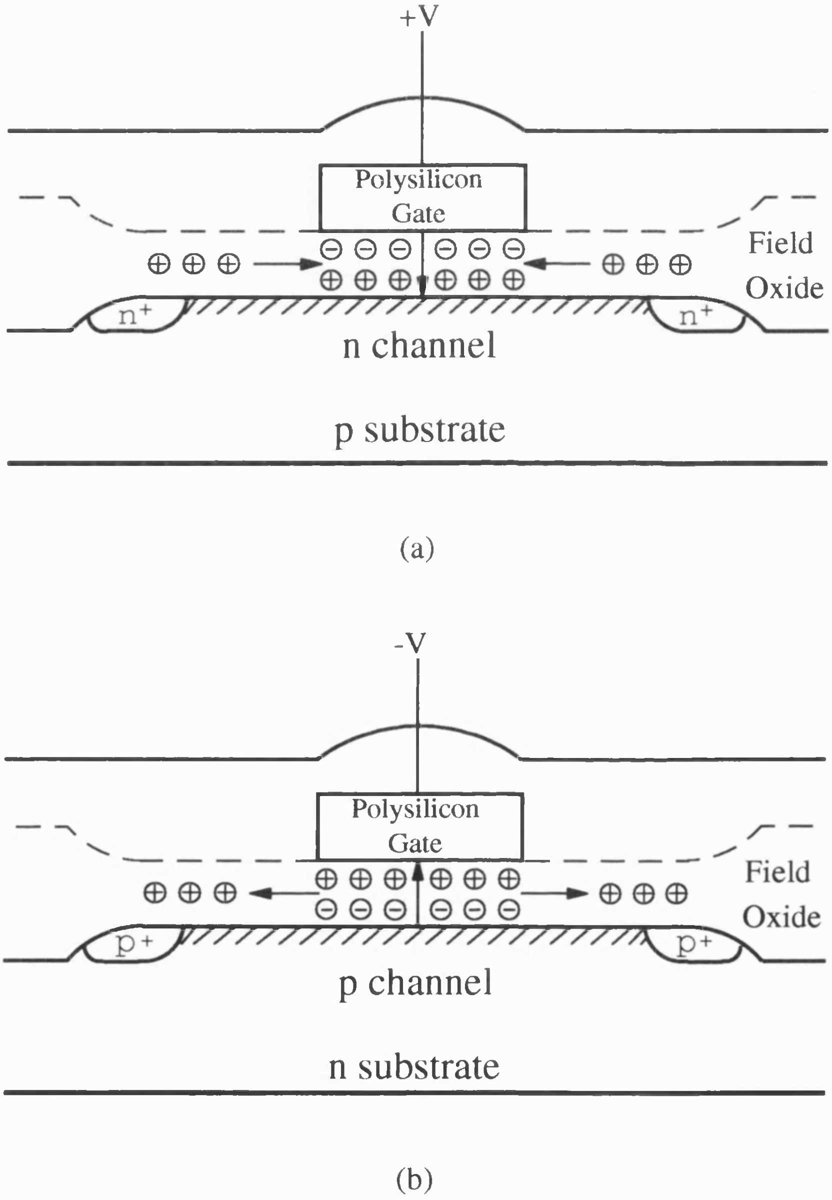
In PMOS devices (Figure 2.2b) with a negative bias applied to the gate, the direction of the electric field at the gate oxide repels positive charge a way from the gate area resulting in an increase in the device threshold voltage. Due to the direction of the electric field at the gate oxide, NMOS devices are more susceptible to mobile-ion threshold shift than PMOS devices.

In bipolar devices, mobile-ions effectively change the concentration of minority/majority carries. This results in a change in the transistor forward current gain and a shift in the collector junction avalanche breakdown voltage, both effects are related to the performance of the device [6].

As an effective barrier against mobile-ions contamination [1,6,10], the outer surface of the chip is usually coated with a passivation layer of phosphosilicate glass (PSG) over which either another layer of PSG or a layer of silicon nitride is laid, this is schematically indicated in Figure 2.3 [1]. Unfortunately, although this is effective against penetration from the external environment it provides no protection against built-in impurities.

2.2.3 Surface Charge Spreading

In addition to the drift of alkali ions in the oxide-bulk, described in the previous section, it is also possible for charge to spread out laterally from biased metal conductors [5,10]. The charge moves either on the oxide surface or along the silicon-oxide interface. Instead of changing the MOS device parameters, the effect of surface charge spreading is the formation of an inversion layer outside the active region of the transistor as illustrated in Figure 2.4 [5]. This inversion region provides a conduction path between two diffused regions, or extends the p-n junction through a high-leakage region.



(a)

(b)

Figure 2.2: Effect of Ionic Contamination on: (a) NMOS, (b) PMOS

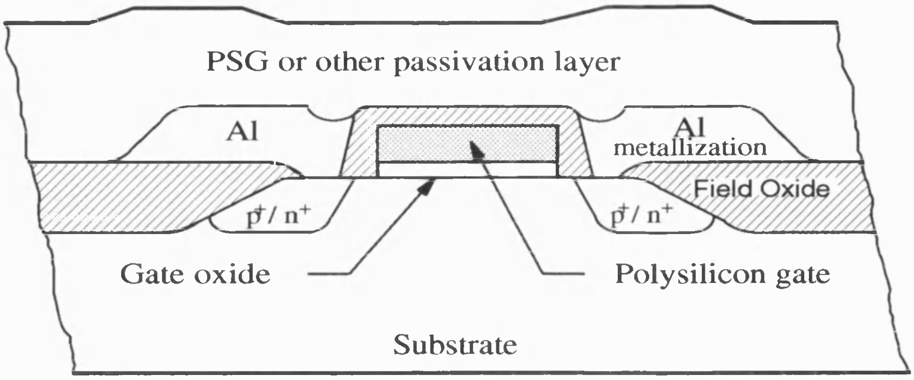


Figure 2.3: Cross-Section of a MOS Device with a Passivation Layer

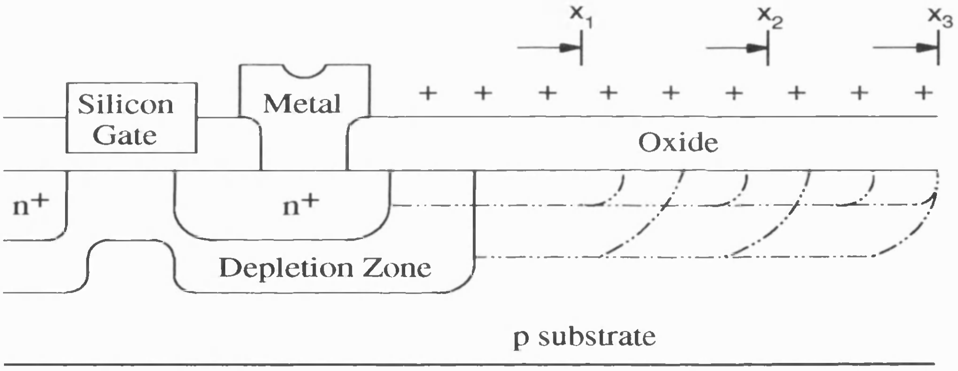


Figure 2.4: Physical Model of Surface Charge Spreading

The speed with which charge spreads increases with temperature, and charge is transported by lateral ion movement or by the surface becoming partially conductive in the presence of moisture.

This mode of failure can result in leakage currents along the oxide surface between neighbouring conductors, short circuiting between adjacent devices, or the formation of parasitic transistors.

Since surface charge spreading depends strongly on high temperature [1], an effective screen against it is either a high-temperature storage bake between 150° C and 250° C or a high-temperature reverse-bias test between 125° C and 250° C.

2.2.4 Charge Effects

This section discusses slow trapping and hot electrons failure mechanisms, which are caused by the movement of charge through a device oxide. These mechanisms are particularly important for MOS memory devices due to their dependence on the critical charge stored at the gate oxide capacitor.

2.2.4.1 Slow Trapping

The programming of some types of memory devices is performed by the transport of charge from the source or drain through the gate oxide to the gate interface. Interstitial states at the Si-SiO_2 interface trap electrons and hold them in the oxide as illustrated in Figure 2.5 [1]. This results in a permanent shift in the transistor threshold voltage, and a decrease in the speed at which the device can be programmed due to the presence of fields that oppose the electrons flow through the oxide.

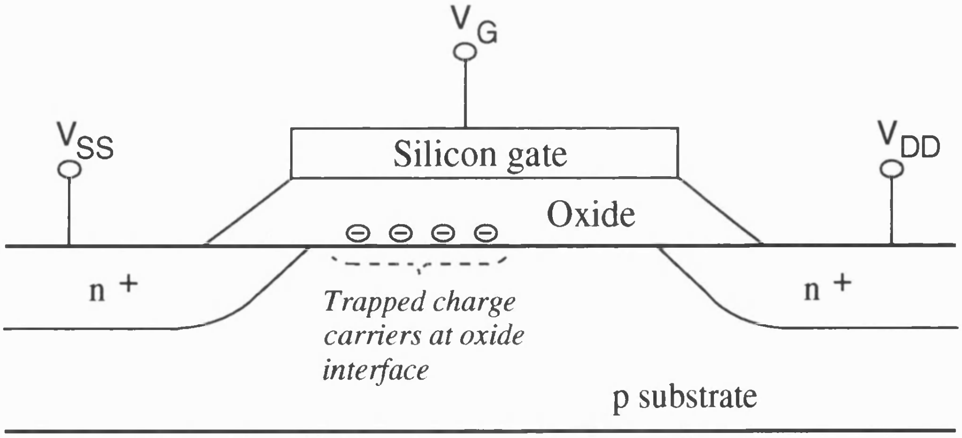


Figure 2.5: Slow Trapping in an NMOS Transistor

In MOS devices slow trapping occurs when sufficiently high temperatures combined with high electric fields provide electrons with enough energy to cross Si-SiO_2 interface.

The slow trapping problem can be reduced by the controlled growth of the gate oxide in order to reduce the density of available electron traps.

2.2.4.2 Hot Electrons

The scaling down of transistor geometry in high-density integrated circuits without a corresponding scaling of the voltages results in the concentration of the current flow in a shallow channel near the silicon surface, high electric fields in the conduction channel, and hence the generation of hot electrons [1,5,11]. The electrons may originate in the substrate due to thermal currents, or from the channel current. The electrons are accelerated, under the influence of a high electric field, from the source towards the drain. Impact ionization collisions scatter the hot electrons and also multiplies the channel current. Some of the scattered electrons will have enough energy to surmount the silicon-silicon dioxide potential barrier and may be injected and trapped in the gate-oxide as illustrated in Figure 2.6 [1]. Subsequent trapping of the injected electrons will cause devices instabilities, such as threshold-voltage shift and transconductance degradation. A number of optimum design structures which will minimize the hot electron trapping problem are investigated in [12].

2.2.5 Piping

This failure mode is caused by shunting paths, created within the silicon by enhanced diffusion through crystal defects generated during the wafer processing. The prevalent cause of these pipes or paths is phosphorus diffusion along the crystal defects [13,14].

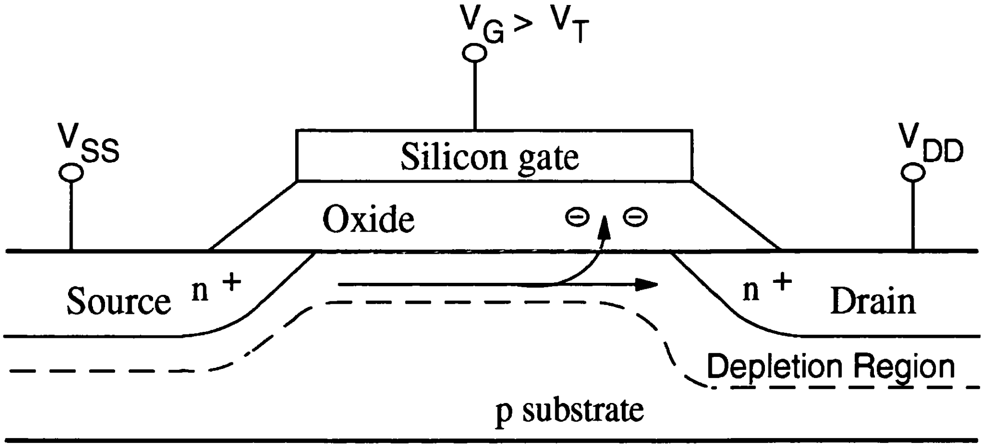


Figure 2.6: Hot Electrons Effect in an NMOS Transistor

The performance of a bipolar transistor is critically dependent on the integrity of the epitaxial layer. Misalignments and shifting of masks lead to the formation of defects within the epitaxial layer. Problems with the epitaxial layer deposition are associated with the need to deposit an epitaxial n^- region on a p substrate, over a buried n^+ layer, such as the case of an n-p-n bipolar transistor.

The predominant failure associated with this mechanism is a collector-to-emitter shunt as depicted in Figure 2.7 [1] with the resulting resistance as low as a few ohms up to megohms. Collector-to-base pipes can also occur. The resistance of these pipes is only linear at low voltages and increases with increasing current density until junction breakdown.

Pipe defects can be reduced by improving epitaxial deposition methods and buried n^+ layer formation at the manufacturing stage.

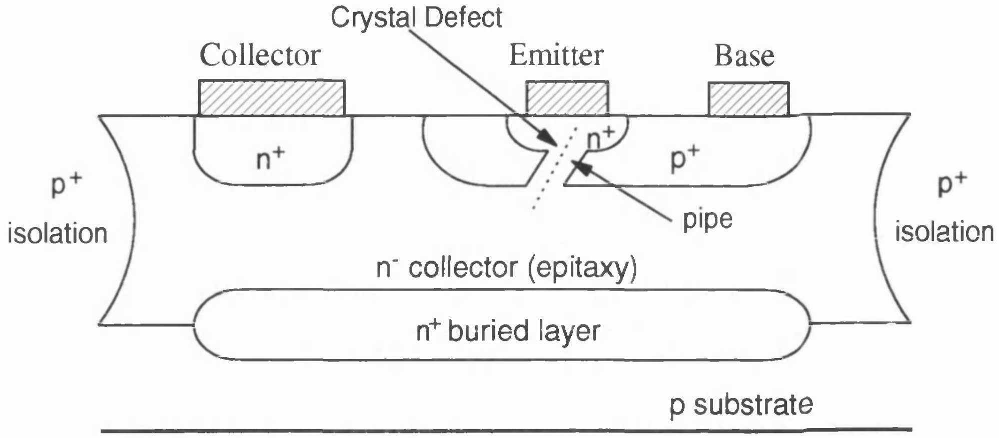


Figure 2.7: An (n) Pipe Through the Base Region Due to a Crystal Defect

2.2.6 Dislocations

Dislocations are caused by diffusion processes within the silicon that affect device performance [1,6,13]. Two types of dislocations may be created by impurity (e.g. phosphorus) diffusion: diffusion-induced-dislocations (DID), which appear within diffused areas, and emitter-edge-dislocations (EED), which appear around edges of the diffused areas.

In bipolar transistors, dislocations result in a reduction of the current gain, an increase in leakage currents of p-n junctions, shorts due to enhanced diffusion at dislocations and an increase of the low-frequency noise. Dislocation in MOS devices result in a shift of the threshold voltage due to an increase of the Si-SiO_2 interface trap densities, and a reduction of transistor transconductance due to a decrease in the surface mobility of charge carriers.

2.3 Extrinsic Failure Mechanisms

Extrinsic modes of failure are attributed to the following manufacturing processes and operating environment conditions:

1- The packaging

2- The metallization

3- Bonding

4- Die attachment

5- Particulate contamination

6- Radiation

2.3.1 The Packaging

The package encapsulating an IC provides the physical support for the silicon device and enables electrical connection to be made to the external circuitry. The package also gives electrical and mechanical protection for the device, therefore it must present an inert and dry atmosphere to the surface of the semiconductor, and must give protection against mechanical and thermal shock. The two main types of packaging technologies used are [1,2]:

1- Hermetic or CERDIP (ceramic dual-in-line package)

2- Plastics (PEDs - plastic encapsulated devices)

Hermetic packages consist of a ceramic material which forms a protective hollow shell around the semiconductor. The shell is filled with a non-reactive gas, such as nitrogen, which creates a sterile operating environment about the bare semiconductor device. The package is proven to be very strong and reliable, however the need to make connections to the outside world means that complete protection cannot be guaranteed. Therefore, poor hermeticity of package allows moisture and alkali ions to penetrate on to IC, causing corrosion and oxidation of metallization, and ionic contamination problems [1].

The plastic package is seen most often in the high volume produced ICs. The encapsulation is performed by a moulding process. The die is attached to a lead frame which is encapsulated by transfer moulding the plastic around it, thereby creating a solid unit. Therefore, the die is in direct contact with the plastic material, thereby eliminating the need for a hermetic seal. The absence of a hermetic seal may lead to mechanical stress on the structure during the moulding process, which can directly affect the semiconductor properties of the silicon. In addition, because plastic compounds are porous and moisture ingress contaminants from the plastic can transfer to the silicon resulting in ionic contamination, corrosion and oxidation of metallization [2,15].

2.3.2 The Metallization

Aluminium is used predominantly for metallization and interconnections in integrated circuits. It forms good low resistance contacts and is very reliable. The uniformity of the metallization layer is of paramount importance, and the contours of the semiconductor surface must be precisely followed to prevent the formation of voids or cracks. The integrity and reliability of the metallization [1,13] is affected by the corrosion, electromigration, contact migration, microcracks and mechanical stress relaxation mechanisms.

2.3.2.1 Corrosion

For corrosion to occur all the following conditions must be present simultaneously [1]:

1- moisture

2- dc operating potentials

3- sodium or chloride ions to act as a catalyst in the reaction.

The effect of corrosion is generally the formation of open circuits in the metallization, with the area around the bonding pads being most susceptible to it. This mode of failure affects both bipolar and MOS device. However, due to their lower power dissipation, MOS devices are more susceptible to corrosion than bipolar, because the higher power dissipation at the semiconductor surface of bipolar devices reduces the moisture content and hence eliminates one of the vital conditions necessary for corrosion to occur [5,7].

PEDs are more susceptible to corrosion failures than CERDIPs, because the porosity of the plastic results in the diffusion of moisture through the package and down to the metallization. The incidence of corrosion can be reduced by coating the chips, after fabrication, with a protective passivation layer (e.g. SiO_2 & PSG).

2.3.2.2 Electromigration

Electromigration, due to the passage of high current densities along aluminium metallization tracks, is one of the important mechanisms of failure in integrated circuits.

Electromigration is caused by a transfer of momentum during collisions between conducting electrons and aluminium ions which results in a flow of metal ions in the direction of electron flow. A void is therefore created at one end of the metallization track while metal is piled up at the other end. Once voids have formed, the local current density and temperature are increased and electromigration proceeds at an increasing rate until localised open circuit failure occurs [1,2,5,7].

Both bipolar and MOS devices are susceptible to electromigration. However, the electromigration phenomenon is now well understood and its effect can be reduced by adhering to stringent design rules, which for example require that the current density in the metallization should not exceed 10^6 A/cm^2 [1].

2.3.2.3 Contact Migration

Unlike electromigration, contact migration is concerned with the migration of atoms at the metal-semiconductor interface. The physical bond formed at the contact window on the silicon surface is ideally formed by the exchange of a few atoms of aluminium and silicon. Such a bond would be strong. However, inadequate control of conditions during the exchange process result in the continuation of interdiffusion which leads to changes in the ohmic resistance of the contacts.

Diffusion of silicon into the aluminium results in voiding at the contact and an open circuit is formed. Diffusion of aluminium into silicon causes spiking which could extend into the substrate with a resultant short circuit being formed through the device as depicted in Figure 2.8 [1].

The use of improved alloys of the contact, such as Al/Si and Al/Cu, minimizes the contact migration problem. However, the continuous reduction in devices geometries makes them susceptible to smaller amounts of interdiffusion.

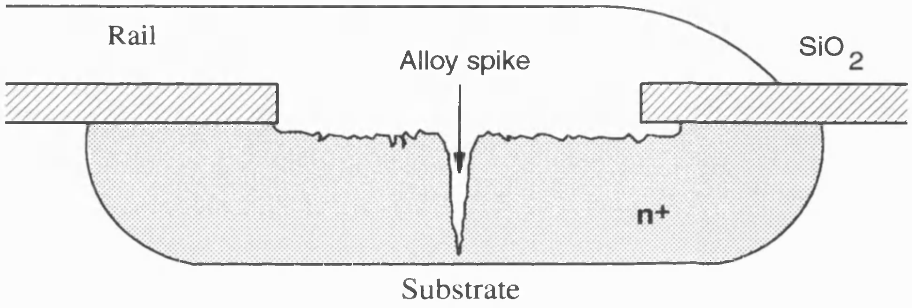
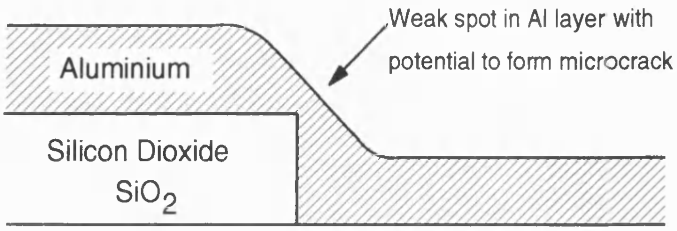


Figure 2.8: Schematic of Spike Formation

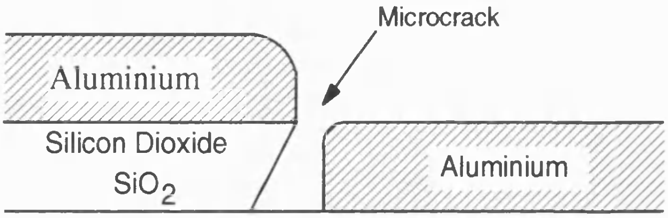
2.3.2.4 Microcracks

All integrated circuits technologies have metallization lines crossing the edges of oxide or dielectric, hence creating a step. This step is likely to result in a thinly spread metallization over the step area as shown in Figure 2.9a . Thinning of metallization weakens it and increases its susceptibility to electromigration. If the step is under cut, as illustrated in Figure 2.9b, then microcracks can form in the aluminium metallization even for thin oxides.

Microcracks result in open circuits in the metal tracks [1,2,5,6]. Tapering of the oxide into each step as depicted in Figure 2.10 and other improvements in the manufacturing process reduced this failure mode substantially.



(a)



(b)

Figure 2.9: Schematics of Structures with Potential for Microcracks: (a) Step at the Edge of a Thick Oxide, (b) Undercutting of a Thin Oxide

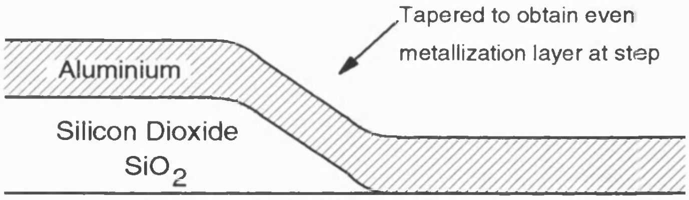


Figure 2.10: Tapering of Oxide Step to Prevent Microcracks

2.3.2.5 Mechanical Stress Relaxation

In this mode of failure the metal atoms migrate from areas of high stress in order to equalize the stresses with the resulting deformation of the metal [16].

The effect of this mechanism is the growth of whiskers due to compressive stress. The presence of metal whiskers may lead to short circuits between adjacent metal tracks. A reduction in whiskers growth can be achieved by using Al alloys (e.g. Al-Si) for metallization.

2.3.3 Bonding

The bond, shown schematically in Figure 2.11 [1], is considered one of the weakest areas of the integrated circuit packaging. The failure modes associated with bonding include [1,7]:

1- Formation of intermetallics. The most common wire bond is made by gold thermo-compression bonding achieved by a ball or wedge bond of fine gold wire to the aluminium contact pad as illustrated in Figure 2.11. Due to Au-Al interdiffusion, intermetallic compounds can form at the bond-land interface. This is known as "purple plague" due to the colour of the compound. The presence of Si acts as a catalyst to the reaction increasing the degradation of the contact. The intermetallic compound is hard and brittle but strong and conductive, and in itself does not undermine the performance of the bond. The presence of voids under the bond tends to weaken the bond strength and increase the resistance. The voids tend to merge together and migrate, causing the bond to lift and failure of the device.

2- Bond looping and lagging. As shown in Figure 2.11 a loop is formed to make a connection between the die and the IC lead frame. If the loop lags too much it may result in short circuits by touching adjacent bond wires. Conversely, if the loop is too tight, the tension created at the heel and neck of the bond and in the wire itself, leads to fracturing and slippage of the bond metal.

3- Bond integrity. Moisture, contamination by foreign particles in general and the contamination of gold by carbon in particular are factors which contribute to a reduction in the integrity of the bond wire.

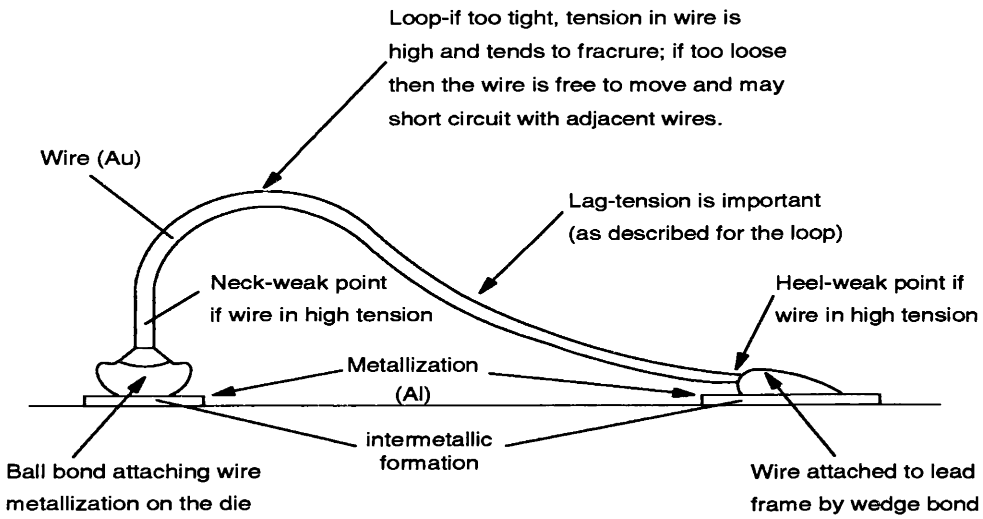

Loop-if too tight, tension in wire is high and tends to fracrure; if too loose then the wire is free to move and may short circuit with adjacent wires.
Wire (Au)
Lag-tension is important (as described for the loop)
Neck-weak point if wire in high tension
Heel-weak point if wire in high tension
Metallization (Al)
intermetallic formation
Ball bond attaching wire metallization on the die
Wire attached to lead frame by wedge bond

Figure 2.11: Typical Wire Bonding in ICs, Showing Weakest Parts of the Bonding

4- Whisker growth. The growth of whiskers at the Al-Si bond pads in order to equalize compressive stresses may lead to short circuits.

5- Backwash in the moulding process. PEDs are susceptible to this mechanism because the plastic compounds used may result in forcing the bond wires against each other and causing short circuits.

6- Bonding pressure. Low bonding pressure can lead to poor fracture strengths of bonds joints. On the other hand, excessive bonding pressure can cause holes or even cracking of the chip, plastic deformation of chip around active areas of devices and the failures discussed in (2), (4) and (5) above.

2.3.4 Die Attachment Failures

The die integrity and corrosion are two failure modes associated with die attachment [1]. The integrity of the die attachment depends upon a proper contact being made between the die and the lead frame. Voids in the die attachment result in detachment or thermal/electrical failure, such as burn-out and parametric shifts, due to the bad contact. Corrosion due to contamination and moisture introduced by the lead frame or the epoxy die attachment system lead to a weak contact.

2.3.5 Particulate Contamination

In hermetic packages loose particles (or debris) in a packaged IC are the cause of particulate contamination failure mode [1]. The presence of conducting particles, such as silicon and gold flakes, in these devices may result in short circuits between metal tracks, burn-outs and shorting of bond wires.

Screening against this mechanism is difficult and expensive. Therefore, improving processing techniques is the practical way to minimise the presence of such particles.

2.3.6 Radiation

Radiation particles affect ICs through the generation of electron-hole pairs in the bulk of the device. Two sources of radiation may affect an IC operation [1]:
1- Extrinsic radiation: i.e. γ-rays, cosmic rays and X-rays
2- Intrinsic radiation: i.e. α-particles radiation, β radiation.

2.3.6.1 Extrinsic Radiation

Extrinsic radiation depends on the environment in which a device is operated (e.g a space probe). Both bipolar and MOS devices are affected by extrinsic radiation [1]. Exposure to γ-radiation leads to an increase in the low-frequency noise level in linear bipolar devices, while it causes latch-up of CMOS devices. X-rays mainly affect MOS devices causing a shift in the threshold voltage as a result of the generation of positive charges in the oxide layer.

2.3.6.2 Intrinsic Radiation

Intrinsic radiations are generated by trace impurities of radioactive elements (e.g uranium and thorium) present in the packaging material. Uranium (U-238) and thorium (Th-232) decay, for example, emit α-particles with energies in the range of 4 to 9 MeV. An average α-particle entering the silicon substrate with an energy of 5 MeV penetrates a depth of 25 μm and generates up to two and a half million electron-hole pairs in a period of several picoseconds [1]. The generation of electron-hole pairs upsets the stored data in dynamic memory and other programming devices, producing 'soft errors'. Soft errors are defined [17] to be random, nonrecurring, single-bit errors in memory devices. The errors are not permanent, i.e., no physical defects are associated with the failed bit. In fact, a bit showing a soft error is completely recovered by the following write cycle with no greater chance of showing an error than any other bit in the device. β-particles produce effects similar to those of α-particles discussed above.

To reduce the effect of radiation the number of particles entering the chip must be reduced, by using organic coating, and maintaining a large critical charge on the circuit nodes, for instance by increasing cell capacitance in memory chips.

2.4 References

[1] E.A. Amerasekera and D.S. Campbell, Failure Mechanisms in Semiconductor Devices, John Wiley and Sons, Chichester 1987.

[2] M.J. Howes and D.V. Morgan, Reliability and Degradation Semiconductor Devices and Circuits, John Wiley and Sons, Chichester 1981.

[3] D.R. Alexander, "Electrical Overstress Failure Modeling for Bipolar Semiconductor Components", IEEE Trans. Components, Hybrids and Manufacturing Technology, Vol. CHMT-1, No. 4, pp. 345-352, December 1978.

[4] K. Bell and A.S. Pinkerton, "The Fatal Spark", IEE Review, Vol. 3, No. 2, pp. 51-53, 20 Feb. 1992.

[5] D.G. Edwards, "Testing for MOS IC Failure Modes", IEEE Trans. Reliability, Vol. R-31, No. 1, pp. 9-16, April 1982.

[6] N.D. Strojadinovic, "Failure Physics of Integrated Circuits - A Review", Microelectronics and Reliability, Vol. 23, No. 4, pp. 609-707, 1983.

[7] F. Fantini and C. Morandi, "Failure Modes and Mechanisms for VLSI ICs - A Review", IEE Proceedings, Part-G, Vol. 132, No. 3, pp. 74-81, June 1985.

[8] E.S. Anolick and G.R. Nelson, "Low-Field Time-Dependent Dielectric Integrity", IEEE Trans. Reliability, Vol. R-29, No.3, pp. 217-221, August 1980.

[9] J. Lange, "Sources of Semiconductor Wafer Contamination", Semiconductor International, pp. 124-128, April 1983.

[10] N.E. Lycoudes and C.C. Childers, "Semiconductor Instability Failure Mechanisms Review", IEEE Trans. Reliability, Vol. R-29, No. 3, pp. 237-248, August 1983.

[11] K.K. Ng and G.W. Taylor, "Effects of Hot-Carrier Trapping in N and P Channel MOSFETs", IEEE Trans. Electron Devices, Vol. ED-30, No. 8, pp. 871-876, August 1983.

[12] E. Takeda, H. Kume, T. Toyabe and S. Asai, "Submicrometer MOSFET Structure for Minimizing Hot-Carrier Generation", IEEE Trans. Electron Devices, Vol. ED-29, No. 4, pp. 611-618, April 1982.

[13] K.V. Ravi, Imperfections and Impurities in Semiconductor Silicon, Jon Wiley and Sons, New York 1981.

[14] J. Partridge, "Testing for Bipolar Integrated Circuit Failure Modes", Proc. IEEE Test Conference, pp. 397-406, 1980.

[15] R.O. Jones, "Developments Likely to Improve the Reliability of Plastic Encapsulated Devices", Microelectronics and Reliability, Vol. 17, No. 2, pp. 273-278, 1978.

[16] T.E. Turner and R.D. Parsons, "A New Failure Mechanisms : Al-Si Bond Pad Whisker Growth During Life Test", IEEE Trans. Components, Hybrids and Manufacturing Technology, Vol. CHMT-5, No. 4, pp. 431-435, December 1982.

[17] T. May and M. Woods, "Alpha-Particle-Induced Soft Errors in Dynamic Memories", IEEE Trans. Electron Devices, Vol. ED-26, No. 1, pp. 2-9, Jan. 1979.

CHAPTER THREE

AN OVERVIEW OF TESTING

This chapter presents an overview of testing digital, analogue and mixed-signal ICs. The various stages of an IC testing are outlined. The various fault models available and the one adopted in this thesis are then discussed. The test pattern generation algorithms used in testing digital ICs are reviewed. The difficulties encountered in testing analogue and mixed-signal ICs, and the proposed solutions in the literature are studied.

3.1 Stages of Integrated Circuit Testing

Testing is a procedure that is applied to an IC to ensure that it performs all of the functions for which it was designed. Ideally, all of the physical abnormalities, such as those outlined in Chapter 2, which cause or likely to cause, an IC to malfunction should be detected. Consequently, testing is meant to detect all faults that may occur in an IC.

A typical IC goes through many stages of testing as illustrated in block diagram in Figure 3.1. During wafer manufacturing, in-line measurements are made to determine if process control parameters such as sheet resistivities are within specified limits. After the wafers are completely fabricated, functional tests and parametric tests are applied to each die. Functional tests, for a digital IC, apply logical values to the IC inputs and compare the output responses with predetermined values. Parametric measurements verify that parameters such as dc voltages and the IC's power consumption fall within specified limits. A number of test chips which contain special purpose structures are usually included on each wafer. These test structures are designed to provide information about various processing parameters and consist of structures such as diodes, contact chains and metal mazes.

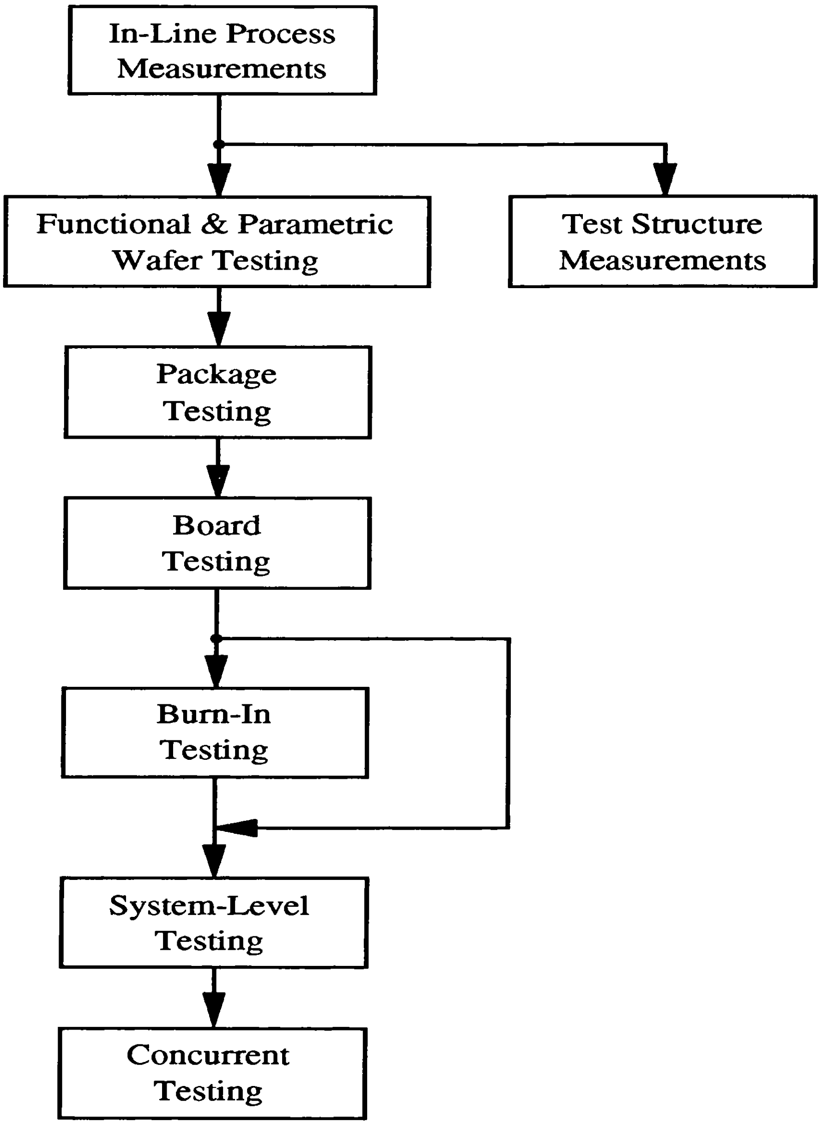

In-Line Process Measurements
Functional & Parametric Wafer Testing
Test Structure Measurements
Package Testing
Board Testing
Burn-In Testing
System-Level Testing
Concurrent Testing

Figure 3.1: Testing Stages of a Typical Integrated Circuit

After wafer testing, the dies that have passed all previous tests are packaged on some type of chip carrier and re-tested. The packages that pass this test can then be mounted on boards. Another functional testing phase at this packaging level takes place which verifies that the chip carriers or dies have been mounted correctly. The boards are then packaged in systems where system-level testing takes place. This type of testing verifies that the boards are correctly mounted and that the entire system can perform its desired function at full speed.

During the operation of a system, additional testing is performed to detect faults that occur as the system is in normal operation. Real-time testing which takes place as the component is performing its function is referred to as concurrent testing. This type of testing may not only detect the presence of faults (called fault detection), but it may also have the capability to mask faults so that a computer error or data corruption does not occur (called fault correction or fault tolerance).

Due to the high cost associated with IC failure occurring during normal operation, an additional testing procedure called burn-in may be applied. This procedure is used to detect reliability failures, i.e. faults that may not initially be detectable, but that later cause system failure. An example of this type of defect is a break due to metal migration in which an open circuit due to a migration of metal caused by a high current density. During burn-in, high temperature and high voltage environmental conditions are applied to the IC to accelerate the phenomena causing such defects. An additional testing phase is applied immediately after burn-in to detect these types of defects.

It should be pointed out that each one of the testing phases described above, is applied to achieve different goals. Consequently, each phase may have an incompatible testing procedure that may cause some dies to be called fault-free at one level of assembly and then faulty at another level due to different fault coverage of the applied testing techniques.

An important final point is the cost of testing in relation to the level of assembly. It was estimated in [1] that the cost of detecting a fault increases 10 folds with each increase in assembly level. For example, if it costs 10 units to detect a fault at the chip level, then it would cost 100 units to detect the same fault when it was embedded at the board level. Therefore, faulty ICs that escape detection at a particular level of assembly will be much more expensive to detect at the next level.

3.2 Fault Models

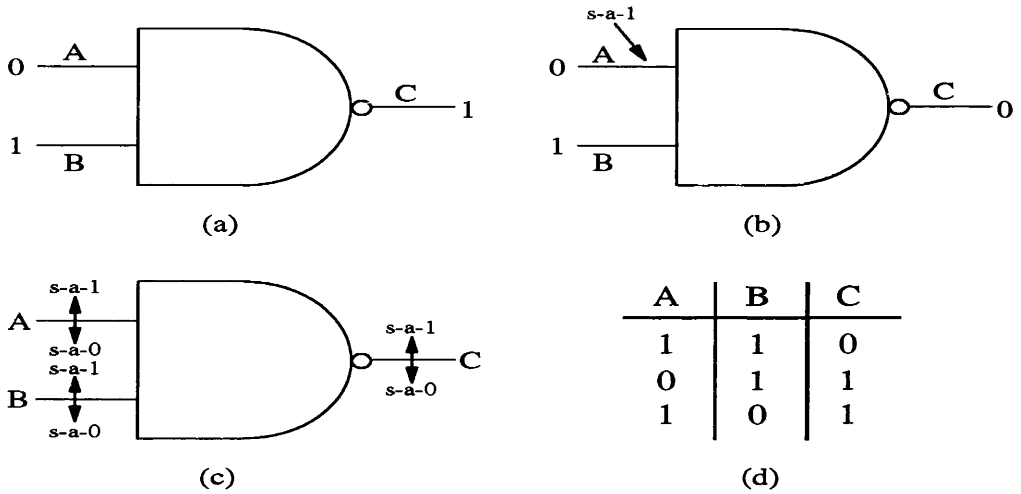
Failure mechanisms, as described in chapter 2, may cause a wide variety of faulty circuit behaviour which is technology, layout and process dependent. To represent the behaviour of defective integrated circuits, fault models are used.

Fault models serve two purposes during the testing process. First, they help generate tests as described in section 3.3. Second, they help evaluate test quality defined in terms of coverage of modelled faults. This section describes the stuck-at, stuck-open, stuck-on, bridging, layout-driven and transistor based fault models. The first four models were specifically derived for digital integrated circuits.

3.2.1 Stuck-At Model

The most universally accepted fault model for representing defective digital circuit behaviour in the classical approach to IC testing is the single stuck-at model [2,3]. It operates at a Boolean-gate level of abstraction. The stuck-at model assumes that all defects manifest themselves as a permanent logical value of 0 or 1 on a logic gate input or output. It also assumes that only one fault may occur per circuit.

A typical application of the stuck-at model is illustrated by the simple example shown in Figure 3.2. In this example, Figure 3.2a depicts a fault-free 2-input NAND gate with input A set on 0 and input B set on 1 resulting in a 1 on output C. Figure 3.2b shows the NAND gate with input A stuck-at-1 (s-a-1) and the same input pattern of 01 applied to inputs A and B respectively. Since input A is s-a-1 the gate perceives the A input as set at 1 irrespective of the actual input being applied, hence it performs as NAND with the 1 applied to input B resulting in the faulty logic value of 0 at output C. Therefore, the input pattern 01 applied to A and B respectively, is considered a test because the response of the fault-free gate is different from that of the faulty one. If they had the same response then that pattern would not have constituted a test for A s-a-1 fault. The six potential stuck-at faults are shown in Figure 3.2c, and the input test sequence that will cause the gate's output node to have a different logical value in a fault-free circuit from a faulty circuit with any single stuck-at fault is depicted in Figure 3.2d.



A	B	C
1	1	0
0	1	1
1	0	1

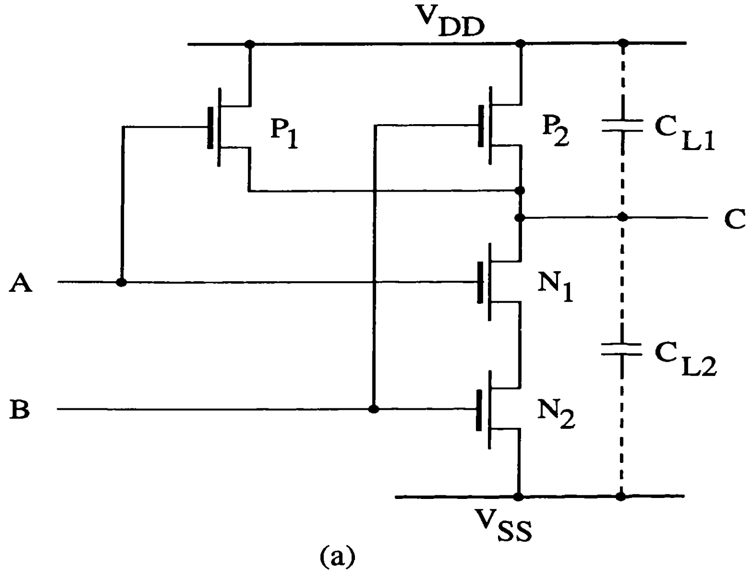
Figure 3.2: Testing for Stuck-at Faults in a NAND Gate. (a) Fault-Free Gate, (b) Gate with A s-a-1, (c) The Six Possible Stuck-at Faults, (d) Test Pattern to Detect all Six Stuck-at Faults

The stuck-at fault model described above, was first proposed for dealing with the early DTL (diode-transistor-logic) and RTL (resistor-transistor-logic) logic-circuit families when discrete components were used. It was applied with great success to printed-circuit-boards (PCBs) loaded with small and medium scale TTL (transistor-transistor-logic) components. The success and popularity of the model are mainly due to its simplicity and its representation of the likely physical defects in a PCB. For example, a floating input of a TTL gate exhibits a s-a-1 behaviour and a short of a line to ground by a splash of solder will be s-a-0.

However, some of the faults that are likely to occur in modern MOS VLSI circuits exhibit behaviours that cannot be modelled by the stuck-at model. The subsequent subsections will discuss some of the fault models devised for MOS circuits.

3.2.2 Stuck-Open Model

The stuck-open fault model [4,5] assumes that a defect can cause a MOS transistor to be permanently in the non-conducting state. This type of fault is particulary difficult to detect since a combinational network may behave sequentially due to the presence of a memory element. Therefore, the test pattern derived using the classical stuck-at model for the combinational network are no longer effective in testing the network in the presence of stuck-open faults.



(a)

A	B	C
1	1	0
0	1	1
1	0	1

(b)

A	B	C
1	1	0
0	1	1
1	1	0
1	0	1

(c)

Figure 3.3: Test for Stuck-Open Fault in CMOS NAND Gate. (a) NAND with P2 Stuck-Open, (b) Stuck-at Test Sequence, (c) Test Sequence to Detect all Stuck-at & Stuck-Open Faults

To illustrate the stuck-open model and the sequential behaviour, consider the 2-input NAND gate implemented in CMOS technology as shown in Figure 3.3a, and assume that there is a defect causing transistor P2 to be stuck-open. The capacitors

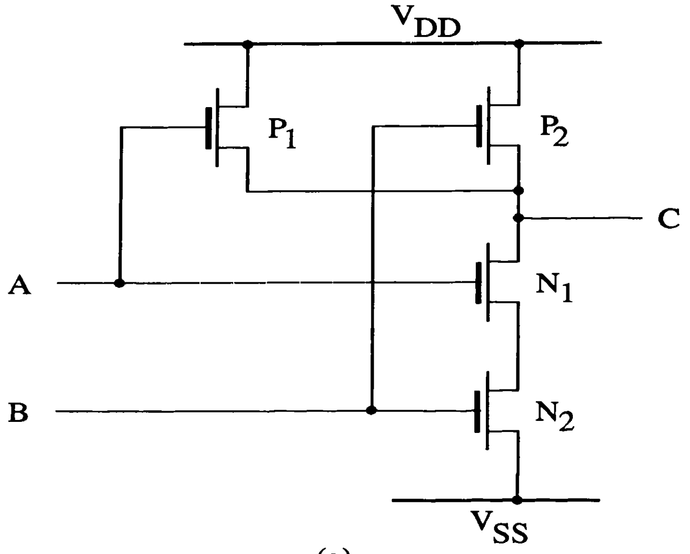
C_{L1}, and C_{L2} represent capacitive loads which result vhen connecting the output to the input of another CMOS gate. The capacitors C_{L1} anl C_{L2} are insignificant for simple circuit analysis in a fault-free network, but they pla a crucial role in the presence of a stuck-open fault. If the stuck-at test sequence shown in Figure 3.3b is applied to this network, the output at C will have an incorrect logical value in the presence of any stuck-at fault. But, the stuck-open transistor P2 will not be detected since the logical value 1 on the output node in response to test vector 2 will be retained for test vector 3 due to the capacitances on the output node Therefore, the sequence of the applied vectors is also important. Figure 3.3c shows a sequence that will detect all stuck-open faults in the 2-input NAND gate. Test vector (11) in Figure 3.3c ensures that the output is initialised to 0 irrespective of the fault.

3.2.3 Stuck-On Model

Similar to the stuck-open fault is the stuck-on fault which assumes that a defect may cause a transistor to be permanently in the fully conducting region [4,6,7]. Any gate-oriented testing approach cannot assume that this type of fault is detected since the logical value at the gate output is dependent on the ratios of the transistors when the source and drain terminals of the faulty transistor are connected to opposite power supplies. For example, assume that transistor N2 shown in Figure 3.4a is permanently conducting and that all transistors have a W/L (Width/Length) ratio of 12/3. The output of the gate of Figure 3.4a will not achieve a correct output response for vector 1 as shown in Figure 3.4b. Now consider the same circuit except that the n-channel transistors (N1 & N2) have a W/L ratio of 18/3. For this circuit, the output will perform correctly as shown in Figure 3.4c. Although the second circuit may functionally perform correctly, it may have an additional circuit delay that can cause component failure.

3.2.4 Bridging Fault Model

A bridging fault occurs when two different nodes in a network are connected by a defect [8,9]. For a circuit having a bridging fault, the logical values of the connected nodes are a function of the state of the circuit, the resistance of the bridge and the characteristics of the circuit such as transistor sizes and line resistances. Since both of the connected nodes may be driven high or low simultaneously, they can switch between the high and the low logical values (e.g. a value less than 5 volts and greater than 0 volts) and this fault is not well-modelled as a stuck-at fault. When the connected nodes are driven to opposite potentials, the voltage of the nodes is difficult to determine. Hence, the bridging fault model is not easily adapted to gate-level test generation methodologies which do not model the circuit at the transistor level [9].



(a)

A	B	C
1	1	1
0	1	1
1	0	1

(b)

A	B	C
1	1	0
0	1	1
1	0	1

(c)

Figure 3.4: Test for Stuck-on Fault in CMOS NAND Gate. (a) NAND Gate with N2 Stuck-on, (b) When W/L = 12/3 for all Transistors, (c) W/L = 12/3 for P-Transistors and W/L = 18/3 for N-Transistors

3.2.5 Layout Driven Fault Modelling

This approach is an attempt to improve the accuracy of fault modelling and to derive realistic fault models [10-12]. In [10] a procedure called inductive fault analysis is described, where given an IC layout a fault model and a ranked list of the likely faults is automatically generated taking into account the technology, layout, and process characteristics. The procedure consists of three major steps:

1- Defect generation and analysis,
2- Defect-to-fault translation,
3- Fault classification and ranking.

In step-1, the probable physical defects are generated from an IC layout using known statistical information about defects. The information can be obtained from an actual fabrication line or from published data. After analyzing the significant defects in step-1, the circuit-level behaviour caused by these defects are extracted in step-2. The extracted faults are then classified and ranked according to their likelihood of occurrence in step-3.

Historically, all faults have been assumed to be equally likely to occur, but the fault ranking procedure described above can help to improve the accuracy of testability analysis by demanding the generation of new and more effective test sets. From the classical stuck-at fault modelling and test generation view point, however, this approach is computationally expensive to perform and the extracted faults are not always compatible with conventional test generators.

3.2.6 Transistor Based Fault Model

From the review of mechanisms of failure in Chapter 2, and the study of integrated circuit yield [13,14], faults in an IC basically fall into two categories:
1- Catastrophic faults (Hard faults)
2- Parametric faults (Soft faults).

Catastrophic faults are random defects, which cause structural deformations leading to hard failures such as shorts and opens, in an IC component. Examples of random defects include over or under etching of various layers, oxide pinholes, spot defects, and photolithographic errors. Spot defects, for example, are geometrical features that occur during the manufacturing process that were not originally defined by the IC layout. The main source of spot defects are lithography spots, which are caused by the presence of contaminants like dust particles, on the surface of the mask and photoresist layers. Figure 3.5. shows how the presence of a spot defect results in a missing gate oxide, leading to a short between the gate and source of a MOS device.

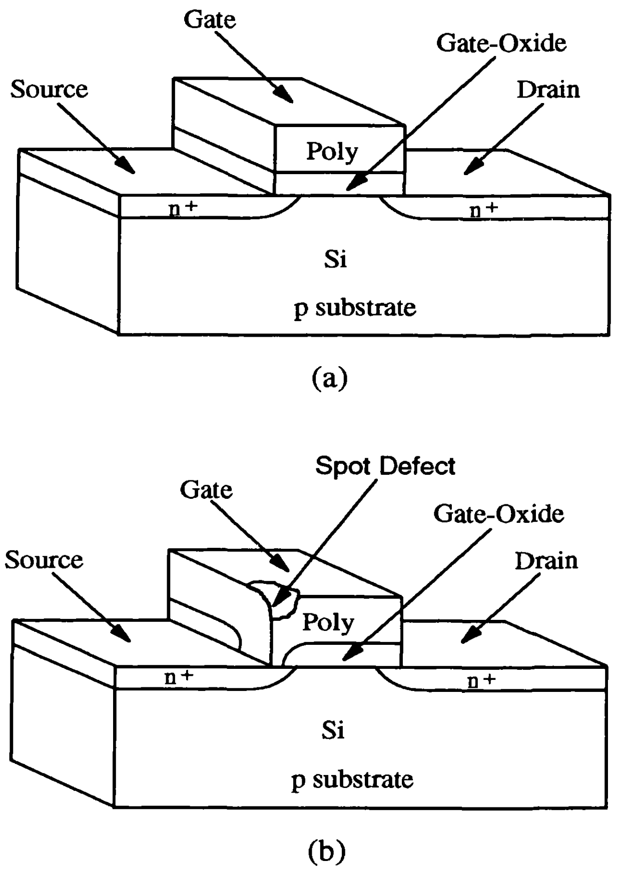


Figure 3.5: The Effect of a Spot Defect. (a) Fault-Free MOS Transistor, (b) MOS Transistor with Gate-Source Shorted Due to Spot Defect

Parametric faults are excessive statistical variations in the manufacturing process conditions, such as a turbulent flow of gasses and inaccuracies in the control of furnace temperature, which cause a soft failure of components of an IC. A soft failure is one which is not sufficient to result in a completely malfunctioning IC, but sufficient to cause performance to deviate outside the limits of the allowable tolerance region. An example of a parametric fault, is a deviation in the width-length ratio of a transistor causing the gain of the device not to meet the specifications.

Our focus is the development and use of a fault model to efficiently detect catastrophic faults during production testing. Developing a model for parametric faults, especially for analogue circuits, is difficult and would result in a complex model due to the multitude of such faults. In practice, parametric faults are usually screened out during a final test, which is often necessary to determine the value of each performance parameter.

Testing of analogue and mixed-signal circuits, requires a model that would be compatible with both analogue and digital functions. The fault models described previously, with the exception of the layout driven one, are for digital ICs and are based on either a gate-level (stuck-at) or a switch-level (stuck-open, stuck-on). Hence, they are not suitable for analogue circuits due to their dependence on transistor behaviour in the linear region. As for the layout driven fault model, it is complex, requires statistical process information, and specialized simulations tools.

Therefore, the simple fault model illustrated in Figure 3.6 is adopted in this thesis to evaluate the applicability of the testing techniques, to be developed in Chapters 4 and 5, for analogue and mixed-signal ICs. The model is a physical one at the transistor level, it synthesizes the most likely catastrophic faults in a MOS transistor based on the work reported in [15-17]. Since the adopted model is a physical one, it is compatible with both analogue and digital circuits, and can be simulated using a transistor level circuit simulator such as HSPICE [18].

The MOS fault model in Figure 3.6 assumes that multiple faults are very unlikely. Further, it indicates that the likely single faults are one of the following: drain-open, source-open, gate-drain-short, gate-source-short, and drain-source-short. These faults are caused by open circuits in the diffusion and metallization layers, and short circuits between adjacent diffusion and metallization layers. No probabilities are associated with each fault, i.e. it is implicitly assumed that the source contact open of one MOS transistor is as likely as, say, the gate-drain-short of some other MOS transistor. Table 3.1 summarizes the faults depicted by Figure 3.6 and the states of the switches to synthesize each fault.

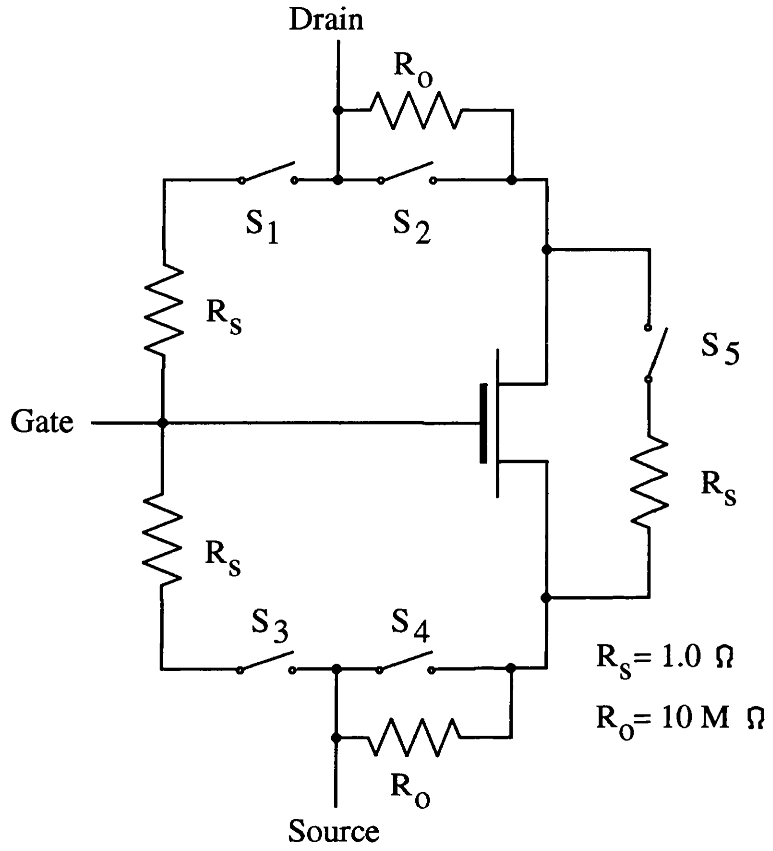


Figure 3.6: Physical MOS Transistor Fault Model

Table 3.1: The Likely MOS Faults and Status of Fault Model Switches

MOS Device Failure	Status of Fault Model Switches				
	S1	S2	S3	S4	S5
Drain Contact Open	OFF	OFF	OFF	ON	OFF
Source Contact Open	OFF	ON	OFF	OFF	OFF
Gate-Drain Short	ON	ON	OFF	ON	OFF
Gate-Source Short	OFF	ON	ON	ON	OFF
Drain-Source Short	OFF	ON	OFF	ON	ON

3.3 Test Generation Algorithms for Digital Circuits

The objective of test generation algorithms is to deduce a minimum set of test vectors that achieve a high fault coverage at an affordable cost. To generate the set of test vectors a fault model is used to mimic the behaviour of the defective circuit, and derive the appropriate input stimulus and output response to detect such behaviour. The majority of test generation algorithms, both in the literature and used in practice, use the stuck-at fault model.

This section discusses 3 digital test generation algorithms:

1- D-algorithm (Multiple-path sensitization)
2- Boolean Difference
3- Switch-level

The test generation algorithms in this section and most of those used in practice are devised to detect a single fault. The reason is that [19] there are 3^{u+v} -1 possible multiple stuck-at, stuck-open and stuck-on faults in a MOS circuit containing u distinct lines and v distinct transistors in which signals may fail. This results in a great increase in the computation time for test generation and fault simulation, even for comparatively small circuits. Therefore, the primary objective of all the testing techniques to be investigated in this thesis will be to detect single fault conditions.

3.3.1 D-Algorithm

The most popular method for generating multiple-path (or n-dimensional path) sensitization is due to Roth's D-algorithm [2]. This is based on the algebra of D-cubes, where D stands for the discrepancy between the faulty and fault-free behaviour. The algorithm generates vectors which will cause a logical difference at one of the ICs outputs for each of the possible stuck-at faults. The D-algorithm is valid for non-redundant combinational logic circuits only. To apply the algorithm to sequential logic the circuit may need to be modified as discussed in section 3.4.

For a given stuck-at fault, the D-algorithm consists of four steps:

1- Fault Excitation: The inputs are conditioned such that the line (or node) to be tested is driven to a logical value opposite of that produced by the fault. As an example, in Figure 3.7, the fault E s-a-0 is excited by setting line E to 1, which requires both inputs A and B to be 1.

2- Fault-Effect Propagation: In this step the fault-effect is propagated closer to a primary output, by conditioning the appropriate gates along the path. For example, to propagate the effect of the fault E s-a-0 in Figure 3.7 to line H it is necessary to set line G to 0.

3- Line-Value Justification: The implication of the gate values assigned in step-2 is propagated backwards to the circuit primary inputs. If a contradiction if found during this backwards propagation, backtracking must be performed and step-2 repeated using an alternative path to propagate the fault effect to a primary output. In the example in Figure 3.7, line G was set to 0 but not justified during step-2. To justify it the primary input on line C and the internal line F must be set to 0. Setting line F to 0 is consistent and justified, because primary input B was already set to 1 in step-1.

4- Line-Value Implication: The operations described in steps 1 to 3 above are carried out incrementaly and generally involve specifying one or more line values. The effect of such specification may ripple through in the forward direction by implication. For instance, setting one of the inputs of a NAND gate to logical 0 would force the gate output to logical 1.

The algorithm outlined in the steps above uses D and $\bar{D}$ symbols to generate compact gate-level models called primitive D-cubes of failure (Pdcf). These D-cubes represent the necessary conditions to propagate a fault. In a pdcf the D symbol represents logical 1 in the fault-free circuit and logical 0 in the faulty circuit, $\bar{D}$ represents the opposite logical values. Figure 3.8 depicts the pdcf's of some basic logic gates. The D-algorithm guarantees finding a test for a given stuck-at fault if such a test exits.

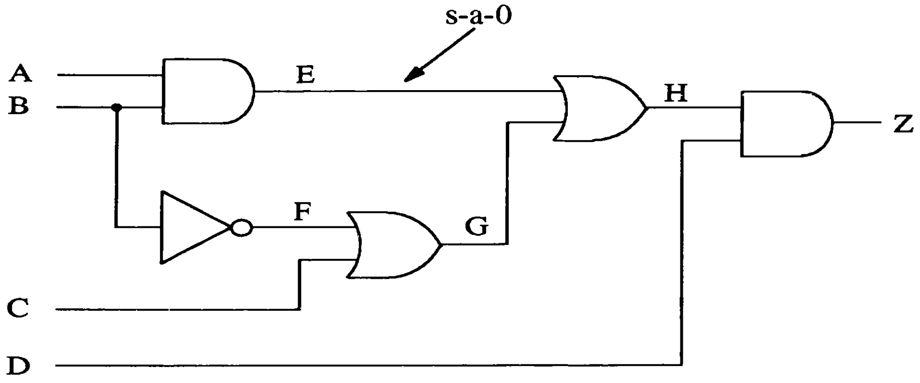
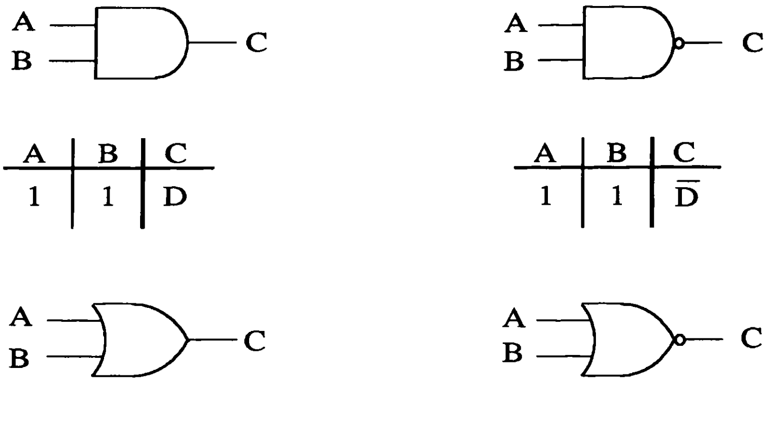


Figure 3.7: A Combinational Logic Circuit with Line E s-a-0



A	B	C
1	1	D

A	B	C
1	1	$\bar{D}$

A	B	C
0	0	$\bar{D}$

A	B	C
0	0	D

Figure 3.8: Primitive D-Cube of Failure (pdcf) for Basic Logic Gates

A problem of the D-algorithm is that there are many possible paths for which the D or $\bar{D}$ may be propagated and the path chosen is Step-2 above is arbitrary. Hence, the backtracking effort needed due to the choice of the wrong path may be significant. As an improvement to the D-algorithm, PODEM (path oriented decision making) [20] was proposed to minimize the amount of backtracking needed. PODEM uses a branch and bound technique for test generation. It repeatedly assigns input values and determines the effect on the fault-under-test until either a test vector is generated or the fault found to be untestable. PODEM implementations typically run an order of magnitude faster than the D-algorithm for most circuits.

FAN is another algorithm that is a further efficiency enhancement to the D-algorithm [21]. It performs extensive analysis of the circuit connectivity in a preprocessing step to minimize backtracking. The enhancements that were implemented in PODEM and FAN are heuristics, and therefore test vectors for most of the faults can be found more quickly, but the worst-case search times are identical. In addition, these algorithms cannot determine which faults are untestable until all possible paths have been searched.

3.3.2 Boolean Difference

An alternative method to test generation in combinational circuits is the Boolean difference method (Boolean partial derivatives) [22]. This mathematically elegant method defines the logical behaviour of a logic circuit as a Boolean function defined by the state of its primary inputs. It then uses a Boolean form of differential calculus to derive the tests necessary to detect a specific stuck-at fault. Assume that the Boolean expression given by

$$Z = f(X_1 , X_2 , \dots , X_k , \dots , X_n) \quad , \; i = 1, 2, \dots,n$$

defines the function of the fault-free circuit, where X_i are the primary inputs to the circuit. If input X_k is stuck-at fault, then a new function Z_k is defined as

$$Z_k = g(X_1 , X_2 , \dots ,\overline{X}_k , \dots , X_n)$$

which is formed by replacing X_k by $\overline{X}_k$. The Boolean Difference is defined as

$$\partial Z/\partial Z_k = Z \oplus Z_k = h(X_1, X_2, \ldots, X_n)$$

where $\oplus$ is the exclusive-OR operation. As an example, consider the logic circuit in Figure 3.9. The fault-free Boolean function is

$$Z = X_1 X_2 + \bar{X}_2 X_3$$

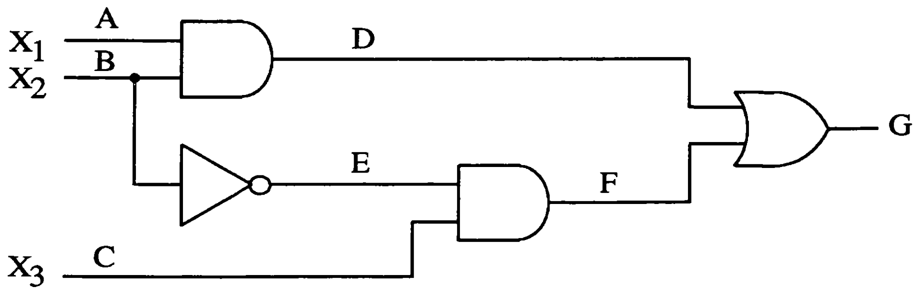


Figure 3.9: A Combinational Logic Circuit with Line A Stuck-at Fault

If line A has a stuck-at fault, then Z_{X1} is

$$Z_{X1} = \bar{X}_1 X_2 + \bar{X}_2 X3$$

$$\begin{aligned}\partial Z/\partial Z_{X1} &= Z \oplus Z_{X1} \\ &= (X_1 X_2 + \bar{X}_2 X_3) \oplus (\bar{X}_1 X_2 + \bar{X}_2 X_3) \\ &= X_1 X_2 X_3 + X_1 X_2 \bar{X}_3 + \bar{X}_1 X_2 \bar{X}_3 + \bar{X}_1 X_2 X_3\end{aligned}$$

The four minterms in $\partial Z/\partial Z_{X1}$ above, define the full set of input tests that will detect both types of stuck-at faults (i.e. s-a-1 or s-a-0) on line A. The function $\partial Z/\partial Z_{X1}$ can be partitioned into two separate lists to identify the tests that detect s-a-1 and s-a-0 faults. This is achieved by separating the list of all tests into those containing X_k and those containing $\bar{X}_k$, where the former will require a 1 on X_k and therefore test for X_k s-a-0, and similarly the latter will test for X_k s-a-1. In the example above, separating the terms of $\partial Z/\partial Z_{X1}$ gives

$$[X_1 X_2 X_3 \, , \, X_1 X_2 \bar{X}_3]_{A \text{ s-a-0}}$$

and

$$[\bar{X}_1 X_2 \bar{X}_3 \, , \, \bar{X}_1 X_2 X_3]_{A \text{ s-a-1}}$$

The above test generation method can be extended to derive tests for faults on internal (i.e. non-primary) circuit lines. This is achieved by representing the original circuit function as $Z = F(X_1 , \ldots , X_n , f_k)$ where f_k is dependent on $(X_1 , \ldots , X_n)$ and represents the Boolean function at the internal line to be tested. The partial derivative of $Z = F(X_1 , \ldots , X_n , f_k)$ in terms of f_k leads to the required tests. The Boolean difference method discussed above enables the generation of test sequences for both single and multiple faults [22], and the propagation of a fault to a particular circuit output. However, the method is limited to small circuits due to the amount of algebraic computation involved and the high storage required. Its main advantage lies in identifying essential tests since once these are known other more efficient methods, such as the D-algorithm, can be used to determine all other faults covered by them.

3.3.3 Switch-Level Test Generation

In MOS circuits two types of faults are likely to occur. These are the stuck-at faults (s-a-1 and s-a-0) at the gate terminals, and the transistor (stuck-on and stuck-open) faults. The D-algorithm and the Boolean difference test generation methods, however, are only capable of generating tests for gate level stuck-at faults. A switch-level test generation algorithm, that can handle both types of faults in MOS circuit was proposed in [23-25]. This algorithm transforms the transistor structure into an equivalent logic gate structure. To model the memory state that may result from a transistor fault, one new logic gate is introduced. If the logic gate equivalent circuit is combinational, the D-algorithm can be applied to generate the necessary tests. However, if the circuit has memory elements (i.e. sequential), an initialization procedure is added before the D-algorithm resulting in two-pattern test sequences.

To explain the switch-level algorithm, consider the CMOS NOR gate implementation in Figure 3.10(a) [24]. The two inputs, A and B, are connected to the gate terminals of the transistors. Depending upon the logical value of an input signal (i.e. 0 or 1), the corresponding transistor behaves like a perfect switch (i.e. open or short). To set the output "High" a conducting path is created between the output and V_{DD} by closing both P-MOS transistors. Similarly, the output is set "Low" by creating a path to V_{SS}, which is achieved by closing at least one of the N-MOS transistors. This operation is represented by the switch model in Figure 3.10(b). The status of switches S_N and S_P (i.e. open or closed) is a function of the inputs A and B. In a CMOS circuit, the states of S_N and S_P are complementary resulting in only one switch being closed at a time. The open switch presents a floating or high impedance state to the output node. If, however, both S_N and S_P are open, then the output node will retain its previous value. This is due to the charge retention capability of an isolated node in a MOS network. In the opposite situation, if both S_N and S_P are shorted then the output will be a 0 or a 1 depending upon the relative resistances of the load and the driver transistors in their conducting states.

Figure 3.10(c) shows the logic gate model of the transistor circuit in Figure 3.10(a). The transformation to a gate model that produces the switch functions S_N and S_P in Figure 3.10(b), is achieved by applying the rules in Table 3.2 [23]. The logic gate model in Figure 3.10(c) consists of conventional logic gates and one additional block called B-block. The B-block models the high impedance (or memory) state. The signal S_P, which is the output of the path from V_{DD}, should be connected to B-block terminal marked P. Conversely, the signal S_N which is the output of the path from V_{SS}, should be connected to the terminal marked N. The symbol M, in the table describing the function of the B-block, refers to the memory or previous state (before S_N and S_P changed to 0) of the output line.

To demonstrate the test generation process, lets assume that transistor N1 is stuck-open. To excite the fault in Figure 3.10(c), input A is set to 1 and to propagate it through gate G2 input B is set to 0. This forces the normal values to the P and N inputs of the B-block as 0 and 1, respectively, so the fault-free output is 0. Under the fault, only input N complements, which means both inputs of the B-block are zero. The output M means that it could be either 0 or 1 depending on the previous value.

Thus, to detect the fault, the previous output should be initialised to 1. The complete test is, therefore, a two-vector test: the first vector (0,0) properly initialises the output, then the second vector (1,0) produces differentiated outputs for the fault-free and faulty cases.

Due to internal circuit delays, the two-pattern test sequence approach discussed above, can be invalidated because temporary internal node states may charge the gate output to the correct logical level through alternate paths. Robust two-pattern test vectors can be generated that do not have this problem [25].

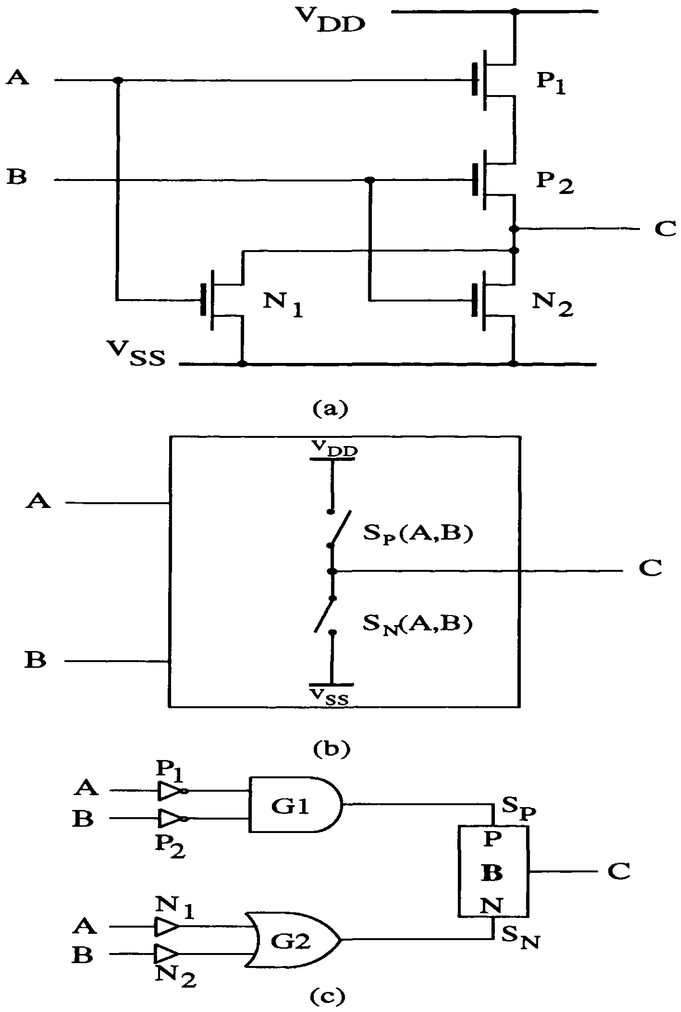
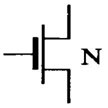
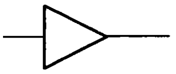
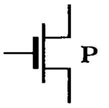
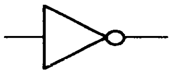
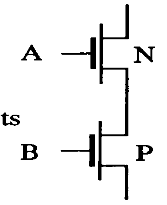
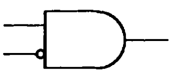
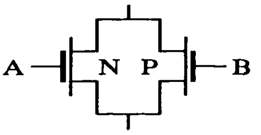
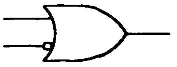
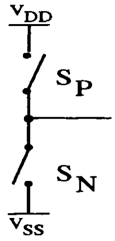
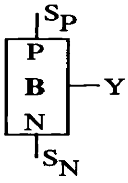


Figure 3.10: MOS Transistor Switch-Level Test Generation. (a) CMOS NOR Gate Transistor Structure, (b) NOR Gate Switch Model, (c) NOR Gate Logic Gate Model

Table 3.2: MOS Transistor to Logic Transformation

MOS Transistor Structures	Equivalent Logic Gates
NMOS 	
PMOS 	
Series Elements 	
Parallel Elements 	
MOS Gate Output 	S_P S_N Y 1 1 0 1 0 1 0 1 0 0 0 M 

3.4 Design for Testability of Digital Circuits

Design for testability (DFT) refers to any design change that enhances the test generation and test application procedures. The key concepts underlying all considerations for DFT are: controllability and observability. Controllability is the ability to set and reset every internal circuit node. Observability is the ease of being able to observe the state of any internal node at the primary outputs.

The term testability represents the relative ease of test generation for a given IC design. Given the circuit structure, testability analysis programs such as SCOAP

[26] and ITTAP [27] can identify the circuit nodes that are difficult to control and observe. Hence, any potential testing problems can be identified early in the design phase, allowing modifications by introducing DFT techniques to improve the final testability of the circuit.

An important objective of introducing DFT techniques to a network is to enable the testing of sequential circuits. This is achieved, as will be explained in the following sub-sections, by transforming a sequential circuit to a combinational one, thereby allowing the application of a test generation algorithm like the D-algorithm.

The DFT techniques used in practice to enhance the testability of digital circuits fall into three categories [1]:

1- Ad-hoc techniques
2- Structured techniques
3- Built-in self-testing

The three DFT categories and examples of the prominent techniques that fall under each categories are described in the subsections below.

3.4.1 Ad-hoc techniques

Ad-hoc DFT techniques are those techniques which can be applied to solve a problem for a given design. They are not generally applicable to all designs, and are not directed at solving the general sequential problem [1]. Two of the DFT techniques that fall in the ad-hoc category will be discussed in the following subsections; partitioning and test point insertion.

3.4.1.1 Partitioning

Partitioning techniques [1] adopt a divide-and-conquer policy, where large circuits are divided into smaller circuits or modules which can be tested in isolation.

An example of partitioning using multiplexers is illustrated in Figure 3.11. In this example the two control signals S_1 and S_2 allow normal operation, with connection from module A to module B and vice versa, inter-module signals monitoring at the primary outputs of the other module, and testing A and B separately. Multiplexers are also used to break an overall feedback path as shown in Figure 3.12, hence enabling the application of test signals.

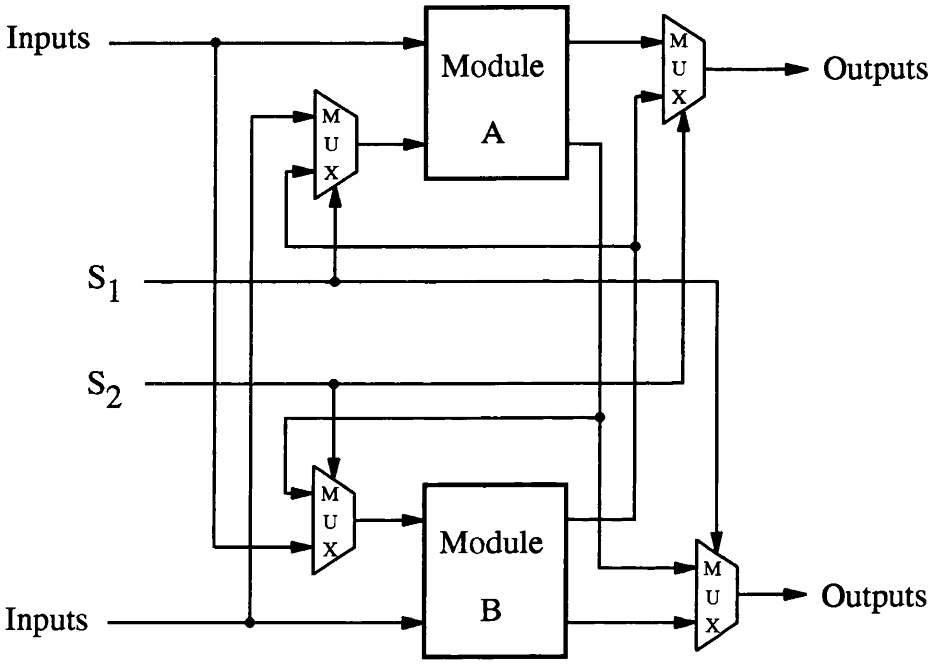


Figure 3.11: The Use of Multiplexers for Partitioning

The presence of a free-running oscillator (clock generator) on a circuit board makes testing it extremely difficult if not impossible. This is due to the great difficulty in synchronizing the tester to the activities of the circuit board. To overcome this problem the oscillator can be degated as depicted in Figure 3.13. The degating logic allows the application of an external test clock, that can be controlled by the tester to provide a more controlled test environment.

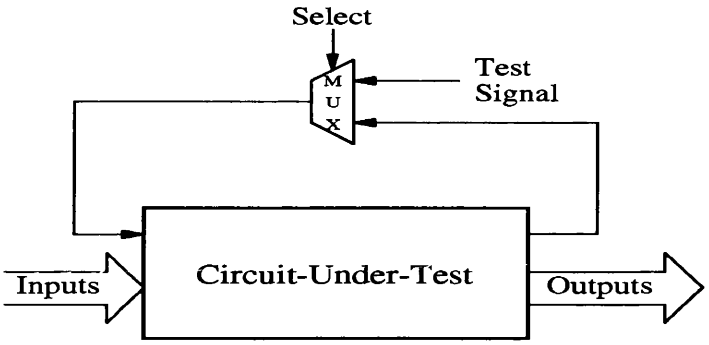


Figure 3.12: The Use of Multiplexers to Break a Feedback Path

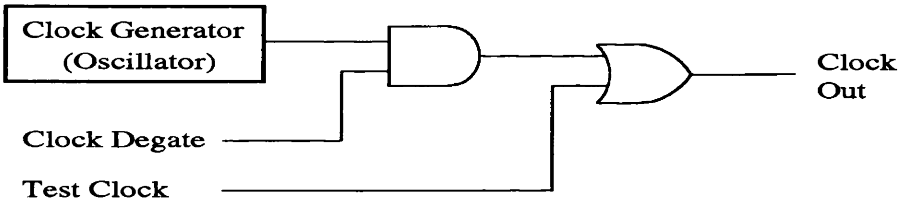


Figure 3.13: Degating a Clock Generator

3.4.1.2 Test Point Insertion

Inserting test points to make certain internal nodes accessible enhances the network testability. The test points can be used as primary inputs, outputs or both.

In Figure 3.14 a degating function is used to control the three output lines connected to the extra pins (i.e. test points). When degating is enabled the extra pins can be used as primary inputs to Module-2, hence improving its controllability. On the other hand, when degating is disabled the extra pins can be used as primary outputs, to observe the output nets of Module-1 connected to them. Therefore, controllability and observability were enhanced, in this example, by inserting extra test points and degating.

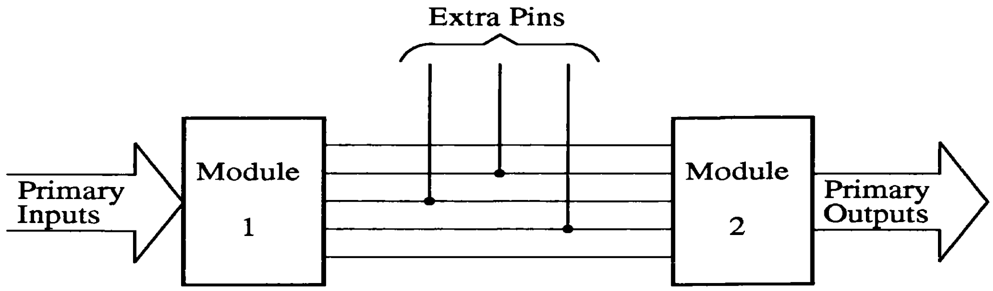


Figure 3.14: The Use of Test Points as Both Inputs and Outputs

3.4.2 Structured Techniques

The ad-hoc techniques discussed above are often introduced to the design as an afterthought to solve the testing problem of that particular network. Structured techniques for DFT, however, are generally applicable formal methods, that are introduced during the formulation of the design, in order to ensure that the basic architecture of the network will facilitate easier testing.

The objective of all structured techniques is to facilitate the testing of complex sequential networks, by enhancing the controllability and observability of their state variables. In essence, then, the complex task of testing a sequential machine is transformed to the simpler task of testing a combinational machine.

A number of structured DFT techniques are used in practice. Of these, level-sensitive scan design (LSSD) and boundary-scan are prominent and will thus be discussed in the subsections below.

3.4.2.1 Level-Sensitive Scan Design

Level-Sensitive Scan Design (LSSD) [1,28] is one of the best known and the most widely practiced methods for synthesizing testable logic circuits. The "level-sensitive" aspect of the method means that a sequential network is designed so that the steady-state response to any state change is independent of the dynamic characteristics of the logic components, such as rise and fall times and propagation delays, within the network. Also if a state change involves the changing of several signals, the response must be independent of the order in which they change. These conditions are ensured by the enforcement of certain design rules, particulary pertaining to the clocks that control state change in the network. "Scan" refers to the ability to shift into and out of any state of the network.

The key component in the LSSD method is the Shift-Register Latch (SRL) shown in Figure 3.15, which is used to implement all the storage elements in the network. The operation of the SRL is independent of the ac characteristics of the clock, and requires only that the clock is held high long enough to stabilise the feedback loop, before being returned to the low state. The D and C lines in Figure 3.15 form the normal mode memory function, while lines I, A, B and L2 comprise additional circuitry for the shift register function.

In a network that implements the LSSD techniques, the SRLs can be threaded by connecting the output line L2 to the scan-in line I, resulting in a serial shift register called a "Scan Path". The operation of the scan path is controlled by the two-phase (non over lapping) clocks on lines A and B. The scan path enables access to each storage element in the circuit, and hence control and observation of the internal states.

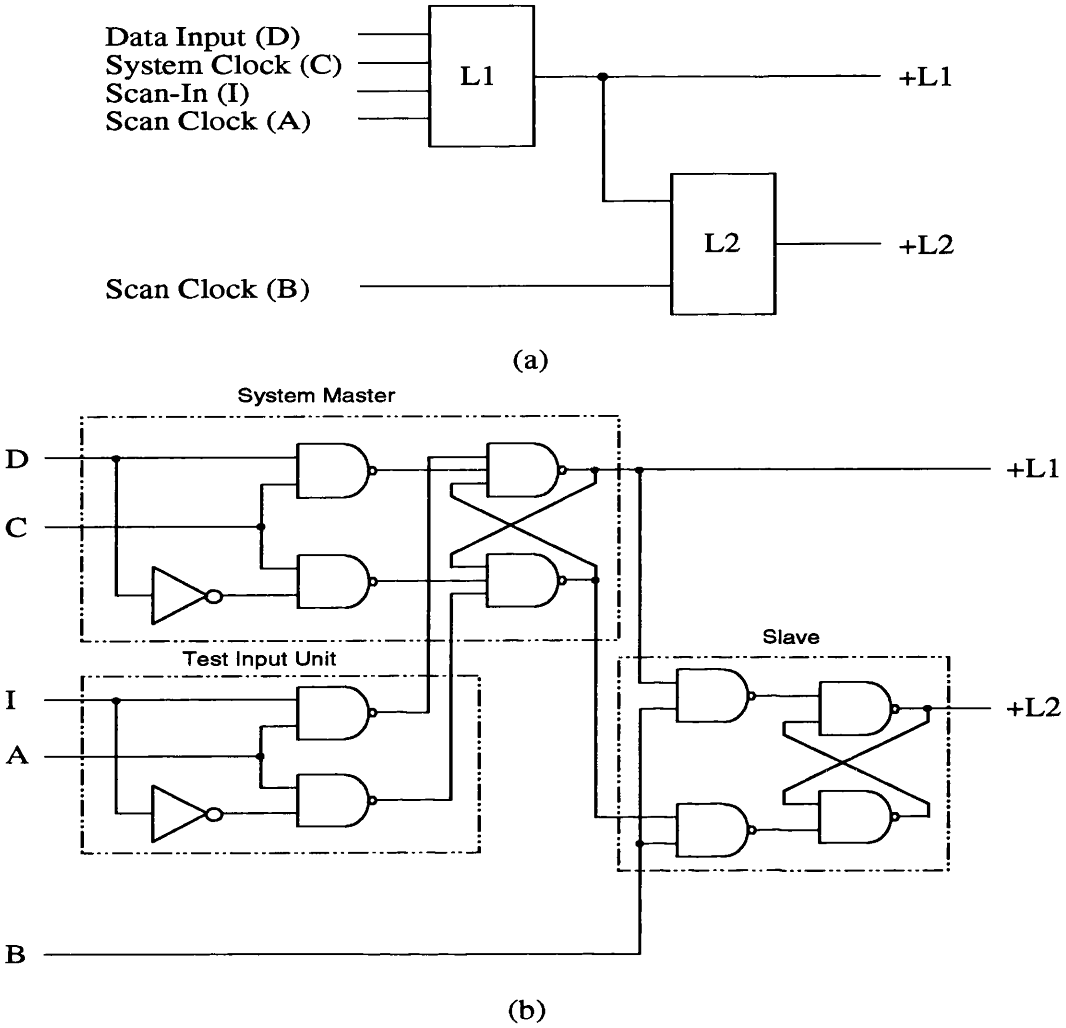


Figure 3.15: The Shift-Register Latch (SRL). (a) Symbolic Representation, (b) Logic Implementation

Figure 3.16 illustrates a general structure for an LSSD system. To test such a system the following steps are performed [28]:

1- Simple "flush" and "shift" tests are applied to ensure that the SRLs are operating as a scan path.

2- The circuit is switched to test mode and the scan-in port is used to load a test pattern to the SRLs. Clocks A and B are pulsed to shift the pattern through the elements of the scan path.

3- The circuit is switched to normal system operation mode and clock C is pulsed on then off. By doing so the combinational sub-systems response gets stored in the L1 latches of the SRL. Clock B is then pulsed to duplicate the values of L1 in the L2 latches.

4- The circuit is switched back to test mode, and the contents of the L2 latches are shifted out via the scan path to the scan-out port by pulsing clocks A and B.

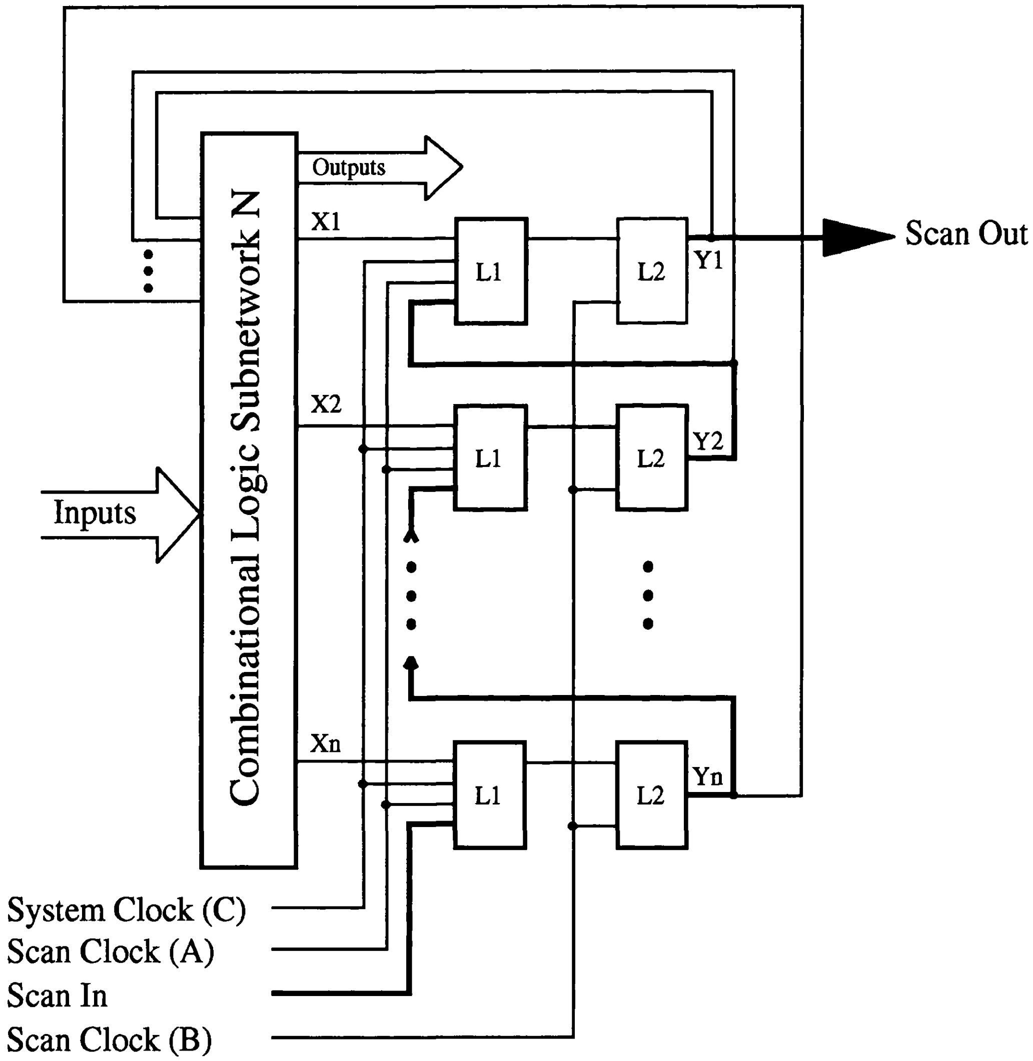


Figure 3.16: General Structure of an LSSD Subsystem

The procedure outlined above for testing the network in Figure 3.16 shows that, by implementing LSSD in scan path configuration the future states of the system can be set up independently of the present states, and internal states can be easily observed. Hence, the problem of testing a sequential network is reduced to that of testing a combinational network.

3.4.2.2 Boundary Scan

The task of testing the new generation of digital printed circuit boards by conventional methods, such as bed-of-nails fixtures, is extremely difficult, inefficient and, hence, costly. This is due to the high density of the boards, the use of surface-mounted device on both sides of the board, and the increase in the complexity of the ICs used.

The boundary-scan technique simplifies the problem of testing complex boards, by inserting in every IC small logic circuits, called boundary-scan cells [29,30]. The cells are inserted between each pin and the chip circuitry to which that pin is normally directly connected, as depicted in Figure 3.17. The architecture of a boundary-scan cell is shown in Figure 3.18. Within each chip, all the boundary-scan cells can be connected to each other to form a shift-register path around its periphery. An input pin (TDI) and output pin (TDO) provide a serial data connection to this register, while the clock (TCK) and control signals (TMS and TRST) control data movement and use according to a defined protocol that is interpreted by a small finite-state machine. On a loaded board, the ICs that conform to the boundary-scan architecture can be daisy chained, as shown in Figure 3.17, to form a single shift-register so that all of the circuitry and interconnections on the board is testable from the edge connector.

The boundary-scan structure has three modes of testing: External, Internal and Sample. These modes are selected by the instructions EXTEST, INTEST and SAMPLE respectively. During an external test mode (EXTEST) the boundary-scan cells are selected such that internal chip circuitry is isolated, and external circuitry and interconnections outside the ICs are fully tested for stuck-at, short-circuit, open-circuit and other fault types. In the internal test mode (INTEST) the boundary-scan cells can be used to perform a slow speed or static functional test of the internal circuitry of the ICs which have been designed with boundary-scan cells. Finally, it is possible to set the multiplexers such that the latches can sample (SAMPLE) the states appearing at the inputs and outputs of each chip during normal operation. The sampled data (snapshots) can be shifted out as a serial pattern under the control of the boundary-scan clock to build-up a picture of the operation of the circuit.

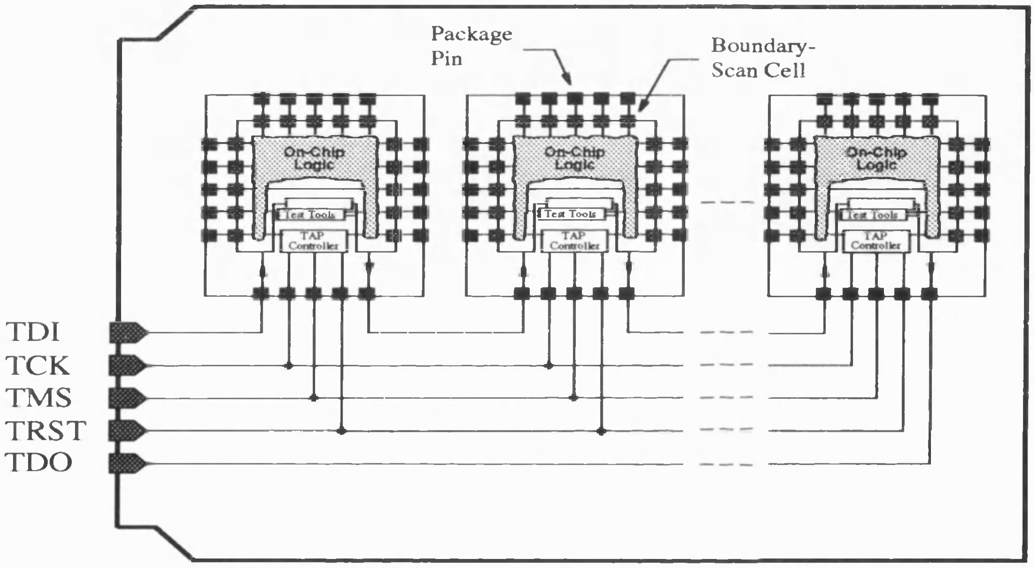


Figure 3.17: A Boundary-Scan Board with ICs Daisy Chained

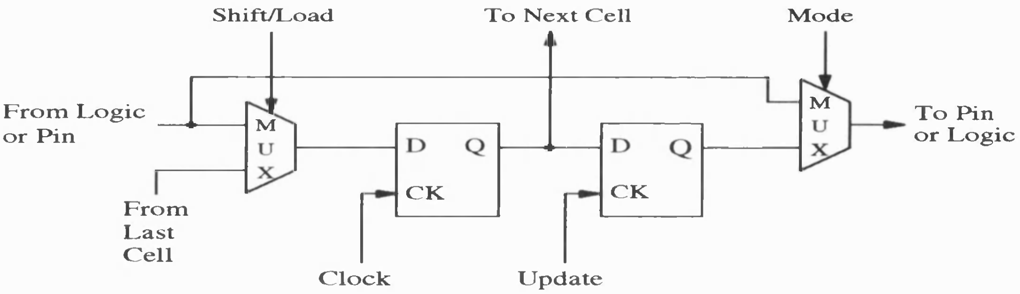


Figure 3.18: The Boundary-Scan Cell

The boundary-scan technique and architecture have evolved into an IEEE standard. The formal name of the standard is ANSI/IEEE std 1149.1, IEEE Standard Test Access Port and Boundary-Scan Architecture [31]. The standard has met wide acceptance by many IC manufacturers and is being implemented in the new generation of integrated circuits.

For an IC that complies with the 1149.1 standard the cost is an extra four or five pins, and the silicon area needed to implement the boundary-scan cells. The advantages of the standard include easing the testing problem of complex boards, containing components from different manufacturers, and reducing the tester requirement of expensive pin electronics for all input/output pins. Coupling the 1149.1 standard with built-in self-test techniques, which will be discussed in subsection 3.4.3, will allow powerful and expensive automatic test equipment (ATE) to be replaced by much simpler, low-cost test systems.

3.4.3 Built-In Self-Testing

Structured DFT techniques simplify the task of testing complex network to some extent. However, a great deal of data must still be processed: generate necessary test patterns, compute and store true value output responses, store and analyze output responses of the circuit-under-test, and determine the fault-coverage. All these processing activities are time-consuming and costly. To alleviate some of the data processing difficulties and enhance circuit testability, built-in self-testing (BIST) techniques have been developed [1,22,28]. The goal of these techniques is to incorporate circuitry to an integrated circuit to enable it to carry out some form of self-testing. Two BIST techniques will be discussed, signature analysis and built-in logic observation.

3.4.3.1 Signature Analysis

The task of checking the results of applying a set of test vectors is a non-trivial one. However, by using a data compression technique, the task can be eased by producing a series of so-called signatures for the circuit-under-test. The key component in the signature analysis technique is the pseudo-random binary sequence (PRBS) generator, also known as a linear feedback shift-register (LFSR). An example is shown in Figure 3.19. It consists of a shift-register with its output exclusive-ORed with a tap point and fed back into the input of its first stage. An appropriate choice of tap point gives, for N bits, a sequence of 2^N-1 bits. Different sequences are produced by different tap points.

Complementary to the LFSR is the signature analysis register, shown in Figure 3.20. In this case, the output of the exclusive-OR in Figure 3.19 is not returned directly to the input of the first stage, instead it is exclusive-ORed with a signal from a circuit-under-test node that needs to be monitored. Therefore, the PRBS in the LFSR will be modified by the signal of the node. The modified bit sequence in the latches of the LFSR is referred to as the "signature" of that particular node.

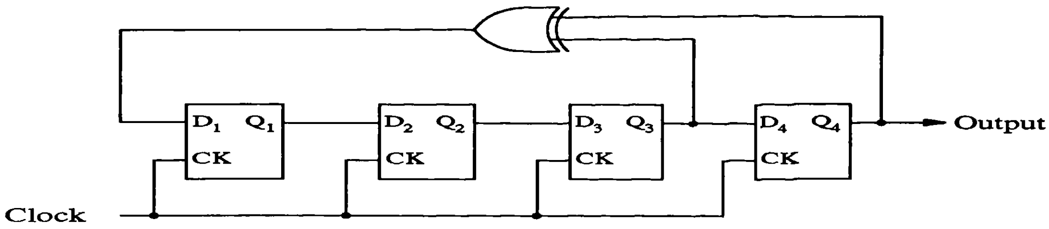


Figure 3.19: A Four-Stage Pseudo-Random Binary Sequence Generator

To demonstrate fault detection by the signature analysis technique, assume that the LFSR is initialized to a particular bit pattern and then exclusive-ORed with a signal of a node in a fault-free circuit. After a prescribed number of K clock pulses, a signature characteristic of the fault-free circuit will be stored in the LFSR. If the same procedure is repeated with a signal from a faulty circuit, then after the same number of K clock pulses a signature which is different from that in the fault-free case will accumulate in the LFSR, enabling the identification of the faulty circuit. Provided that the PRBS is long enough the likelihood of the LFSR ending with a signature of a fault-free circuit when a faulty sequence appears at the monitored node, is statistically remote [22,28]. It can be seen that the effect of the signature analysis technique is the compression of the output data to be analyzed into a single N-bit word, thus eliminating the need to compare long sets of output bit patterns.

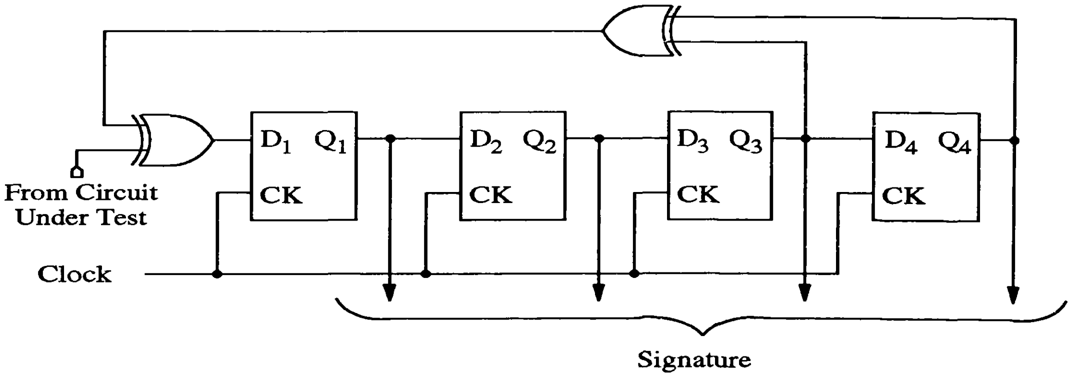


Figure 3.20: A Four-Stage Signature Analysis Register

The signature analysis principle, discussed above, can be extended to allow more than one circuit node to be monitored simultaneously as shown in Figure 3.21. In the circuit, the Z-inputs are connected to the nodes being monitored and, by superposition, the final signature will be determined by the bit pattern appearing at the monitored nodes.

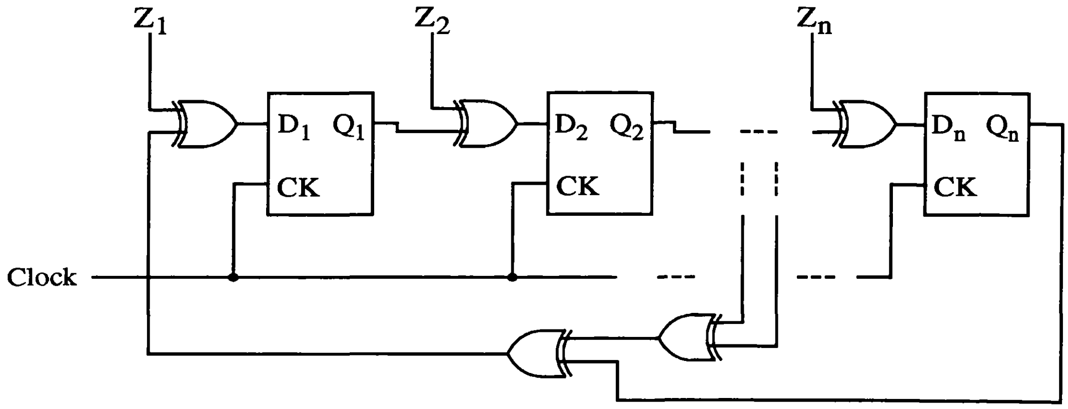


Figure 3.21: Multiple Input Signature Analysis Register (MISAR)

3.4.3.2 Built-In Logic Block Observation

The built-in logic block observation (BILBO) technique [1,32] for built-in self test, combines the scan path and signature analysis concepts discussed earlier. The basic element of this technique is the BILBO register depicted in Figure 3.22. The register is a multi-purpose test module that can be configured to function as an input test pattern generator or an output signature analyzer. The latches L_i (i=1,2..n) are the system latches. Z_i (i=1,2,..n) and Q_i (i=1,2,..n) are the inputs from the combinational logic and latches outputs, respectively. S_{in} and S_{out} are the scan-in input and scan-out output of the BILBO register. The two control inputs B_1 and B_2 are used to select one of the BILBO four function modes:

1- B_1=1, B_2=1: This is the normal system function, when the BILBO register behaves as a latch. The Z_i input data is loaded into the L_i latches simultaneously and the outputs are available on Q_i for system operation.

2- B_1=0, B_2=0: The BILBO acts as an LFSR forming a scan path. Data are serially clocked into the register through S_{in}, while the register contents can be simultaneously read at the Q_i outputs, or can be clocked out serially through S_{out}.

3- B_1=1, B_2=0: The BILBO functions as a multiple-input signature analysis register. In this mode BILBO may be used as a parallel signature analyzer or a PRBS generator. For the later case the logical values of the inputs Z_i must not change.

4- B_1=0, B_2=1: All the latches in the BILBO register are reset.

The application of the BILBO technique for testing is illustrated in Figure 3.23. To test combinational network-1, BILBO-1 is configured to generate pseudo-random sequences, which are applied to the input of combinational network-1, and BILBO-2 is configured as a signature analysis register, which is after K clock cycles will contain the signature of network-1. If the roles of BILBO-1 and BILBO-2 are then reversed combinational network-2 can be tested.

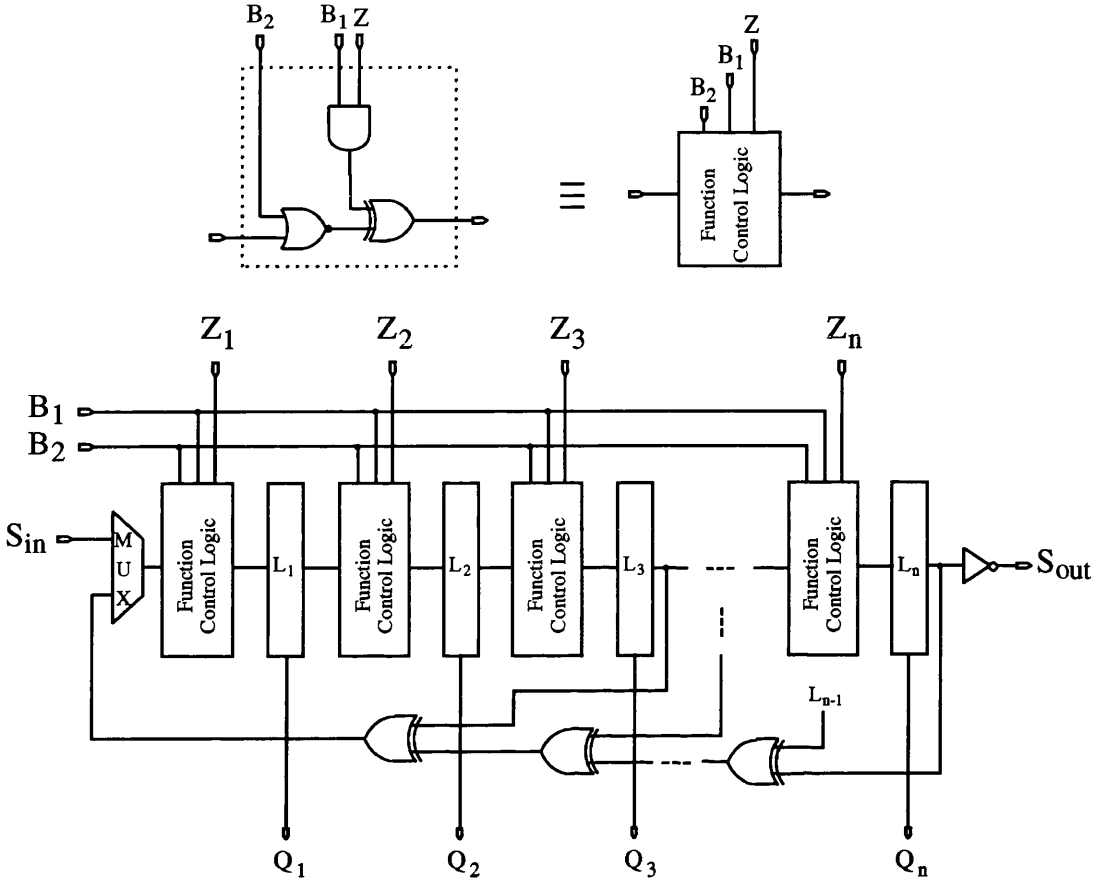


Figure 3.22: The General Structure of BILBO Register

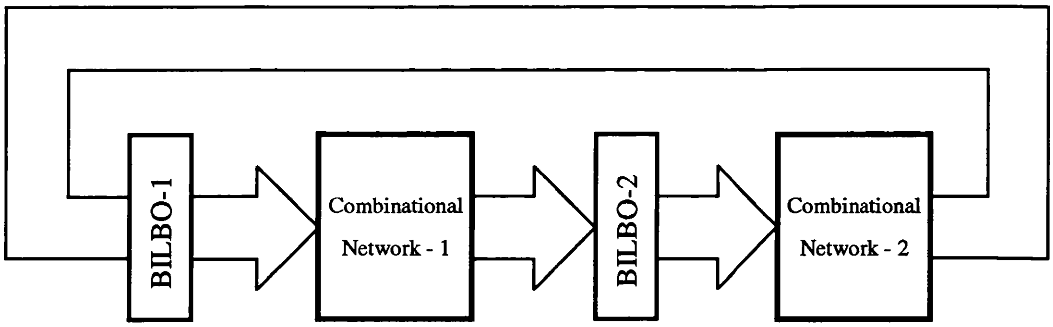


Figure 3.23: The Use of BILBO Registers for BIST

3.5 Testing of Analogue Integrated Circuits

The state-of-the-art computer tools, such as simulators, that are used in the design of digital circuits are more advanced than those for analogue circuits. The testing algorithms and design for testability techniques, described in the previous sections, for digital ICs virtually have no equivalent for analogue ICs. In fact the field of testing analogue ICs is still in its infancy.

The primary reasons why analogue circuits design and test tools are less advanced than their digital counterparts can be attributed to the following:

1- Digital circuits are more structured than analogue circuits.

2- The tolerance problem in analogue circuits, which has no equivalent in digital circuits, imposes limitations and difficulties on designing and testing these circuits.

3- Analogue simulation is usually performed at the device level (e.g. resistor, transistor ... etc), and involves the solution of many linear and nonlinear equations. Digital simulation, on the other hand, is performed at the block level (e.g. AND, OR..etc). Hence, digital circuits simulators are much faster than analogue circuits ones.

4- Analogue circuits testing is usually specification-driven (e.g. gain, bandwidth, total harmonic distortion ... etc), due to the lack of an adequate fault-model. This leads to long testing time and requires expensive test equipment.

5- The modes of failure in analogue circuits are much more than those in digital circuits. For example, a faulty resistor may have an infinite number of possible resistances (outside of the tolerance region). This makes the task of deriving a fault-model a very difficult one. In [33] an attempt is made to model faults in analogue circuits based on experimental statistics of manufacturing defects. The work is, however, still in very early stages and needs a lot of effort before proceeding to derive an abstract fault model.

Despite the above mentioned difficulties, a concerted effort was put, especially during the past two decades, to devise testing algorithms for discrete analogue circuits and systems [34,35]. These testing algorithms can broadly be classified into three categories:

1- Simulation-before-test
2- Simulation-after-test with a single test vector
3- Simulation-after-test with multiple test vectors.

The simulations-before-test approach is also referred to as the dictionary approach for fault location. In this approach the circuit under test (CUT) is simulated, before actual testing (i.e. off-line), using certain input signals for a number of hypothesized faults. The responses are then stored as a dictionary. During actual testing the CUT is excited by the specified input signals and the responses obtained are compared with the simulated results in an attempt to determine the cause of the malfunction of the CUT. The fault candidate that produces the closest simulated response with respect to a certain measure to that produced during actual testing is declared the likely fault.

The implementation of the fault dictionary approach, therefore, consists of two stages. Before conducting the test, the dictionary is constructed. At the time of actual testing a search process is conducted using the stored data and the measurements to locate the fault or fault set that contains the possible fault.

The reported techniques of implementing the dictionary approach [34-38] differ mainly in the type of input/output measurements, fault signature, fault location technique, and degree of diagnosability and fault isolation. Of the various implementations, the dc dictionary approach described in [36] will be discussed in detail in Chapter 4.

In the simulation-after-test using a single test vector category, the responses of the analogue network to a single input stimulus are analyzed to determine the faulty elements of the network. Consequently, rather than simulating the network before the test as in the dictionary approach, most of the network analyses and simulations are performed after conducting the actual testing. The techniques to be followed to locate a fault or faults depend upon the number of available measurements [34,35]. If the number of independent measurements is less than the number of the network elements, which is usually the case, two main approaches are followed:

1- Estimation methods [39-41], where an estimation criterion is used to identify the most possible fault by applying deterministic and probabilistic methods.

2- Fault verification methods [42-48], where an upper bound is assumed on the number of simultaneous faults, usually less than the number of performed measurements. The techniques that fall in this category, basically, address two main issues. The first is the uniqueness of the diagnosable elements in both linear and nonlinear networks [42-45]. The second is the development of techniques to speed up the search for the faulty set [45-47].

If the number of independent measured quantities using a single test vector is equal to the number of network parameters, full identification of the network parameters can be carried out and the faulty elements are thereby isolated [49].

The main limitation of parameter identification is that the number of required measurements, and hence equations to be solved, grows linearly with the complexity of the network.

In Chapter 4, the logical decomposition method [47], which falls in the simulation-after-test with a single test vector category, will be discussed in detail.

The objective of the simulation-after-test with multiple test vectors techniques is to reduce the number of test points required. This is achieved, as the name implies, by exciting a network with multiple test vectors in order to increase the number of equations derivable from a given set of test points [50-52]. The major problem with the techniques that fall in this category, is that the equations to be solved are often nonlinear and the computational effort is quite excessive for the on-line implementation.

Of the three categories discussed above, the simulation-after-test with a single test vector techniques are generally preferred for testing discrete analogue circuits and systems [35]. This is due to the acceptable on-line and off-line computation time, and moderate number of test points required. Factors affecting the suitability of algorithms in each category are studied in detail in [35].

The majority of the techniques outlined above for testing discrete analogue circuits and systems, unfortunately, cannot be readily extended to testing analogue integrated circuits. This is due to the following:

1- Failure modes in integrated circuits are different from those in their discrete counterpart.

2- Most of the algorithms require access to a high number of test points or nodes. In ICs extra test points occupy valuable silicon area, increase the cost of the chip especially if they need to be brought out of the chip as pins, and may complicate the original circuit design.

3- Many of the algorithms are designed to deal with a small number of particular type of components (e.g. only passive), or one kind of faults (e.g. catastrophic).

4- Testing priorities in ICs are different from those in discrete circuits. In the discrete circuits case a lot of effort was put into devising algorithms that locate a fault, and diagnose it. However, in production testing of ICs from a mature manufacturing process, the focus of the diagnosis effort is on the process line and not the circuit. This is due to two reasons. First, it is generally either impossible or very expensive to repair individual ICs that fail testing. Second, for well-designed circuits, when the yield of a manufacturing line drops, it is typically due to a process fault that has escaped the process monitoring system. If this happens, detailed analysis of measurements made on the ICs and the test structures on the wafer are used to diagnose the drop in the process yield. Therefore, in IC volume production testing the objective is to distinguish the good circuit from the faulty ones (i.e. Go/No-Go).

In the next chapter some of the algorithms, that were originally designed for testing discrete analogue circuits, will be evaluated in depth. The objective is to develop the algorithms, if possible, to make them applicable for testing integrated analogue and mixed-signal circuits.

3.6 Testing of Mixed-Signal Circuits

Traditionally, analogue and digital circuits are integrated on separate substrates using different technologies; bipolar for analogue ICs and MOS for digital ICs. However, recent improvements in CMOS and BiCMOS IC fabrication processes enabled the realization of analogue and digital functions on a common substrate. Such mixed-signal devices, having both analogue and digital circuits on the same chip as shown in Figure 3.24, improve performance and reliability, and reduce cost.

Most of the mixed-signal devices are in the application specific integrated circuits (ASICs) market, because they are usually designed for specific functions and produced in relatively small quantities. Major applications of mixed-signal ASICs are in the telecommunications, consumer and automotive markets. The availability of analogue and digital functions in the form of library cells which the designer can choose from to construct the required ASIC, and the continuous advances in processing technology are resulting in an increase in the demand for mixed-signal ASICs. In [53] it is indicated that the current annual growth rate in the mixed-signal ASICs market is around 35-40%, compared with 18% for all ASICs.

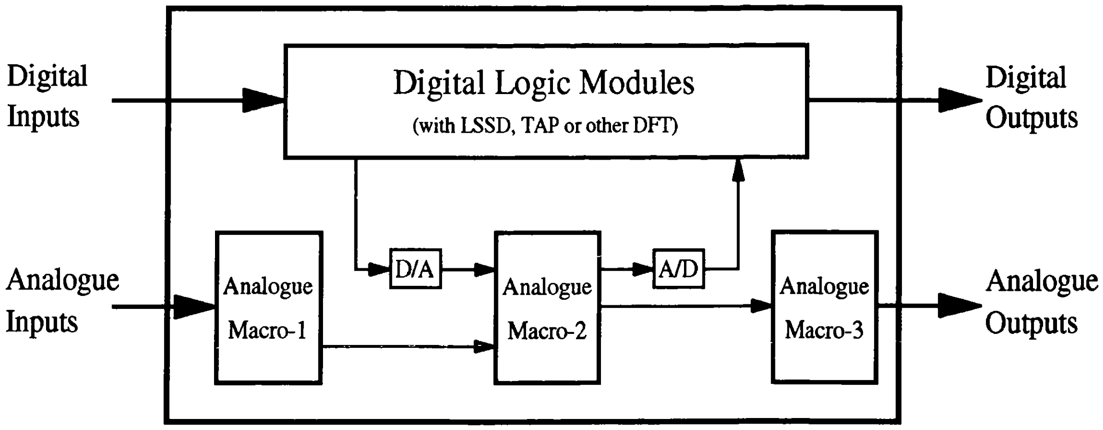


Figure 3.24: Architecture of a General Mixed-Signal IC

The incorporation of analogue and digital circuits in mixed-signal ASICs makes the task of testing such ICs a very difficult one. This is due to the difficulties associated with testing analogue circuits, which were outlined in the previous section, and the lack of controllability and observability of embedded circuit modules. The testing task is exacerbated further by the presence of interface circuit blocks, such as analogue-digital (A/D) and digital-analogue (D/A) converters, and other circuit modules (e.g. switched capacitor circuits) that exhibit both analogue and digital characteristics.

At present the testing strategies that are applied in practice to mixed-signal devices, rely on partitioning the device-under-test into separate analogue and digital blocks, and the application of mode specific tests to each block. To perform mode specific testing two options are possible. The first is using separate testers; one for analogue and another for digital. The second is using one of the commercial mixed-signal testers, which is basically a digital tester with powerful DSP (digital signal processing) capabilities to emulate analogue instruments and generate the required test routines. The first option leads to loss of synchronization between blocks due to the lack of centralized control. The second option provides centralized control to co-ordinate inter-block activities, emulated analogue instruments do not suffer from the usual anomalies associated with actual ones (e.g. noise, drift, improper calibration, non-linearity ..etc) leading to more accurate measurements, and tests can be applied faster than in the previous option. However, testers with DSP capabilities tend to be very expensive and can only be afforded by large companies. An in depth study of DSP-based testing is presented in [54].

To isolate the analogue modules on a mixed-signal IC, and provide access to some of the internal nodes, (i.e. control and observe) of these modules to enable the applications of mode specific test, a number of DFT techniques for the analogue circuits have been proposed [55-59]. Some of these techniques are described below.

In [55,56] an analogue multiplexer (MUX) is placed at the input of each analogue macro to give controllability as depicted in Figure 3.25. The output of the macro is observed by using another MUX which is common to the output of all analogue macros. All the analogue test inputs (T) are routed through a demultiplexer which is not shown in the diagram in the interest of clarity. The MUXs in the diagram are controlled by signal (C).

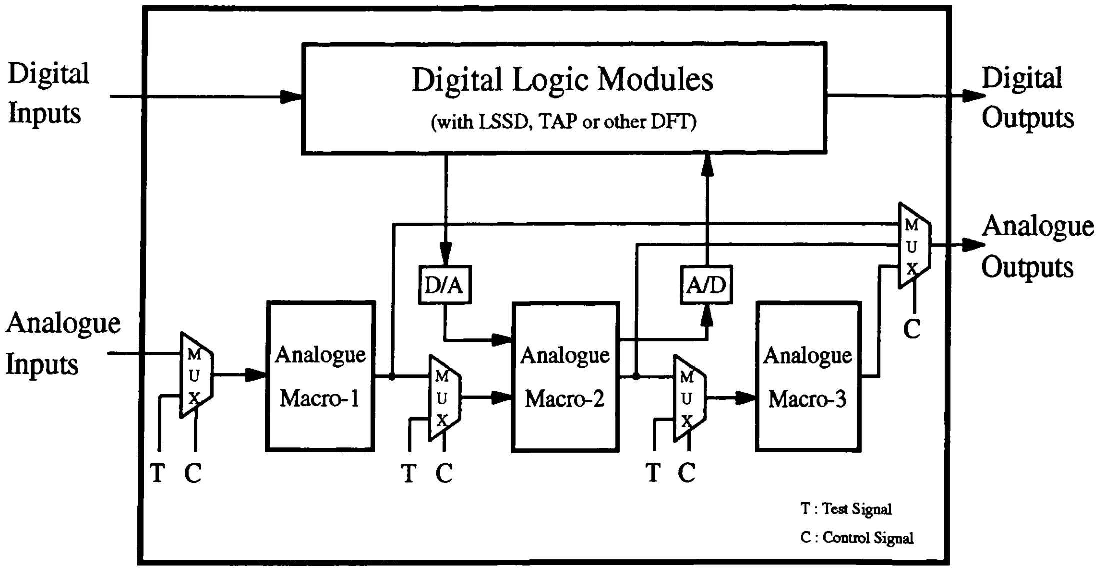


Figure 3.25: Multiplexer-Based Analogue DFT Technique

Another analogue DFT technique, that was presented in [57], uses an analogue shift register (ASR) block to form an analogue built-in-self-test (ABIST) structure (see Figure 3.26) similar to scan path in digital circuits. The ASR, which is the block in a dashed box in Figure 3.26, is a sample-and-hold circuit consisting of a switch (SW), a capacitor (C) and an analogue buffer (VF). For each test point (m_i) a buffer (B) and a switch (SP) are required. The ABIST in Figure 3.26 has two basic modes of operation: normal mode and test mode. In the normal mode, all switches SW_i's are turned off and no test data are sampled. On the other hand, during the test mode sampled test data are loaded in parallel by turning switches SW_i's on and SP_i's off. The sampled data are then passed serially to S_{out} by turning SW_i's off and then SP_i's on in a sequence so that the charge on C_i is passed before the charge on C_{i-1}. A secondary mode of operation enables the ABIST to test itself by turning SW_{in} on and applying a test signal to input S_{in} and propagating it to output S_{out}. The pin over head of the ABIST is two pins: S_{in} an S_{out}.

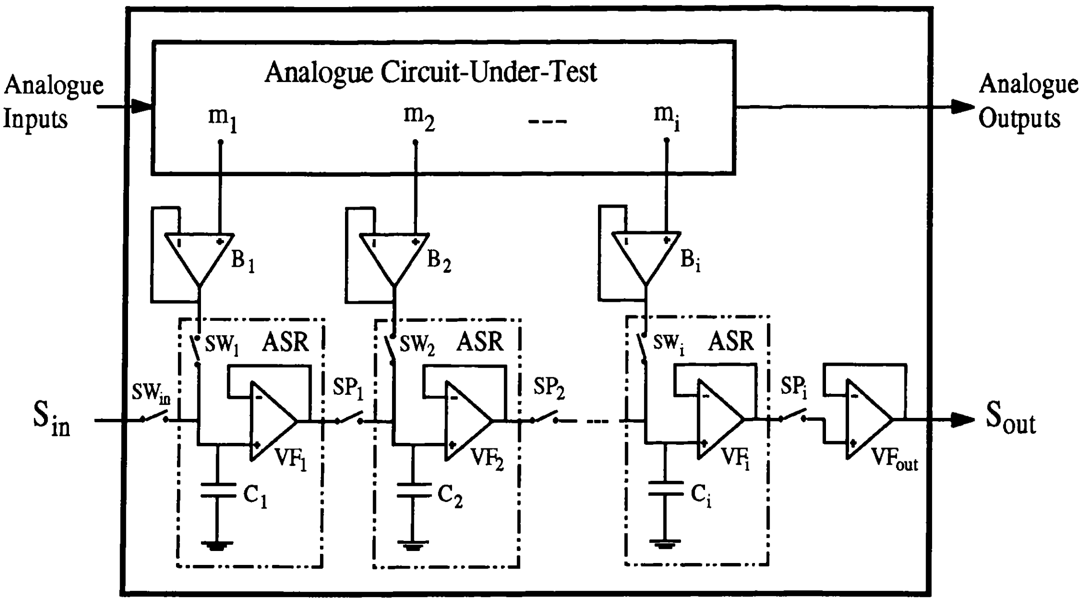


Figure 3.26: The ABIST Structure

In [58] a hybrid built-in-self-test (HBIST) strategy is described. The strategy is applicable to ICs which combine large digital kernel systems with peripheral analogue sub-circuits either sited on the input side only or on both the input and output of the chip. It uses the BIST of the digital section to scan in the test data for the analogue section, and the D/A converter to generate a multi-level piece-wise-constant signal, from the scanned in data, to be applied to the analogue section. The response of the analogue section is then converted to digital formats by the A/D converter and scanned out by the digital BIST.

The use of MOS switches to isolate various active filter stages, hence improve their testability was also suggested [59].

Partitioning a mixed-signal IC, by utilizing one of the above DFT techniques or other strategies, to apply mode specific test has many disadvantages. It constrains the designer to consider how a chip should ideally be partitioned during the initial design phase. The majority of the partitioning approaches involve additional circuit blocks (e.g. MUXs and buffers) and probe pads, resulting in a substantial increase in silicon area overhead and complex designs, leading to a reduction in the overall yield. For example, the ABIST approach requires two buffers, two switches, and a capacitor for each test point. Therefore, the size of the analogue DFT section may be close to or more than the size of the analogue macro. This increase in area is not acceptable since, in general, the analogue sections tend to occupy no more than 30% of the total chip area. The two stage testing would be time consuming, costly and would not allow the testing of the analogue/digital interface and vice versa accurately.

Given the disadvantages of the partitioning strategies, there is a great need for, ideally, a unified testing technique for mixed-signal ICs. Such a technique would apply to both the analogue and digital modules simultaneously, can be performed on one test setup, and incurs no or minimum area overhead due to the elimination of the need for partitioning. The remaining chapters of this thesis will be concentrating on devising unified testing techniques that are close to the objectives of the ideal one.

An important final problem that greatly contributes to the difficulty in designing and testing mixed-signal ASICs is the lack of a true mixed-signal simulator. This is due to the inherent problems associated with simulating analogue circuits, which were outlined in the previous section. The commercial mixed-signal simulators, that are currently available, basically have two simulation engines: one for analogue and another for digital. Such simulators are expensive, slow and lack flexibility because the analogue and digital simulators have to be tightly coupled. The subject of mixed-signal simulation, like the mixed-signal testing, is the focus of on going research [60-62].

3.7 References

[1] T.W. Williams and K.P. Parker, "Design for Testability - A Survey", Proceedings of the IEEE, Vol. 71, No. 1, pp. 98-112, January 1983.

[2] J.P. Roth, W.G. Bouricius and P.R. Schneider, "Programmed Algorithms to Compute Tests to Detect and Distinguish Between Failures in Logic Circuits", IEEE Trans. Electronic Computers, Vol. EC-16, No. 5, pp. 567-580, October 1967.

[3] A.K. Susskind, "Diagnostics for Logic Networks", IEEE Spectrum, pp. 25-34, October 1973.

[4] R.L. Wadsack, "Fault Modeling and Logic Simulation of CMOS and MOS Integrated Circuits", The Bell System Technical Journal, Vol. 57, No. 5, pp. 1449-1474, May-June 1978.

[5] A.A. Ismaeel, "Testing for Stuck Faults in CMOS Combinational Circuits", ICE Proceedings on Circuits, Systems and Devices - Part G, Vol. 138, No. 2, pp. 191-197, April 1991.

[6] Y. Malaiaya and S.Y. Su, "A New Fault Model and Testing Technique for

CMOS Devices", Proc. Int. Test Conference, pp. 25-34, November 1982.

[7] K.W. Chiang and Z.G. Vranesic, "On Fault Detection in CMOS Logic Networks", Proc. 20th Design Automation Conference, pp. 50-56, June 1983.

[8] K.C. Mei, "Bridging and Stuck-at Faults", IEEE Trans. Computers, Vol. C-23, No. 7, pp. 720-727, July 1974.

[9] H.C. Shih and J.A. Abraham, "Transistor-Level Test Generation for Physical Failures in CMOS Circuits", Proc. 23rd Design Automation Conference, pp. 243-249, June 1986.

[10] J.P. Shen, W. Maly and F.J. Ferguson, "Inductive Fault Analysis of MOS Integrated Circuits", IEEE Design and Test of Computers, December 1985.

[11] J. Gracio, P. Bicudo, N. Rua, A. Oliveira, C. Almeida and J. Teixeira, "Test Preparation and Fault Analysis Using a Bottom-Up Methodology", Proc. European Test Conference, pp. 168-174, 1989.

[12] M. Jacomet, "FANTESTIC: Towards a Powerful Fault Analysis and Test Pattern Generator for Integrated Circuits", Proc. Int. Test Conference, pp. 633-642, 1989.

[13] C. Stapper, F. Armstrong and K. Saji, "Integrated Circuit Yield Statistics", Proceedings of the IEEE, Vol. 71, No. 4, pp. 453-470, April 1983.

[14] W. Maly, A. Strojwas and S. Director, "VLSI Yield Prediction and Estimation: A Unified Framework", IEEE Trans. Computer-Aided Design, Vol. CAD-5, No. 1, pp. 114-130, January 1986.

[15] J. Galiay, Y. Crouzet and M. Vergniault, "Physical Versus Logical Fault Model MOS LSI Circuits: Impact on Their Testability", IEEE Trans. Computers, Vol. C-29, No. 6, pp. 527-531, June 1980.

[16] L. Milor and V. Visvanathan, "Detection of Catastrophic Faults in Analog Integrated Circuits", IEEE Trans. Computer-Aided Design, Vol. 8, No. 2, pp. 114-130, February 1989.

[17] J. Soden and C. Hawkins, "Electrical Properties and Detection Methods for CMOS IC Defects", Proc. European Test Conference, pp. 159-167, 1989.

[18] HSPICE User's Manual, Meta-Software Inc., Campell, California 1988.

[19] H.C. Shih and J.A. Abraham, "Fault Collapsing Techniques for MOS VLSI Circuits", Proc. 16th Fault Tolerant Computing Symposum, pp. 370-375, 1986.

[20] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits", IEEE Trans. Computers, Vol. C-30, No. 3, pp. 215-222, March 1981.

[21] H. Fujiwara and T. Shimono, "On the Acceleration of Test Generation Algorithms", IEEE Trans. Computers, Vol. C-32, No. 12, pp. 1137-1144, December 1983.

[22] V.N. Yarmolik, "Fault Diagnosis of Digital Circuits", John Wiley and Sons, Chichester, England 1990.

[23] S.K. Jain and V.D. Agrawal, "Test Generation for MOS Circuits Using D-Algorithm", Proc. 20th Design Automation Conference, pp. 64-70, June 1983.

[24] V.D. Agrawal and S.C. Seth, Test Generation for VLSI Chips, IEEE Computer Society Press, New York, 1988.

[25] S.M. Reddy, M.K. Reddy and P. Agrawal, "Transistor Level Test Generation for MOS Circuits", Proc. 22nd Design Automation Conference, pp. 825-828, June 1985.

[26] L.H. Goldstein and E.L. Thigpen, "SCOAP: Sandia Controllability and Observability Analysis Program", Proc. 17th Design Automation Conference, pp. 190-196, 1980.

[27] D.K. Goel and R.M. McDermott, "An Interactive Testability Analysis Program - ITTTAP", Proc. 19th Design Automation Conference, pp. 581-586, 1982.

[28] G. Russell, D.J. Kinniment, E.G. Chester and M.R. McLauchlan, CAD for VLSI, Van Nostrand Reinhold (UK) Co. Ltd, England 1985.

[29] C.M. Maunder and R.E. Tulloss, The Test Access Port and Boundary Scan Architecture, IEEE Computer Society Press, California 1990.

[30] C.M. Maunder and R.E. Tulloss, "Testability on TAP", IEEE Spectrum, Vol. 29, No. 2, pp. 34-37, February 1992.

[31] IEEE Standard 1149.1, Standard Test Access Port and Boundary-Scan Architecture", IEEE Computer Society, 1990.

[32] B. Koenamann, J. Mucha and G. Zwiehoff, "Built-in Logic Block Observation Technique", Proc. IEEE Test Conference, pp. 37-41, 1979.

[33] M. Soma, "An Experimental Approach to Analogue Fault Model", Proc. IEEE Custom Integrated Circuits Conference, pp. 13.6.1-4, May 1991.

[34] P. Duhamel and J.C. Rault, "Automatic Test Generation Techniques for Analog Circuits and Systems: A Review", IEEE Trans. Circuits and Systems, Vol. CAS-26, No. 7, pp. 411-440, July 1979.

[35] J.W. Bandler and A.E. Salama, "Fault Diagnosis of Analog Circuits", Proceedings of the IEEE, Vol. 73, No. 8, pp. 1279-1325, August 1985.

[36] W. Hochwald and J.D. Bastian, "A DC Approach for Analog Fault Dictionary Determination", IEEE Trans. Circuits and Systems, Vol. CAS-26, No. 7, pp. 523-529, July 1979.

[37] H.H. Schreiber, "Fault Dictionary Based Upon Stimulus Design", IEEE Trans. Circuits and Systems, Vol. CAS-26, No. 7, pp. 529-537, July 1979.

[38] K.C. Varghese, J.H. Williams and D.R. Towill, "Simplified ATPG and Analog Fault Location Via a Clustering and Separability Technique", IEEE Trans. Circuits and Systems, Vol. CAS-26, No. 7, pp. 496-504, July 1979.

[39] M.N. Ransom and R. Saeks, "Fault Isolation with Insufficient Measurements", IEEE Trans. Circuit Theory, Vol. CT-20, pp. 416-417, 1973.

[40] J.W. Bandler, R.M. Biernacki and A.E. Salama, "A Linear Programming Approach to Fault Location in Analog Circuits", Proc. IEEE Int. Symp. Circuits and Systems, pp. 256-260, 1981.

[41] L. Hu, Z.F. Huang, Y.F. Huang and R.W. Liu, "A Stochastic Model for Analog Fault Diagnosis with Tolerance", Proc. IEEE Int. Symp. Circuits and Systems, pp. 680-683, 1984.

[42] J.A. Strazyk and J.W. Bandler, "Nodal Approach to Multiple Fault Location in Analog Circuits", Proc. IEEE Int. Symp. Circuits and Systems, pp. 1136-1139, 1982.

[43] T. Ozawa, "Topological Considerations on the Diagnosability of Analog Circuits", Proc. IEEE Int. Symp. Circuits and Systems, pp. 664-667, 1984.

[44] V. Visvanathan and A. Sangiovanni-Vincentelli, "Diagnosability of Nonlinear Circuits and Systems - Part I: The dc Case", IEEE Trans. Circuits and Systems, Vol. CAS-28, No. 11, pp. 1093-1102, November 1981.

[45] R. Saeks, A. Sangiovanni-Vincentelli and V. Visvanathan, "Diagnosability of Nonlinear Circuits and Systems - Part II: Dynamical Systems", IEEE Trans. Circuits and Systems, Vol. CAS-28, No. 11, pp. 1103-1108, November 1981.

[46] C.C. Wu, K. Nakajima, C.L. Wey and R. Saeks, "Analog Fault Diagnosis with Failure Bounds", IEEE Trans. Circuits and Systems, Vol. CAS-29, No. 5, pp. 277-284, May 1982.

[47] A.E. Salama, J.A. Strazyk and J.W. Bandler, "A Unified Decomposition Approach for Fault Location in Large Analog Circuits", Vol. CAS-31, No. 7, pp. 609-622, July 1984.

[48] T. Ozawa, J.W. Bandler and A.E. Salama, "Diagnosability in the Decomposition Approach for Fault Location in Large Analog Networks", IEEE Trans. Circuits and Systems, Vol. 32, No. 4, pp. 415-416, April 1985.

[49] R. Saeks, S.P. Singh and R.W. Liu, "Fault Isolation Via Component Simulation", IEEE Trans. Circuits Theory, Vol. CT-19, pp. 634-640, 1972.

[50] N. Sen and R. Saeks, "Fault Diagnosis for Linear Systems Via Multifrequency Measurements", IEEE Trans. Circuits and Systems, Vol. CAS-26, No. 7, pp. 457-465, July 1979.

[51] S. Shinoda, I. Yamaguchi, K. Omura and I. Shirakawa, "Parameter - Value Determination Using Several Test Frequencies", Proc. IEEE Int. Symp. Circuits and Systems, pp. 675-679, 1984.

[52] J. Rapisarda and R.A. DeCarlo, "Analog Multifrequency Fault Diagnosis", IEEE Trans. Circuits and Systems, Vol. CAS-30, No. 4, pp. 223-234, April 1983.

[53] R. Massara and K. Steptoe, "Analogue and Mixed-Signal IC Design", IEE Review, Vol. 38, No. 2, pp. 75-79, February 1992.

[54] M. Mahony, DSP-Based Testing of Analog and Mixed-Signal Circuits, IEEE Computer Society Press, 1987.

[55] P.P. Fasang, D. Mullins and T. Wong, "Design for Testability for Mixed Analog/Digital ASICs", Proc. IEEE Custom Integrated Circuits Conference, pp. 16.5.1-4, 1988.

[56] K.D. Wagner and T.W. William, "Design for Testability of Mixed Signal Integrated Circuits", Proc. International Test Conference, pp. 823-828, 1988.

[57] C.L. Wey, "Built-In-Self-Test (BIST) Structure for Analog Circuit Fault Diagnosis", IEEE Trans. Instrumentation and Measurement, Vol. 39, No. 3, pp. 517-521, June 1990.

[58] M.J. Ohletz, "Hybrid Built-In-Self-Test (HBIST) for Mixed Analogue/Digital Integrated Circuits", Proc. European Test Conference, pp. 307-316, 1991.

[59] M. Soma, "A Design-for-Test Methodology for Active Analog Filters", Proc. International Test Conference, pp. 183-192, 1990.

[60] T.H. Morrin, "Mixed-Mode Simulation for Time-Domain Fault Analysis", Proc. International Test Conference, pp. 231-241, 1989.

[61] P.E. Allen, B.P. Chan and W.M. Zuberek, "Comparison of Mixed Analog-Digital Simulators", Proc. Int. Symp. Circuits and Systems, Vol. 1, pp. 101-104, 1990.

[62] E. Berkan and F. Yassa, "Towards Mixed Analog/Digital Design Automation: A Review", Proc. Int. Symp. Circuits and Systems, Vol. 2, pp. 809-815, 1990.

CHAPTER FOUR

DEVELOPMENT OF EXISTING TESTING TECHNIQUES

In this chapter, three techniques that were originally devised for testing discrete analogue circuits will be investigated in depth. The objectives are to develop and assess the suitability of these techniques for testing analogue and mixed-signal ICs.

The three techniques that will be investigated are :

1- DC fault dictionary
2- Digital modelling
3- Logical decomposition

4.1 DC Fault Dictionary

The dc fault dictionary testing strategy presented in this section is an improvement on that proposed in [1]. This strategy consists basically of two stages; a pre-test stage to compile the fault dictionary and a post-test stage to identify (i.e. isolate) the fault.

In the pre-test stage the circuit-under-test (CUT) is simulated under nominal (i.e. fault-free), as well as all preselected catastrophic fault conditions. The induced voltages at all or a selected set of test nodes are calculated. These voltages are then stored and constitute the fault dictionary. In the post-test stage measurements of test node voltages which have been made on the circuit are compared with those stored in the fault dictionary to detect and identify a fault if one is present. The dc input stimuli should be selected to exercise the semiconductor devices in various regions of operation.

The detection process consists of dividing the voltage space of each test node into ambiguity sets. An ambiguity set contains a list of faults which, when they occur, will cause the particular test node to have a voltage value that falls within the range of the set. The range of an ambiguity set for each node is established as follows:

1- In the case of a fault-free circuit each node will have a nominal voltage V_n. But due to the fluctuation of the process parameters caused by changes in the manufacturing environment, the voltages of these fault-free circuits do not lie at exactly their nominal values. Therefore, the voltage at a node would be acceptable if it falls within what is called a nominal ambiguity set. The bounds of this nominal set are determined by V_n and a tolerance voltage ΔV. Hence, the nominal ambiguity set equals $V_n \pm \Delta V$. A fault that produces a voltage that falls within the range of the nominal ambiguity set of a certain node is considered undetectable at that test node.

2- The first fault that produces a nodal voltage value outside $V_n \pm \Delta V$ forms a new ambiguity set with a range of $V_f \pm \Delta V$, where V_f is the calculated faulty voltage of that node. Other faults falling outside the nominal set are checked to establish if they fall within an existing ambiguity set range. If not a new ambiguity set is created. If an overlap between two successive sets occurs, then to eliminate it, the overlap is divided equally between the two sets and an arbitrarily small guarding voltage is used to separate the sets.

To determine which fault is present in the CUT, the ambiguity sets are manipulated by applying two rules:

1- Any ambiguity set that has a single fault within it uniquely defines that fault at that node.

2- Ambiguity sets whose intersection or symmetric difference results in a single fault, also uniquely define that fault. In this case all the nodes for all ambiguity sets involved are required.

The fault dictionary algorithm described can also be based on the measurements of currents in the circuit branches instead of node voltages.

4.1.1 Examples

The algorithm described above was applied to a CMOS comparator and a CMOS operational amplifier circuit. The catastrophic fault conditions introduced to both circuits are based on the MOS fault model in Figure 3.6.

4.1.1.1 Comparator Circuit

The CMOS comparator circuit [2] depicted in Figure 4.1 was simulated under fault-free and faulty conditions. A total of 53 single catastrophic faults were introduced to the circuit.

Simulation conditions: V_{DD} was set at +5.0 volts and V_{SS} at zero. Two tests were conducted. In the first one the inverting terminal (node 7) was set at -5.0 volts and the non-inverting terminal (node 8) was set at +5.0 volts, the calculated output voltage (node 11) under fault-free condition was +4.97 volts. In the second test the input stimuli were interchanged and the fault-free circuit output voltage was virtually zero.

Single faults were introduced to the circuit and both voltages at all the nodes and currents through all the branches were calculated (i.e. access to all nodes and branches was assumed). The calculated voltages and currents were then manipulated separately by a program that implements the dc fault dictionary algorithm detailed above. A listing of the program is in Appendix 1.

The nodal voltages calculated under the two test conditions were manipulated under different tolerance conditions. A summary of these results for tolerances of 15mV, 150mV, 300mV, 500mV and 700mV are show in Table 4.1. The percentage of fault-coverage as used in Table 4.1 and in the rest of this thesis is defined by

Equation 4.1. If the simulation of a particular fault does not converge, that fault is then excluded from the calculation of fault-coverage as indicated by Equation 4.1.

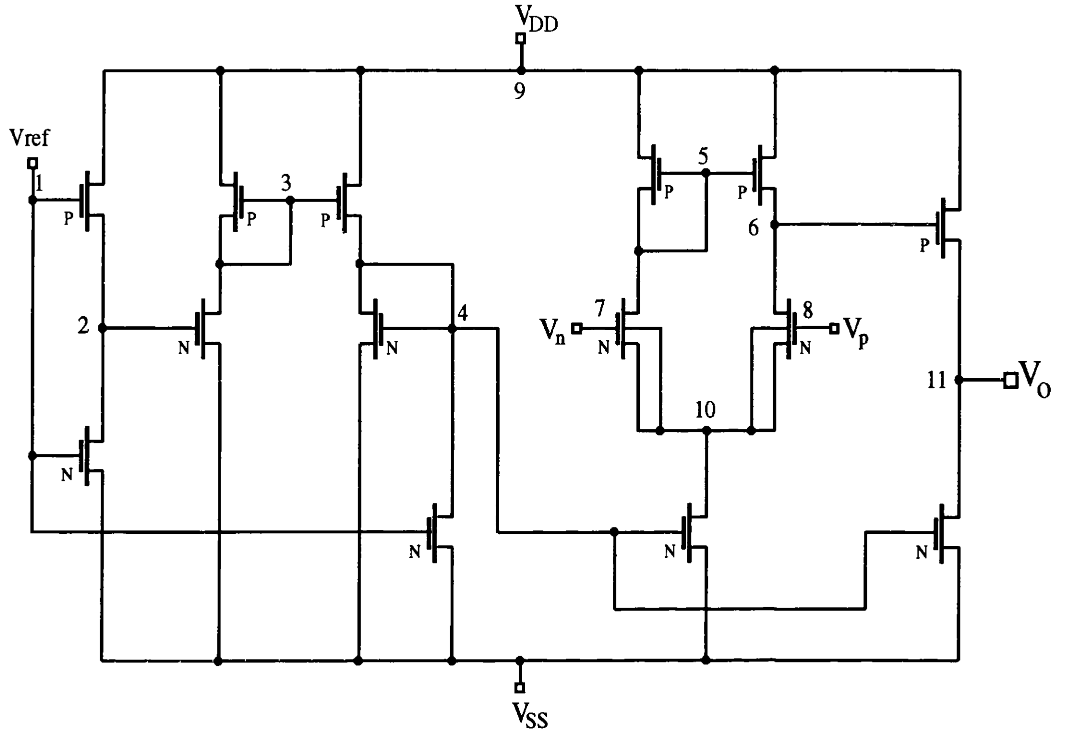


Figure 4.1: CMOS Comparator Circuit [2]

$$\textit{Fault-Coverage} = \frac{F_d}{(F - F_c)} * 100\% \qquad \ldots\ldots\ldots (4.1)$$

where

F : Number of faults simulated

F_d: Number of faults detected

F_c: Number of faults simulations that did not converge

In Table 4.1, for example, for a tolerance of 300mV access to seven circuit nodes is required to enable the detection of 37 faults and resulting in a fault-coverage of 69.8%. The requirement to access this high number of nodes would result in an increase in the chip area and complexity. Therefore, an attempt has been made to evaluate the potential fault-coverage of every circuit node. A summary of the results, for a tolerance of 300mV, is given in Table 4.2.

Table 4.1: Comparator Circuit Results Assuming Access to all Circuit Nodes

CMOS COMPARATOR CIRCUIT				
Tolerance in mV	No. Nodes Required	No. Faults Detected	No. Faults Isolated	% Fault-Coverage
15	7	40	11	75.5
150	7	37	9	69.8
300	7	37	9	69.8
500	7	37	9	69.8
700	7	37	9	69.8

It can be seen from Table 4.2 that a good number of faults can be detected by accessing a single node. Node 10, for example, achieves 60.4% fault-coverage. This is the highest fault-coverage that a single node of the comparator circuit can achieve. However, node 10 is an internal node and accessing it would mean an extra pad is needed. Node 11 on the other hand is the output node, therefore it is readily accessible even after chip packaging. However, node 11 can only detect 21 faults (i.e. 39.6% fault-coverage) which is rather poor.

To increase the fault-coverage, the number of test nodes must be increased. Therefore, access to two test nodes one of which is the output node was assumed. The voltages for every set of two nodes were manipulated and the results are summarized in Table 4.3. The table shows that combinations (5,11) and (10,11) are equivalent in terms of fault-coverage (i.e. each achieved 66.0% fault-coverage). However, combination (5,11) achieves a higher degree of isolation.

The process of systematically selecting the best set of node (or nodes) to incorporate with the output node in order to achieve the highest possible fault-coverage is discussed in subsection 4.1.2.

Table 4.2: Potential Fault-Coverage for Each Node of the Comparator Circuit, Assuming ΔV = 300mV

CMOS COMPARATOR CIRCUIT			
Node Number	No. Faults Detected	No. Faults Isolated	% Fault-Coverage
2	3	1	5.7
3	10	0	18.9
4	18	2	34.0
5	29	5	54.7
6	20	5	37.7
10	32	3	60.4
11	21	2	39.6

It was mentioned earlier that the fault dictionary approach applies to nodal voltages and branch currents. The calculated currents were manipulated in a manner similar to that of voltages. For a tolerance of 1μA a fault-coverage of 75.5% (i.e. 40 faults detected) was achieved.

Although manipulation of currents resulted in a higher fault-coverage than that when manipulating voltages, it is preferrable to deal with voltages because in practice they are easier to measure and the number of branch currents is usually more than the number of node voltages.

Table 4.3: Fault-Coverage for the Comparator Circuit When the Output Node and Another Circuit Node are Accessed Simultaneously. $\Delta V = 300mV$

CMOS COMPARATOR CIRCUIT			
Nodes Number	No. Faults Detected	No. Faults Isolated	% Fault-Coverage
2,11	24	3	45.3
3,11	28	2	52.8
4,11	29	3	54.7
5,11	35	7	66.0
6,11	25	6	47.2
10,11	35	5	66.0

4.1.1.2 Op-Amp Circuit

As in the case of the comparator circuit in the previous example, the CMOS op-amp circuit [3] shown in Figure 4.2 was simulated under fault-free and faulty conditions. A total of 70 single catastrophic fault conditions were introduced to the circuit.

Simulation conditions: the op-amp was tested in open loop configuration, because applying feedback may compensate for the effect of some faults and hence mask them. The voltage supplies V_{DD} and V_{SS} were set at +5.0 volts and -5.0 volts respectively. In the first test +1.0V was applied to the non-inverting terminal and the inverting terminal was connected to ground. The calculated output (node 1) for the first test under fault-free conditions was +2.4V. In the second test -1.0V was applied to the non-inverting terminal and the inverting terminal was kept at ground potential. The calculated output voltage for this test, under fault-free conditions, was -1.7V.

It is recognised that under the test conditions stated above, the fault-free output voltages of the op-amp should be close to the rail voltages. Since no faults could be found in the structure of the op-amp, it is most likely that the transistors sizes (width/length ratios) need to be optimized further. However, this behaviour of the op-amp have no effect on the subsequent analysis that will be carried out to evaluate the effectiveness of the testing strategy.

The strategy adopted for testing the op-amp circuit is similar to that applied for the comparator circuit. The primary objective is, as always, to achieve the highest possible fault-coverage while accessing the smallest number of nodes.

The results of manipulating the voltages of all the nodes under different tolerance values are depicted in Table 4.4. For a tolerance of 300mV Table 4.4 shows that 91.4% fault-coverage can be achieved (i.e. 64 faults detected). However, to achieve this fault-coverage, access to all but one of the internal circuit nodes is required.

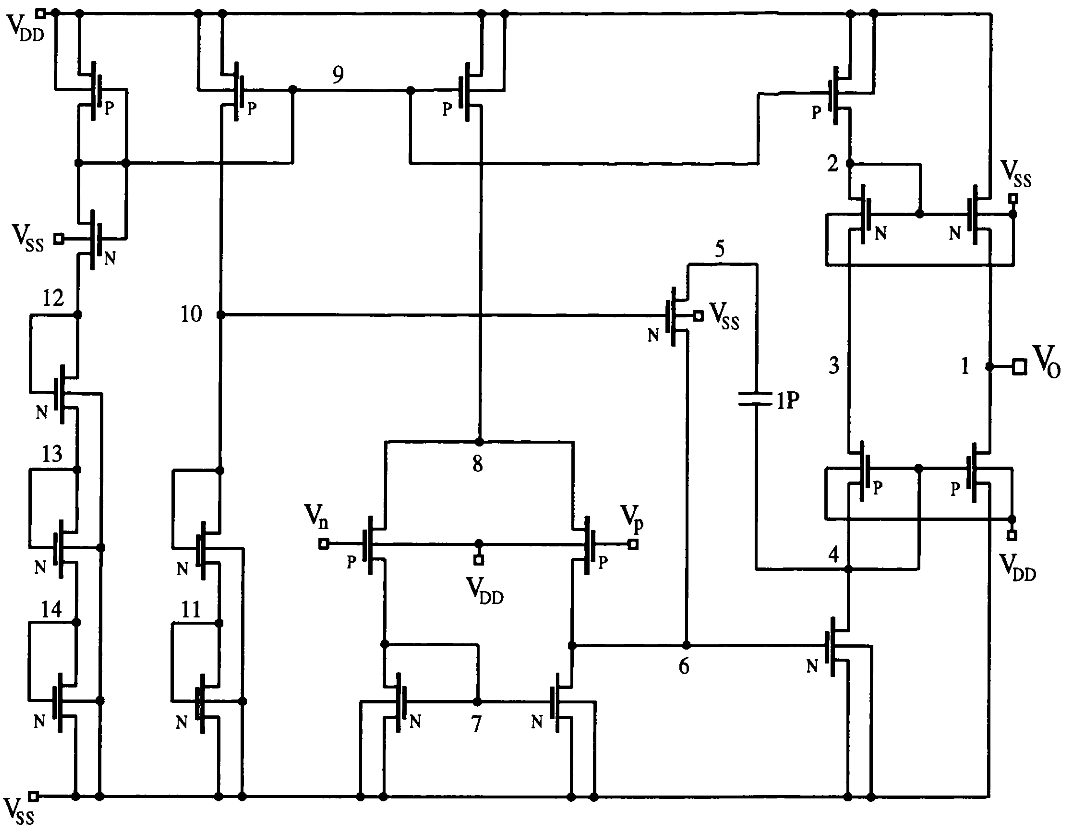


Figure 4.2: CMOS Operational Amplifier Circuit [3]

To reduce the number of nodes that need to be accessed, as in the case of the comparator, the potential fault-coverage for the individual nodes of the op-amp circuit was investigated. The results of manipulating the calculated voltages of the various nodes for a tolerance of 300mV are summarized in Table 4.5. The table shows that the output node (node 1) can achieve 77.7% fault-coverage. To increase the fault-coverage, as in the comparator case, the fault-coverage of sets of two nodes one of which is the output node was calculated. Table 4.6 illustrates the fault-coverage that can be achieved by each combination. From Table 4.6, combination (1,10) achieves 88.6% fault-coverage (i.e. 62 faults detected) which is higher than all other combinations.

Table 4.4: Results for the Op-Amp Circuit Assuming Access to all Circuit Nodes

CMOS OP-AMP CIRCUIT				
Tolerance in mV	No. Nodes Required	No. Faults Detected	No. Faults Isolated	% Fault-Coverage
15	13	68	21	97.1
150	13	64	19	91.4
300	13	64	17	91.4
500	14	64	14	91.4
700	13	64	13	91.4

Table 4.5: Potential Fault-Coverage for Each Node of the Op-Amp Circuit, assuming $\Delta V = 300mV$

CMOS OP-AMP CIRCUIT			
Node Number	No. Faults Detected	No. Faults Isolated	% Fault-Coverage
1	54	3	77.7
2	54	4	77.1
3	56	5	80.0
4	56	7	80.0
5	43	3	61.4
6	39	4	55.7
7	31	2	44.3
8	8	0	11.4
9	21	1	30.0
10	29	3	41.4
11	27	2	38.6
12	20	1	28.6
13	20	1	28.6
14	14	1	20.0

Table 4.6: Fault-Coverage for the Op-Amp Circuit When the Output Node and Another Circuit Node are Accessed Simultaneously. $\Delta V = 300mV$

CMOS OP-AMP CIRCUIT			
Nodes Number	No. Faults Detected	No. Faults Isolated	% Fault-Coverage
1,2	56	6	80.0
1,3	56	5	80.0
1,4	56	7	80.0
1,5	58	4	82.9
1,6	54	5	77.1
1,7	54	4	77.1
1,8	54	3	77.1
1,9	54	3	77.1
1,10	62	3	88.6
1,11	61	4	87.1
1,12	54	4	77.1
1,13	54	4	77.1
1,14	54	4	77.1

4.1.2 Selecting Test Nodes

To systematize the search for the smallest set of test nodes that would achieve an acceptable fault-coverage the following heuristic approach was adopted [4]. It starts by calculating a sensitivity factor called S_n for every node in the CUT. S_n is defined by Equation 4.2.

$$S_n = \sum_{f=1}^{f=F} (V_n - V_f)^2 \qquad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.2)$$

Where

V_n = nominal voltage of node n

V_f = faulty voltage of node n

F = number of faults introduced to the circuit

A high S_n implies that the particular node is sensitive to faults and is a likely candidate as a test node.

Once S_n for all the nodes is calculated a set of potential test nodes is formed. The set consists of the output node (or nodes) and the nodes having highest S_n. If this set of potential test nodes does not achieve an acceptable fault-coverage then the node having highest S_n is replaced with one having an S_n next to highest. The same procedure is repeated until a satisfactory fault-coverage is achieved. The output node/s is always included in the set of potential test nodes because, as was mentioned earlier, it is readily accessible.

The heuristic approach outlined above was applied in testing both the comparator and op-amp circuits shown in Figure 4.1 and Figure 4.2 respectively. For the comparator accessing nodes (Vo & 5) only achieved a fault-coverage of 66%. Similarly, accessing only nodes (Vo & 10) in the Op-Amp circuit achieved 88.6% fault-coverage. In both cases ΔV = 300mV is assumed.

4.2 Digital Modelling

In the digital modelling strategy, catastrophic faults in an analogue module are thought of as stuck-at faults in digital circuits. Two possible strategies to digitally test an analogue block were considered :

1- Digital-Logic Equivalent.

2- Functional K-Map.

4.2.1 Digital-Logic Equivalent

In this strategy [5], an analogue block is conceptually reduced to a combination of conventional digital blocks such as AND, NAND, OR, NOR etc. Then test patterns are developed for the digitized model. The model is checked by applying to the analogue block inputs two-state signals which are sufficiently low and high to guarantee digital switching of the analogue circuitry. The model is verified when the test patterns produce correct output signals on known good circuits.

When the digitized model of the analogue block has been verified, it may be combined with the digital blocks for the development of overall test patterns for the entire mixed-signal IC. The IC may then be tested as a unit on a digital tester.

The concept explained above was applied to an analogue comparator as shown in Figure 4.3 [5].

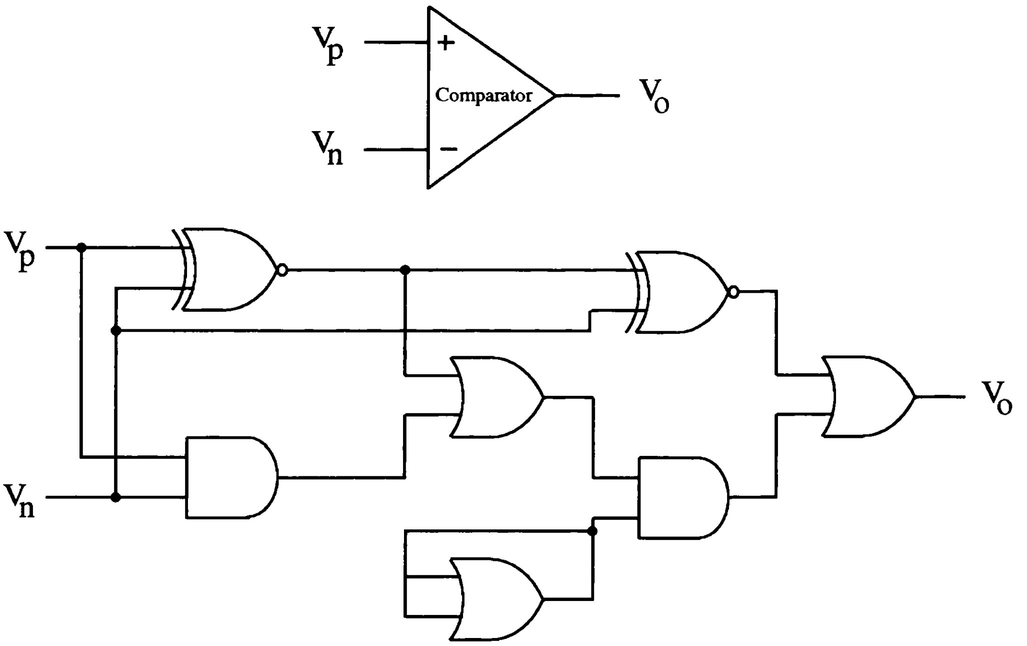


Figure 4.3: Equivalent Digital-Logic Circuit of an Analogue Comparator

4.2.2 Functional K-Map

This testing strategy treats an analogue module as a block (i.e. assumes that access is only available to its input and output terminals) and the dc response of the module is represented by a Karnaugh map (K-map) [4].

The testing algorithm consists of the following steps :

1- The fault-free circuit is simulated and the response to a suitable set of input signals is calculated.

2- The circuit operation in step-1 is translated into a K-map consisting of 1s and 0s. To do this a threshold voltage (V_t) is established so that voltage values above it are considered as 1s and those below it as 0s.

3- The digital function represented by the K-map in step-2 is derived.

4- The tests necessary to detect stuck-at faults in the digital function in step-3 are then derived.

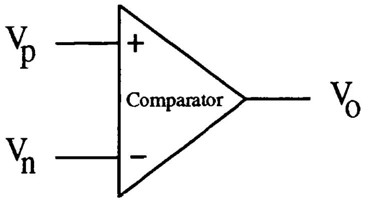
5- The tests in step-4 are translated back to dc voltages and applied to the input terminals of the analogue module to be tested.

The algorithm outlined above was applied to detect catastrophic faults in the comparator and op-amp circuits illustrated in Figure 4.1 and Figure 4.2 respectively.

4.2.2.1 Comparator Circuit

A block diagram of the comparator in Figure 4.1, the results of the fault-free operation, and the K-map when a threshold voltage of V_t = 4.0V is assumed are illustrated in Figure 4.4.

Comparator Fault-Free Operation Table				
Test	Code	$V_{p\ \text{volt}}$	$V_{n\ \text{volt}}$	$V_{o\ \text{volt}}$
T0	00	-3.0	-3.0	0.00
T1	01	-3.0	+3.0	0.00
T2	10	+3.0	-3.0	4.97
T3	11	+3.0	+3.0	3.26



$$\begin{array}{cc|cc} & & \multicolumn{2}{c}{V_p} \\ & & 0 & 1 \\ \hline V_n & 0 & 0 & 1 \\ & 1 & 0 & 0 \end{array}$$

Threshold Voltage: $V_t = 4.0\text{V}$

Boolean Function from K-Map: $V_o = V_p * \overline{V_n}$

Required Tests: T0, T2 & T3

Figure 4.4: Functional K-Map Testing of the Comparator Circuit

The digital function represented by the K-map in Figure 4.4 is a logical AND between V_p and $\overline{V}_n$ (i.e. Vout = $V_p * \overline{V}_n$). If analogue faults are considered as digital stuck-at faults then this equivalent digital function would only require 3-tests to detect those stuck-at faults. The required tests are : (T0, T2 & T3).

To check the validity of the above discussion the comparator circuit was simulated under fault-free and faulty conditions. The same 53 single catastrophic fault conditions, used in evaluating the dc fault dictionary approach, were introduced to the circuit one at a time. For each of the 54 simulations the 4-tests in Figure 4.4 (i.e. T0 T1 T2 T3) were applied.

To determine the faults detected by each of the 4-tests, a window representing the allowed tolerance was placed around the nominal fault-free voltage of each test. The objective of the window is to enable the distinction between good and faulty circuits. A fault is detectable by a particular test if the voltage value at the output

]

node (i.e. the test node) falls outside the window of the test. Otherwise, the fault is considered undetectable by that test. The results of the 4-tests are shown in Table 4.7. Inspection of the faults detected in Table 4.7 reveals that test T1 is a redundant test because it is covered by the other 3-tests. This confirms the hypothesis made earlier regarding treating analogue faults as digital stuck-at faults. From Figure 4.4 tests (T0 T2 T3) result in a 70% fault-coverage.

Table 4.7: Faults Detected by 4-Tests Applied to the Comparator Circuit

CMOS COMPARATOR CIRCUIT			
Test Name	Test Code	No. Faults Detected	List of Faults Detected
T0	00	16	1 5 9 12 14 16 19 23 27 28 31 34 39 42 43 50
T1	01	7	12 15 16 19 34 42 50
T2	10	11	15 19 20 21 22 32 33 34 42 46 51
T3	11	31	5 9 12 13 14 16 17 18 20 21 22 27 31 32 33 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53

To analyze the results in Table 4.7 further, a computer program that determines which one of the 4-tests detects a particular fault with the highest degree of confidence was implemented. The degree of confidence is defined as the magnitude of the difference between the nominal fault-free voltage of a test and the value generated by a particular fault. The higher the difference between the faulty and fault-free values the higher the degree of confidence in the test.

The program output indicated that for detection with high degree of confidence all 4-tests would be required. However, the fault-coverage did not increase.

The uniqueness of tests (T0 T2 T3) was also investigated. To do that the various combinations of 3-tests were formed and the results are summarized in Table 4.8. This table shows that tests sets (T0 T2 T3) and (T0 T1 T3) are equivalent in terms of fault-coverage. However test set (T0 T2 T3) detects 8 faults with higher confidence than test set (T0 T1 T3). Therefore, test set (T0 T2 T3) is the best test set to achieve high fault-coverage with high confidence.

Table 4.8: Sets of 3-Tests for the Comparator Circuit

CMOS COMPARATOR CIRCUIT		
Test Sets	No. Faults Detected	% Fault-Coverage
(T0 T2 T3)	37	70
(T0 T1 T3)	37	70
(T0 T1 T2)	23	43
(T1 T2 T3)	34	64

4.2.2.2 Op-Amp Circuit

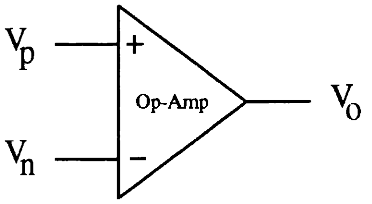
An approach similar to that in the previous section was used to test the op-amp circuit. The block diagram of the op-amp in Figure 4.2, the results of fault-free operation, and the K-map when a threshold voltage of V_t = 1.0V is assumed are shown in Figure 4.5.

The digital function represented by the K-map in Figure 4.5 is again a logical AND between V_p and $\bar{V}_n$ (i.e. same as the comparator). Therefore, the required tests are (T0, T2 & T3).

The results of the 4-tests, when the 70 single catastrophic faults, synthesized in subsection 4.1.1.2, were introduced are detailed in Table 4.9. Inspection of the results shows again that test T1 is a redundant one. Applying tests (T0 T2 T3) results in 80% fault-coverage.

Analysis of the results in Table 4.9 showed that to detect with a high degree of confidence all 4-tests are required. But again the fault-coverage did not increase.

Op-Amp Fault-Free Operation Table				
Test	Code	$V_{p\ \mathrm{volt}}$	$V_{n\ \mathrm{volt}}$	$V_{o\ \mathrm{volt}}$
T0	00	-3.0	-3.0	-0.98
T1	01	-3.0	+3.0	-1.93
T2	10	+3.0	-3.0	2.77
T3	11	+3.0	+3.0	0.31



		V_p 0	1
V_n	0	0	1
	1	0	0

Threshold Voltage: V_t = 1.0V

Boolean Function from K-Map: $V_o = V_p * \overline{V_n}$

Required Tests: T0, T2 & T3

Figure 4.5: Functional K-Map Testing of the Op-Amp Circuit

Table 4.9: Faults Detected by 4 Tests Applied to the Op-Amp Circuit

CMOS OP-AMP CIRCUIT			
Test Name	Test Code	No. Faults Detected	List of Faults Detected
T0	00	27	1 3 4 5 6 10 11 12 13 17 18 20 21 22 23 24 29 30 31 32 33 34 35 36 37 38 40
T1	01	18	1 5 10 11 13 17 18 19 20 21 22 32 33 34 35 39 57 58
T2	10	15	1 2 12 13 16 19 28 31 37 38 39 40 43 47 56
T3	11	51	1 2 3 4 5 6 9 12 13 16 19 20 21 22 23 24 27 28 29 30 31 32 33 34 35 36 37 38 40 41 42 43 44 45 46 47 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70

The uniqueness of tests (T0 T2 T3) was also examined, and the results are in Table 4.10. It shows that test sets (T0 T2 T3), (T0 T1 T3) and (T1 T2 T3) achieved the same fault-coverage. However, set (T0 T2 T3) achieves a higher degree of confidence. Therefore, (T0 T2 T3) is the best test set.

The potential fault-coverage of test sets having only 2-tests was also explored, and the results are summarized in Table 4.11. It shows that set (T1 T3) achieves 80% fault-coverage, however the confidence level decreased substantially.

Table 4.10: Sets of 3-Tests for the Op-Amp Circuit

CMOS OP-AMP CIRCUIT		
Test Sets	No. Faults Detected	% Fault-Coverage
(T0 T2 T3)	56	80
(T0 T1 T3)	56	80
(T0 T1 T2)	37	53
(T1 T2 T3)	56	80

Table 4.11: Sets of 2-Tests for the Op-Amp Circuit

CMOS OP-AMP CIRCUIT		
Test Sets	No. Faults Detected	% Fault-Coverage
(T0 T1)	45	64
(T0 T2)	35	50
(T0 T3)	55	79
(T1 T2)	29	41
(T1 T3)	56	80
(T2 T3)	52	74

4.3 Logical Decomposition

An efficient method for testing analogue networks by logical decomposition was first presented in [6]. In this testing strategy, a nodal decomposition of the network into smaller uncoupled subnetworks is carried out [7]. The measurement nodes (or accessible nodes) must include the nodes of decomposition. These nodes are referred to as D-nodes. The voltage measurements are used to isolate the faulty subnetworks. The incidence relations between subnetworks are checked against KCL (Kirchhoff's current law) to determine the status of those D-nodes. Logic analysis is then carried out to identify faulty subnetworks. This testing method imposes no restrictions on the type or magnitude of the test signals that can be applied.

Based on voltage measurements, the currents associated with the D-nodes are computed and checked against KCL. A D-node is determined as fault-free if the currents associated with this node satisfy KCL. Otherwise, the node is faulty. It will be appreciated that if the currents associated with a fault-free device satisfy KCL, the sum of such currents should be zero. In practice, however, a reasonably small tolerance term ε is used as the bound. In other words, if the sum of the currents is within the bounds of a pre-defined tolerance term, then the node is fault-free, otherwise the node is faulty.

The decomposition strategy fault location algorithm is summarized below :

Step 1: Decompose the network at all accessible nodes by logically breaking the connections at the nodes to obtain smallest possible uncoupled subnetworks.

Step 2: Using the node voltages measured at the accessible nodes and nominal parameter values of subnetworks, compute the currents flowing into accessible nodes from the subnetworks.

Step 3: If the computed currents satisfy KCL at an accessible node, all the subnetworks connected to this node in the original network are considered fault-free. Otherwise at least one of these subnetworks is faulty.

Step 4: Use the fault-free subnetworks to determine the external current of the faulty subnetworks.

Step 5: If faults at the desired level of decomposition are located, then process stops; otherwise steps 1-4 are repeated.

Identification of the faulty subnetwork can be achieved systematically by deriving a logical diagnostic function (LDF) for the complete network with a logical variable σ, which takes a binary value :

$\sigma = 1$ if the subnetwork is fault-free
$\sigma = 0$ if the subnetwork is faulty

A test T_{JK} is the result of applying KCL at a certain D-node. If the test at a particular D-node is a pass (i.e. Σ KCL = 0), then :

$$T_{Jt} = \sigma_{j1} \bigcap \sigma_{j2} \bigcap \ldots\ldots \bigcap \sigma_{jk} \qquad \ldots\ldots\ldots\ldots (4.3)$$

where

Jt = { j1, j2, ..., jk }.

J refers to the subnetwork S_j; K is the number of subnetworks involved in the test.

If the test is a fail (i.e.Σ KCL $\neq$ 0), then :

$$\overline{T}_{Jt} = \overline{\sigma}_{j1} \bigcup \overline{\sigma}_{j2} \bigcup \ldots\ldots \bigcup \overline{\sigma}_{jk} \qquad \ldots\ldots\ldots\ldots (4.4)$$

The Logical Diagnostic Function (LDF) is defined as :

$$T = T_{Jt} \bigcap \bar{T}_{Jt} \qquad \dots\dots\dots\dots\dots\dots \quad (4.5)$$

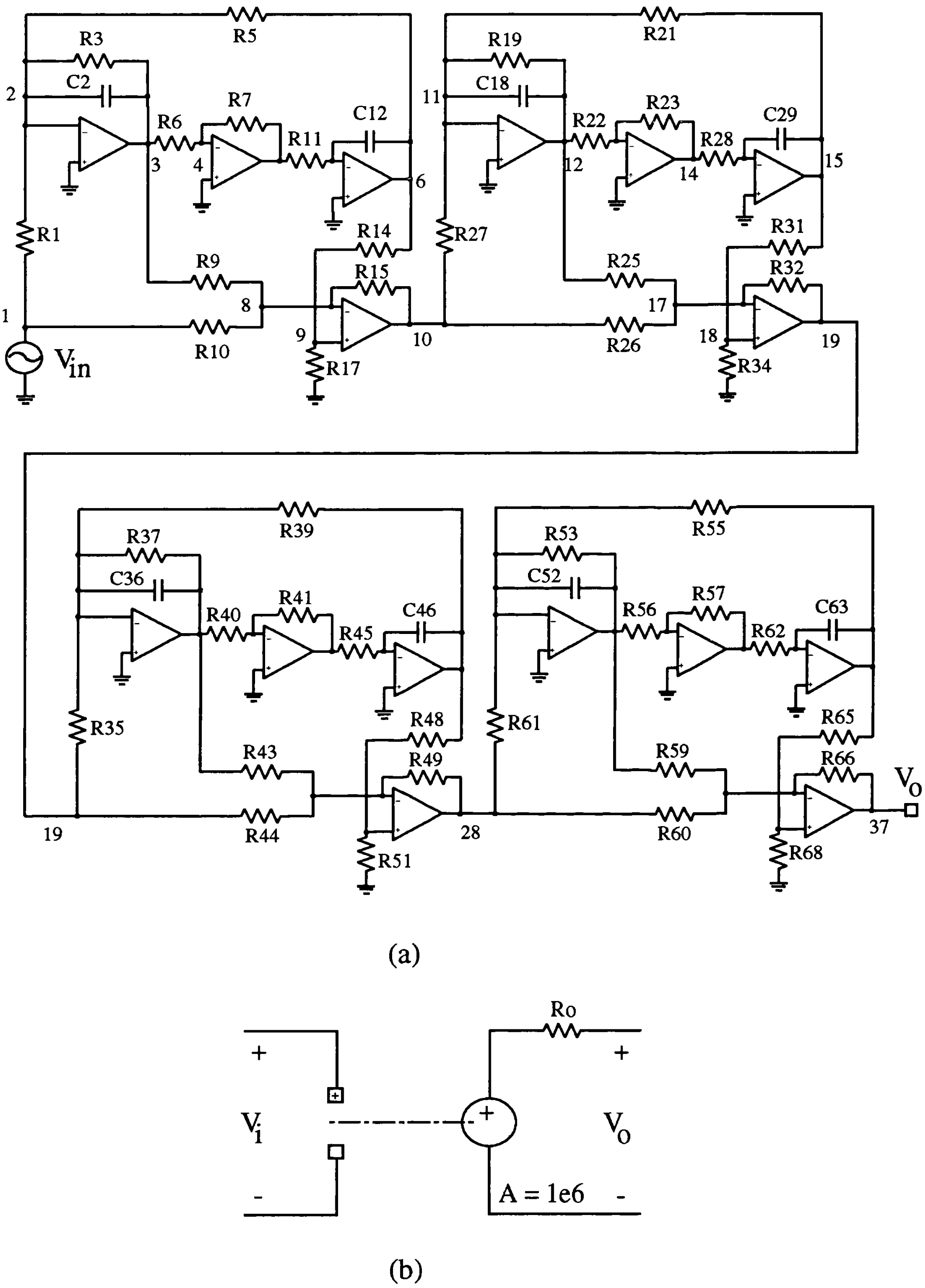
which means that LDF is the logical product of the pass and fail tests. LDF is manipulated by applying ordinary Boolean algebraic principles. A σ in LDF represents a fault-free subnetwork while a $\bar{\sigma}$ represents a faulty subnetwork. If a subnetwork is not represented in LDF nothing is assumed about its state.

Logical decomposition testing strategy can handle multiple catastrophic and soft faults. It is also capable of handling linear as well as non-linear networks. In the case of non-linear networks however, it is best to keep the number of non-linear elements in a subnetwork to a minimum. The strategy was applied to linear, non-linear and mixed-signal networks.

In the following subsections the decomposition strategy described above is applied to a linear and a non-linear circuits. The applicability of the strategy to analogue and mixed-signal MOS integrated circuits is then evaluated.

4.3.1 Linear Case

The active filter network in Figure 4.6(a) [6] was adopted to test the logical decomposition algorithm in the linear case. A list of the filter components values is presented in Table 4.12. To simplify the simulation task each op-amp in Figure 4.6(a) was replaced by the model in Figure 4.6(b). The network was tested under AC conditions with V_{in} = 1.0 sin ($2\pi ft$) where f = 1 KHz. The initial abstract decomposition of the active filter network is illustrated in Figure 4.7. A number of catastrophic and soft fault conditions, some of which are multiple (i.e. more than one fault present at a time), were introduced to the network. In some cases, especially when multiple faults are present, the network may need to be decomposed further or a different set of D-nodes chosen to locate all the fault. Details of the analysis to detect and locate two of the fault conditions introduced are detailed below.



(a)

(b)

Figure 4.6: (a) Active Filter Network, (b) Op-Amp Model Used

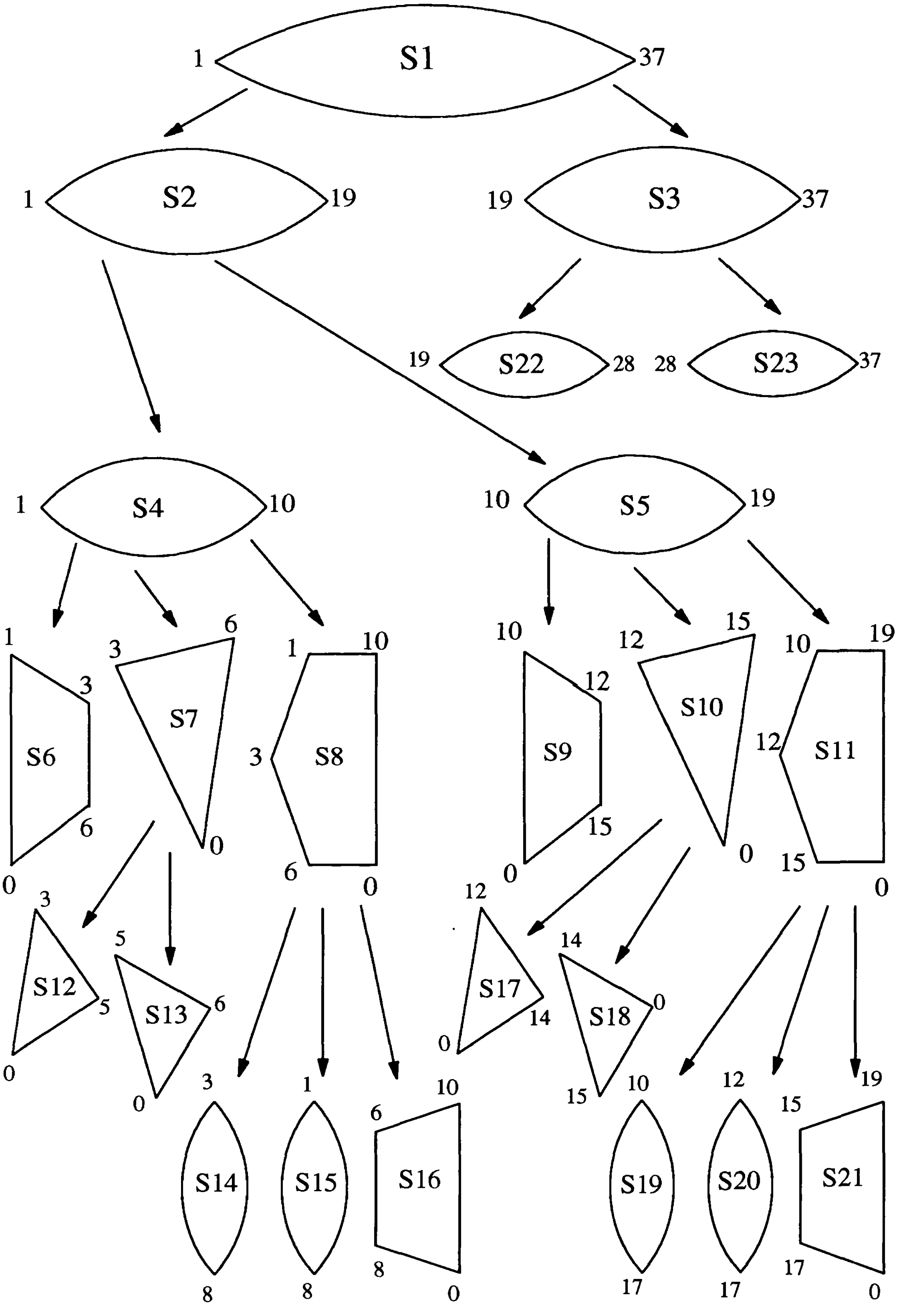


Figure 4.7: Decomposed Active Filter Network in Abstract Form

Table 4.12: Active Filter Components Values

ACTIVE FILTER	
Components	Values
R1, R25	0.182
R3, R37	1.57
R5, R11, R39, R45	2.64
R6, R7, R40, R41	10.0
R9, R43	100.0
R10, R44	11.10
R14, R48	5.41
R15, R17, R49, R51	1.00
R19, R53	4.84
R21, R28, R55, R62	2.32
R22, R23, R56, R57	10.0
R25, R59	500.0
R26, R60	111.1
R27, R61	1.14
R31, R65	72.4
R32, R34, R66, R68	10.0
C2, C12, C36, C46	0.01
C18, C29, C52, C63	0.01
* Note: Resistors in (KΩ) & Capacitors in (μF)	

Fault-1: R1 changed from 0.182KΩ to 0.1KΩ

LDF is given as :

$$T = \bar{T}_{6,8} \cap \bar{T}_{6,7,8} \cap T_{8,9,11} \cap T_{9,10,11} \cap T_{3,11}$$

$$= (\bar{\sigma}_6 + \bar{\sigma}_8)\,(\bar{\sigma}_6 + \bar{\sigma}_7 + \bar{\sigma}_8)\,(\sigma_8\ \sigma_9\ \sigma_{11})\,(\sigma_9\ \sigma_{10}\ \sigma_{11})\,(\sigma_3\ \sigma_{11})$$

$$= \sigma_3\ \bar{\sigma}_6\ \sigma_8\ \sigma_9\ \sigma_{10}\ \sigma_{11}$$

$\therefore$ Fault-1 is located in subnetwork S_6

Fault-2: R1 0.182KΩ → 0.1KΩ, R23 10.0KΩ → 6.0KΩ & C2 0.01μF → 0.02μF

LDF is given as :

$$T = \bar{T}_{6,15} \cap \bar{T}_{6,12,14} \cap T_{12,13} \cap T_{14,15,16} \cap \bar{T}_{9,10,11} \cap \bar{T}_{11,22} \cap T_{22,23}$$

$$= (\bar{\sigma}_6 + \bar{\sigma}_{15}) (\bar{\sigma}_6 + \bar{\sigma}_{12} + \bar{\sigma}_{14}) (\sigma_{12}\, \sigma_{13}) (\sigma_{14}\, \sigma_{15}\, \sigma_{16}) (\bar{\sigma}_9 + \bar{\sigma}_{10} + \bar{\sigma}_{11}) (\bar{\sigma}_{11} + \bar{\sigma}_{22})(\sigma_{22}\, \sigma_{23})$$

$$= \bar{\sigma}_6\, \bar{\sigma}_{11}\, \sigma_{12}\, \sigma_{13}\, \sigma_{14}\, \sigma_{15}\, \sigma_{16}\, \sigma_{22}\, \sigma_{23}$$

∴ Fault-2 is located in subnetworks S6 and S11. To locate the third faulty subnetwork with C2, the network must be decomposed further.

4.3.2 Non-Linear Case

To test the decomposition method for the non-linear case it was applied to the video amplifier circuit in Figure 4.8 [6]. The abstract decomposition of the video circuit is illustrated in Figure 4.9. The network was tested under dc conditions with V_{CC} = 28V and V_{EE} = -28V. A total of 15 fault conditions were simulated and all were detected and subsequently located. The process of locating two of the faults introduced is detailed below.

Fault-1: R10 0.078KΩ → 7.8KΩ

LDF is given as :

$$T = \bar{T}_{3,5,6,7} \cap T_{2,3} \cap T_{3,4,5,6} \cap T_{5,8} \cap T_{6,9}$$

$$= (\bar{\sigma}_3 + \bar{\sigma}_5 + \bar{\sigma}_6 + \bar{\sigma}_7) (\sigma_2\, \sigma_3) (\sigma_3\, \sigma_4\, \sigma_5\, \sigma_6) (\sigma_5\, \sigma_8) (\sigma_6\, \sigma_9)$$

$$= \sigma_2\ \sigma_3\ \sigma_4\ \sigma_5\ \sigma_6\ \bar{\sigma}_7\ \sigma_8\ \sigma_9$$

$\therefore$ Fault-1 is in subnetwork S_7.

<u>Fault-2:</u> Q1 E-Open, Q3 C-Open and Q4 CB-Short

LDF is given as :

$$T = \bar{T}_{3,5,6,7} \cap \bar{T}_{2,3} \cap \bar{T}_{3,4,5,6} \cap \bar{T}_{5,8} \cap \bar{T}_{6,9} \cap T_{3,4,7,8,9}$$

$$= (\bar{\sigma}_3 + \bar{\sigma}_5 + \bar{\sigma}_6 + \bar{\sigma}_7)\ (\bar{\sigma}_2 + \bar{\sigma}_3)\ (\bar{\sigma}_3 + \bar{\sigma}_4 + \bar{\sigma}_5 + \bar{\sigma}_6)\ (\bar{\sigma}_5 + \bar{\sigma}_8)\ (\bar{\sigma}_6 + \bar{\sigma}_9)$$
$$(\sigma_3\ \sigma_4\ \sigma_7\ \sigma_8\ \sigma_9)$$

$$= \bar{\sigma}_2\ \sigma_3\ \sigma_4\ \bar{\sigma}_5\ \bar{\sigma}_6\ \sigma_7\ \sigma_8\ \sigma_9$$

$\therefore$ Fault-2 is in subnetworks S_2, S_5 and S_6, indicating the presence of 3 faults.

4.3.3 MOS Analogue and Mixed-Signal ICs

The logical decomposition testing strategy presented above was applied to a number of CMOS analogue and mixed-signal circuits, comprising the comparator (Figure 4.1), the op-amp (Figure 4.2) and some combinational logic gates. However, due to the fact that currents in CMOS devices are very small it was very difficult to distinguish between fault-free and faulty subnetworks when applying KCL at the nodes of decomposition. Even by avoiding decomposition at the transistors gates, KCL was still close to zero at the decomposition nodes although some of the subnetworks were faulty. Therefore, it was concluded that this testing strategy is not suitable for CMOS devices [8].

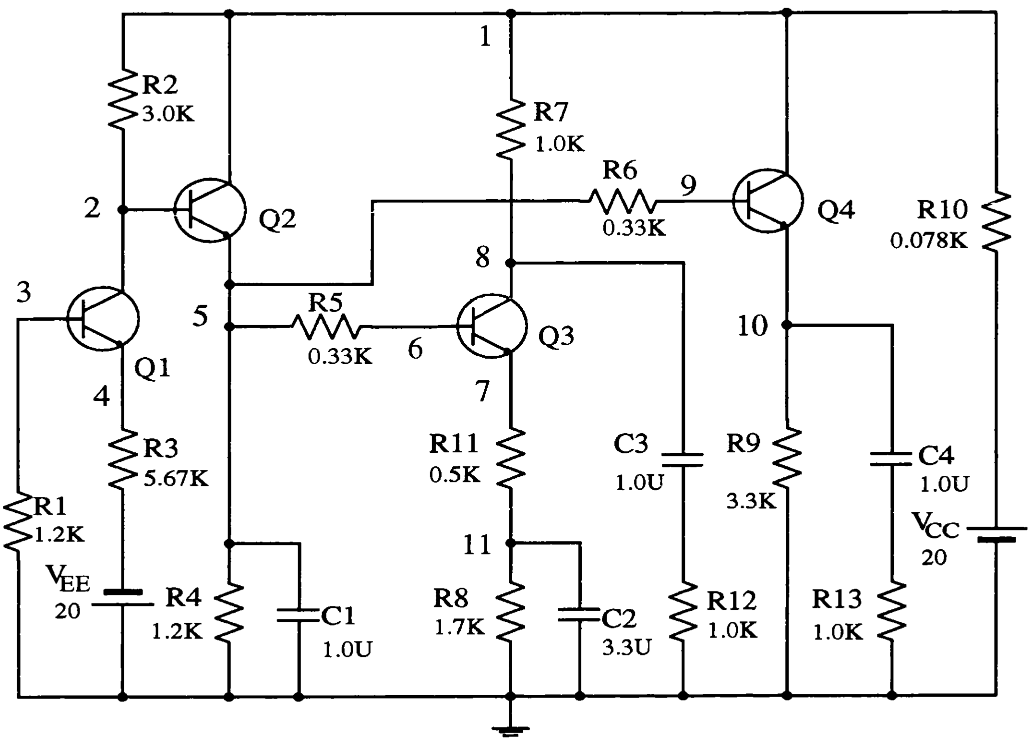


Figure 4.8: Video Amplifier Circuit

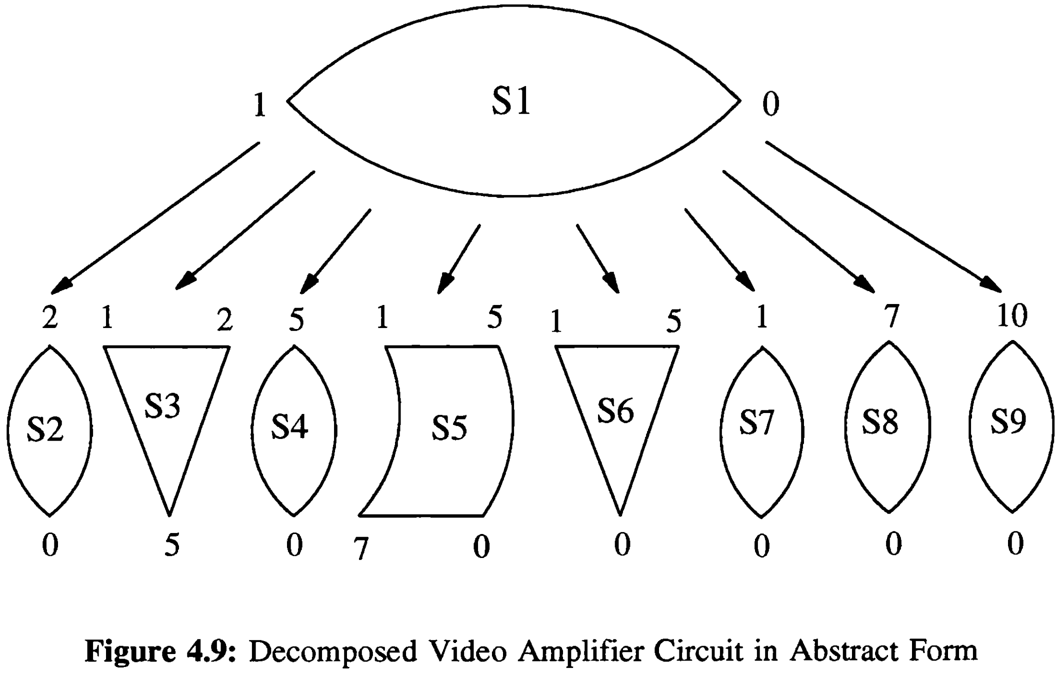


Figure 4.9: Decomposed Video Amplifier Circuit in Abstract Form

4.4 Summary

The dc fault dictionary approach achieved a good level of fault-coverage for the comparator circuit and a high fault-coverage for the op-amp circuit. The approach is more suitable for production (i.e. go/no-go) testing than diagnostic testing. This is due to the high number of accessible nodes required for the later.

The digital-logic equivalent technique is an ideal one for testing analogue and mixed-signal ICs using a digital tester. However, good knowledge of the behaviour of the analogue block is required, and translating that behaviour to an equivalent digital one is a difficult task.

Functional K-map testing achieved fault-coverage levels, for both the comparator and op-amp circuits, which are comparable to those of the fault dictionary while only accessing the external nodes. The technique requires less computational effort than the fault dictionary.

Logical decomposition is an efficient testing strategy that places no restrictions on the type of CUT or the signals handled. However, the strategy is not suitable for CMOS ICs due to the extremely small amounts of currents in these ICs.

4.5 References

[1] W. Hochwald and J.D. Bastian, "A D Approach for Analog Fault Dictionary Determination", IEEE Trans. Circuits and Systems, Vol. CAS-26, No. 7, pp. 523-529, July 1979.

[2] J. Compton, Silicon Microsystems Ltd, Malmesbury, U.K., Private Communications, June 1988.

[3] C. Toumazou, ASCOT Progress Report, Dept. of Electrical Engineering, Imperial College of Science, Technology and Medicine, London, U.K., September 1988.

[4] M.A. Al-Qutayri and P.R. Shepherd, "Testing Approaches for Mixed-Mode (Analogue/Digital) Integrated Circuits", Proc. Silicon Design Conference, pp. 37-45, London, November 1989.

[5] D. Brown and J. Damianos, "Method for Simulation and Testing of Analog-Digital Circuits", IBM Tech. Disclosure Bull., Vol. 25, pp. 6367-6368, 1983.

[6] A.E. Salama, J.A. Strazky and J.W. Bandler, "A Unified Decomposition Approach for Fault Location in Large Analog Circuits", IEEE Trans. Circuits and Systems, Vol. CAS-31, pp. 609-622, July 1984.

[7] H. Happ, "Diakoptics - The Solution of Problem by Tear", Proc. of IEEE, Vol. 62, July 1974.

[8] M.A. Al-Qutayri and P.R. Shepherd, "On the Testing of Mixed-Mode Integrated Circuits", Journal of Semicustom ICs, Vol. 7, No. 4, pp. 32-39, June 1990.

CHAPTER FIVE

Time-Domain Testing Technique

In the previous chapter, with the exception of the logical decomposition approach, the techniques presented test a circuit-under-test (CUT) under dc conditions. This means that some components, such as capacitors, cannot be tested and the dynamic behaviour of the CUT is not evaluated.

This chapter presents a time-domain testing strategy that is applicable to analogue and mixed-signal ICs, and can be implemented on a digital tester. Being time based, the technique overcomes the limitations of the dc strategies discussed earlier.

An important characteristic of the technique is that it tests a mixed-signal IC as one complete entity. Hence, the requirement to partition a mixed-signal CUT to allow mode specific testing is eliminated.

Both voltage at the output nodes or pins and supply current are calculated and processed to evaluate their effect on fault-coverage. The technique is applied to three circuit examples with varying degrees of complexity to evaluate its effectiveness.

5.1 The Time-Domain Approach

The time-domain testing strategy is based on the excitation of the CUT with a sequence of pulses, and subsequent measurement of the transient response at the output node/s [1-4]. Both the transient voltage at the output node/s and the transient current Idd of the supply current are measured. The transient current (Idd) testing technique is similar to the IDDQ testing technique [5-7], except that IDDQ testing is performed under static conditions while Idd testing is performed under dynamic conditions.

A major advantage of the time-domain approach is that the transient response contains all the necessary information about the CUT. This information can then be processed by applying digital signal processing (DSP) and other techniques to extract measures of the various CUT parameters. In this thesis, the primary objective when processing the transient response data is to establish whether the CUT is faulty or not (i.e. Go / No-Go), and not to locate and identify the type of fault present if any.

Since the test sequence to be applied consists of pulses, there are two basic parameters that can be varied in a pulse; amplitude and width. Testing by applying a sequence of pulses with varying amplitude was discussed in [8], where the author used the analogue network under-test poles and, therefore, element values to determine the various amplitude levels of the input test sequence. Varying the amplitude of the pulses within a test sequence virtually rules out the use of a digital tester. On the other hand, varying the width of a pulse is easy to achieve with a digital tester.

The type of test sequence to be applied to a CUT is a pseudo random binary sequence (PRBS). The PRBS is chosen because it can be readily generated by a digital tester, such sequences have well defined properties and can be used to extract the impulse response of an analogue CUT as discussed below. If the CUT is a mixed-signal network, the response generated by applying a PRBS test signal can be used as a signature to characterize the network under-test.

If the analogue CUT is represented by the basic block diagram in Figure 5.1, where h(t), x(t) and y(t) are the impulse response, the input and the output of the block respectively.

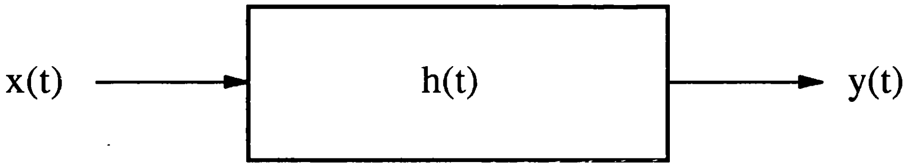


Figure 5.1: A Block Diagram of an Analogue Circuit

The output signal y(t) is the convolution of the x(t) and h(t),

$$y(t) = \int_{-\infty}^{\infty} h(\nu)\, x(t-\nu)\, d\nu \qquad \cdots\cdots\cdots\cdots (5.1)$$

Cross-correlating y(t) with x(t) gives,

$$\phi_{XY}(\tau) = \int_{-\infty}^{\infty} x(t)\, y(t+\tau)\, dt \qquad \cdots\cdots\cdots (5.2)$$

substituting for y(t) in Equ. 5.2 from Equ. 5.1 gives,

$$\phi_{XY}(\tau) = \int_{-\infty}^{\infty} x(t) \int_{-\infty}^{\infty} h(\nu)\, x(t+\tau-\nu)\, d\nu\, dt$$

$$\phi_{XY}(\tau) = \int_{-\infty}^{\infty} h(\nu)\, \phi_{XX}(\tau-\nu)\, d\nu \qquad \cdots\cdots (5.3)$$

where $\phi_{XX}(\tau\text{-}\nu)$ is the auto-correlation function of x(t) and τ and ν are time shift constants.

Equation 5.3 above states that the cross-correlation between x(t) and y(t) is the convolution of the impulse response h(ν) with the auto-correlation of x(t). If x(t) is a white noise signal, then its auto-correlation function is a delta function and substituting this in Equation 5.3 gives

$$\phi_{XY}(\tau) = \int_{-\infty}^{\infty} h(\nu)\ \delta(\tau-\nu)\ d\nu$$

$$\phi_{XY}(\tau) = h(\tau) \qquad \dots\dots\dots\dots\dots (5.4)$$

because $\delta(\tau\text{-}\nu) = 1$ at $\tau=\nu$ and 0 anywhere else.

If x(t) is not a white noise signal, Equation 5.3 can still lead to a useful result. Taking the Fourier transform of both sides of Equation 5.3 gives

$$\int_{-\infty}^{\infty} \phi_{XY}(\tau)\ \exp(-j\omega\tau)\ d\tau = \int_{-\infty}^{\infty} \exp(-j\omega\tau) \int_{-\infty}^{\infty} h(\nu)\ \phi_{XX}(\tau-\nu)\ d\nu\ d\tau$$

$$S_{XY}(\omega) = H(\omega)\ S_{XX}(\omega) \qquad \dots\dots\dots\dots (5.5)$$

$$H(\omega) = S_{XY}(\omega)\ /\ S_{XX}(\omega) \qquad \dots\dots\dots (5.6)$$

where $S_{xy}(\omega)$ is the cross-power spectral density of x(t) and y(t), $S_{XX}(\omega)$ is the auto-power spectral density of x(t), and H(ω) is the transfer function. The impulse response can be extracted by taking the inverse Fourier transform of Equation 5.6.

The generation of the white noise input signal required by Equation 5.4 is impossible, because white noise contains equal amounts of all frequencies which means an infinite bandwidth is needed to generate it. The PRBS signals, however, have very good randomness properties and are a very good approximation to white noise. The simplest way to generate a PRBS is by a maximum-length linear shift register with an appropriate modulo-2 feedback function [9], similar to the one in Figure 3.19. If a PRBS is long then the probability of finding a 1 is almost equal to that of finding a 0. The auto-correlation function of a PRBS signal is triangular [10] with a base-width equal to two clock periods as shown in Figure 5.2. This is a very good approximation of the delta function required by Equation 5.4.

The relationship between the time and frequency domain of a PRBS signal is illustrated in Figure 5.3. The bit interval T should be selected so that the spectral components of the PRBS fall in the desired location of the CUT response, such as the corner frequency of a filter.

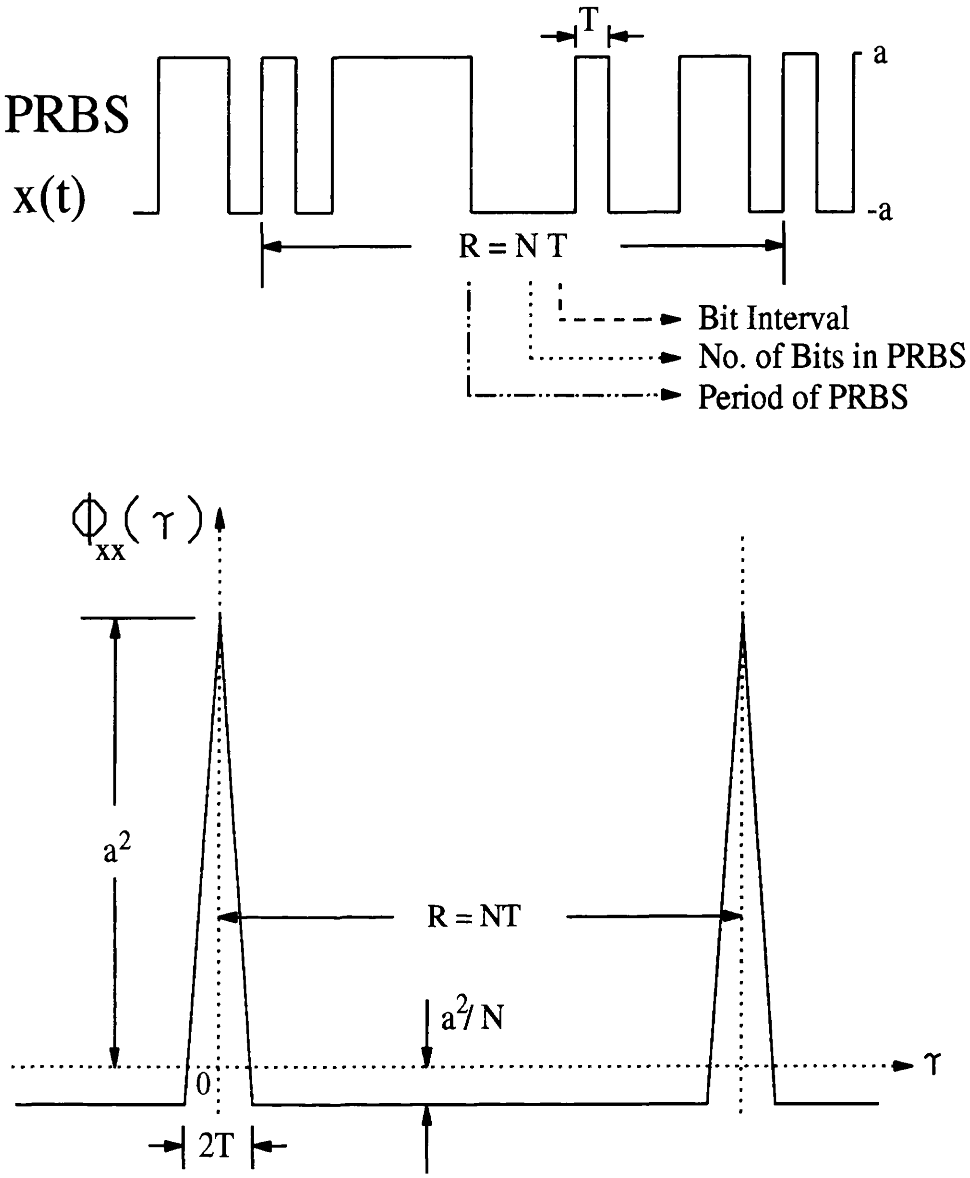


Figure 5.2: Auto-Correlation Function of an N Bits PRBS

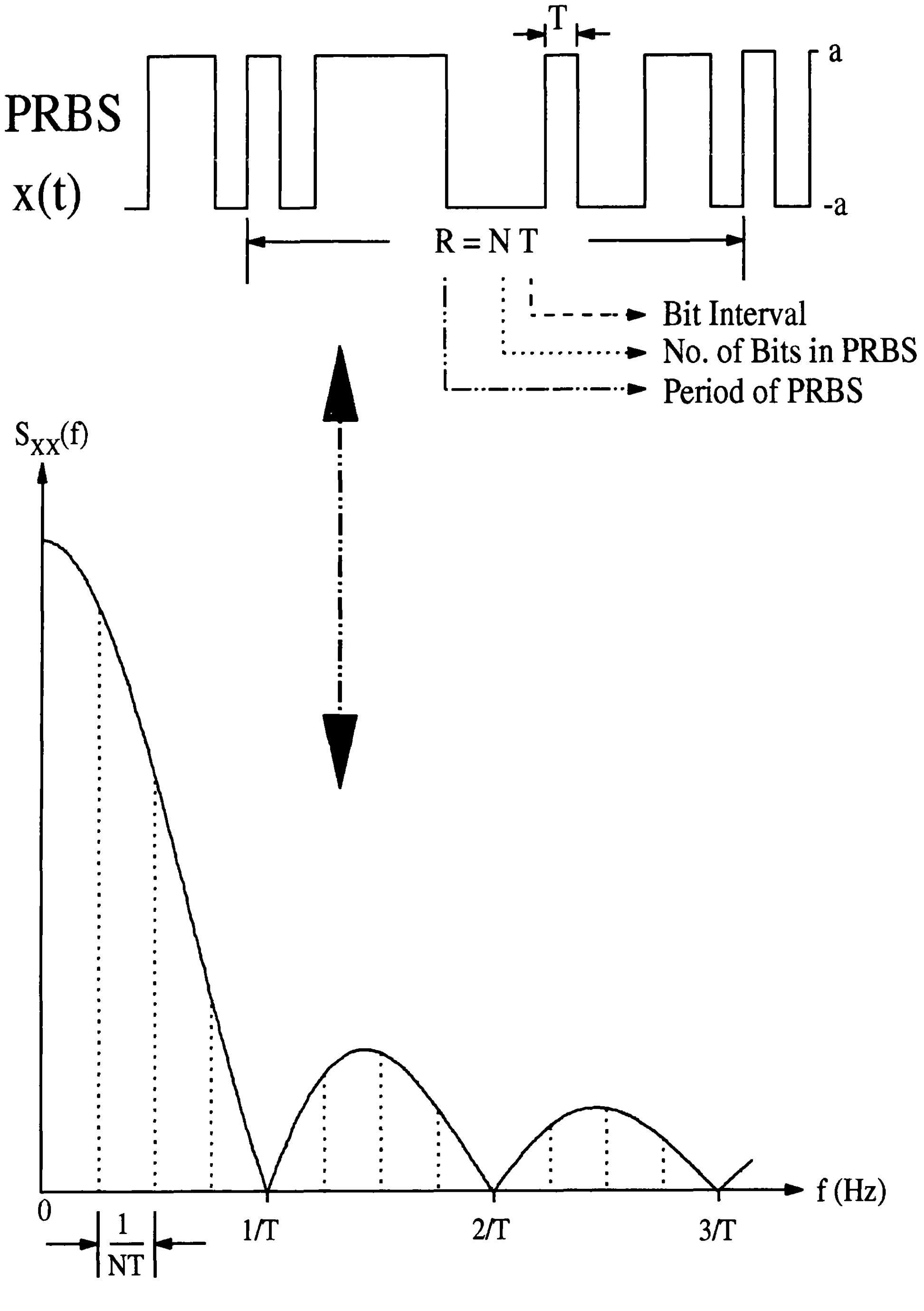


Figure 5.3: PRBS Time and Frequency Domain Relationship

5.2 Analysis Methods

Four methods are proposed to analyse the transient response data extracted at the external nodes of a CUT as a result of exciting it with a PRBS signal. The objectives are to establish which type of measurement (i.e. voltage or current) is best at detecting a particular fault, which method of analysis achieves the highest fault-coverage and which one is most efficient in terms of computation. The methods of analysis are [11-13]:

1- Samples Values
2- Rate of Change
3- Auto and Cross Correlation
4- Response Digitization

5.2.1 Samples Values

In this method a fault is detected by comparing the values of the samples of the response of the CUT with those of the fault-free toleranced response. The number of instances (i.e samples) at which the CUT response falls outside the tolerance envelope are counted, and the percentage of deviation from the ideal response is accumulated. A parameter called the Coefficient of Variation (CV) is calculated for each fault that was detected at least at one instant. The objective of calculating CV is to determine which type of measurement, voltage or current, detects a particular fault with higher degree of confidence. CV is defined by Equations 5.7 and 5.8.

$$D_i = \left| (Yf_i - Yn_i) / \overline{Yn} \right| * 100\% \quad \text{.........} (5.7)$$

$$CV = (dn / M) * \sum_{i=1}^{M} D_i \quad \text{.............} (5.8)$$

where :

i = 1,2, ... M

M = Total Number of Samples

D = Percentage of Deviation

Yn = Response of Fault-Free CUT

$\overline{\text{Y}}$n = Average of Fault-Free CUT Response

Yf = Response of CUT

dn = Number of Detection Instances

CV is then normalised to make it easier to compare the results of processing the transient responses of the voltage and current measured.

5.2.2 Rate of Change

In this method, the rate of change (R_t) of the response of a CUT between the sampling intervals (Δt) is calculated according to Equation 5.9.

$$R_t = (m_{i+1} - m_i) / \Delta t \quad \text{...............} (5.9)$$

where m_i and m_{i+1} are the values of the response at samples (i) and (i+1) respectively.

The rates of change for the CUT is then compared with that of the fault-free response. This method of analysis is capable of detecting faults which produce responses similar to that of the fault-free one but are shifted in time.

5.2.3 Auto and Cross Correlation

The auto-correlation function of the output transient voltage y(t), and the cross-correlation function between y(t) and the input PRBS test sequence x(t), of a fault-free circuit are calculated. A tolerance envelope is then wrapped around the ideal fault-free correlation functions. The fault-free toleranced correlation functions are then compared with the correlation functions of the CUT. The objective is to determine the number of instances at which the CUT correlation functions fall outside the toleranced functions and hence detect the presence of a fault.

The auto-correlation function of the CUT is the correlation between its transient response and the fault-free response, and the CUT cross-correlation function is the correlation between the its response and the PRBS input signal.

To determine whether auto or cross correlation detects a particular fault with better confidence a coefficient of variation similar to the one in Equation 5.8 is calculated and normalised. In this case the number of samples (M) in Equation 5.8 is replaced with the length of the correlation function which is (2M -1).

The attraction of using correlation functions is that these functions have well defined properties [10]. An important one of these properties, as far as testing circuits is concerned, is the symmetry of the autocorrelation function. If the response of the CUT does not match that of the fault-free circuit, the autocorrelation function will be asymmetrical and hence the presence of a fault is easily detectable.

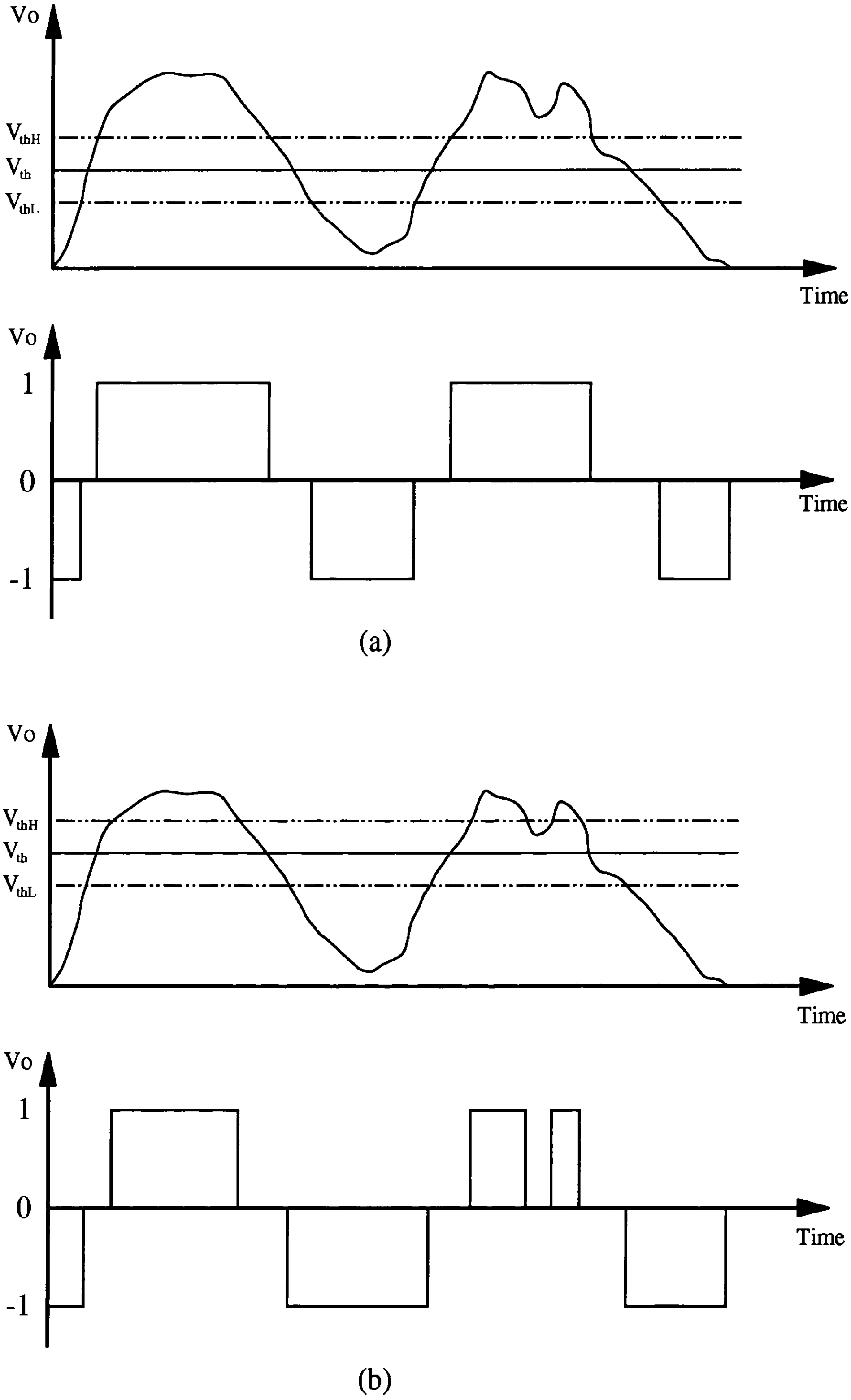
5.2.4 Response Digitization

In the digitization method of analysis the transient response waveforms of both output voltage and supply current are digitized into 3-levels : 1, -1 and 0. This is achieved by assuming a threshold value, V_{th} and I_{th} for voltage and current respectively, with a small bound round it. If a sample data value falls above the upper-bound (V_{thH} or I_{thH}) of the threshold value it is considered a logic high (1), if it falls below the lower-bound (V_{thL} or I_{thL}) it is considered a logic low (-1), and otherwise the logic is considered unresolved and denoted by 0. The digitized fault-free and CUT responses are then compared to determine the number of instances at which a fault is detected.

Varying the threshold value may lead to the detection of faults that otherwise will not be detected, and an increase in the number of detection instances for some faults. Figure 5.4 illustrates the digitization process and the effect of varying the threshold value on the digitized waveform generated. The program that implements the digitization method varies the threshold value by requesting the number of times the analysis is to be repeated. It uses this to divide the space between the maximum and the minimum values of the fault-free response to a uniform set of threshold values, each one of these values is considered a test. The program keeps track of all these tests, then compares them to determine the overall fault-coverage and which test is best at detecting a particular fault and the highest number of detection instances achieved.

5.3 Simulation and Analysis Results

Three circuit examples, a first-order low-pass filter, a fourth-order low-pass filter and a mixed-signal circuit, were simulated to demonstrate the time-domain testing technique and the analysis methods described above. All the results are based on simulating a CUT under fault-free and faulty conditions using HSPICE [14], the analogue circuit simulator. The data is then analysed by the above four analysis methods. The methods of data analysis are all implemented using the mathematical software package MATLAB [15]. To be able to process the data using MATLAB the data have first to be extracted from the HSPICE listing file and put in a format acceptable to MATLAB. This task is performed by the Pascal program listed in appendix 3.



(a)

(b)

Figure 5.4: Waveform Digitization and the Effect of Varying Threshold Value

In all the circuits tested, if the response of the CUT falls outside the bounds of the toleranced response the circuit is considered faulty, hence a fault is detectable. The tolerance region around a circuit ideal fault-free response is calculated by varying both HSPICE process parameters (e.g. VTO, TOX, L & W, etc.) and some of the circuit component values and repeating the simulation. The maximum deviation (i.e. worst case) is then used to wrap an envelope around the ideal fault-free response. The size of this tolerance region depends on the particular IC process used. The tolerance values in this thesis have been developed using typical values to illustrate the technique and do not represent any particular process.

The following subsections discuss the three circuit examples tested and the results of analysing the transient response voltage and current measurements.

5.3.1 First-Order Low-Pass Filter

The first-order active low-pass filter (LPF) circuit, with a 3-dB bandwidth of 2-KHz, and the schematic of the op-amp used are illustrated in Figure 5.5 and Figure 5.6 respectively. The low-pass filter circuit was tested by applying a PRBS signal 31-bits long and having a period of 31T. The bit interval T is 250 µsec. The transient response of the circuit was sampled at five times the bit interval (20-KHz), resulting in 155 samples for each signal measured. Figure 5.7 shows the input PRBS test sequence and the toleranced responses of the output voltage (Vout) and supply current (Idd).

A total of 65 single catastrophic fault conditions in the op-amp transistors were simulated. The simulation for 4 of these faults did not converge, therefore nothing will be assumed about the detectability of these faults and they will be excluded from any fault-coverage or other subsequent calculations.

The samples values analysis method was applied to the data for Vout and Idd. The results indicate that the percentage of fault-coverage of Vout and Idd is the same and equals 92% (i.e. 56 faults out of the 61 that converged were detected).

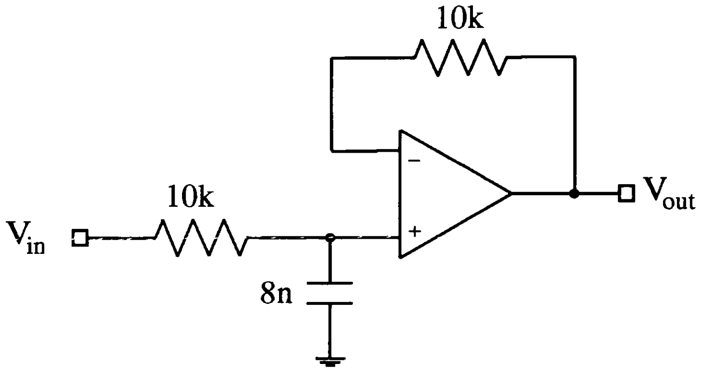


Figure 5.5: First-Order Low-Pass Filter Circuit

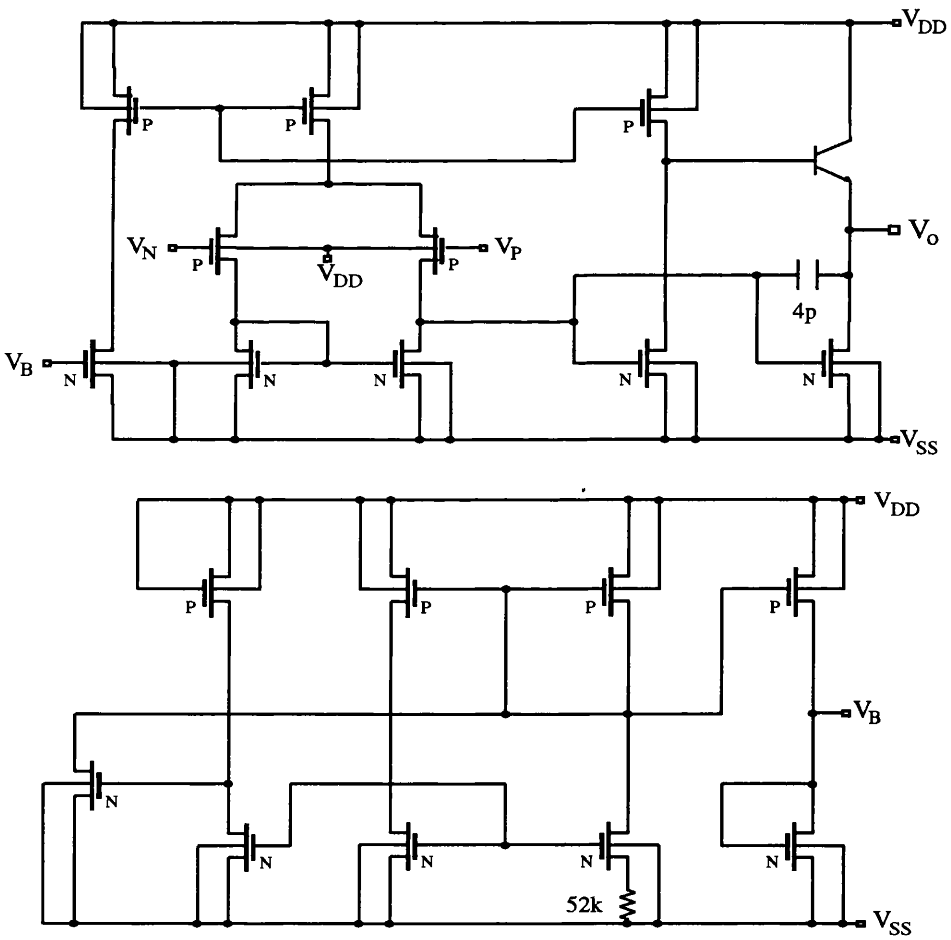


Figure 5.6: Schematic of the Op-Amp Used [16]

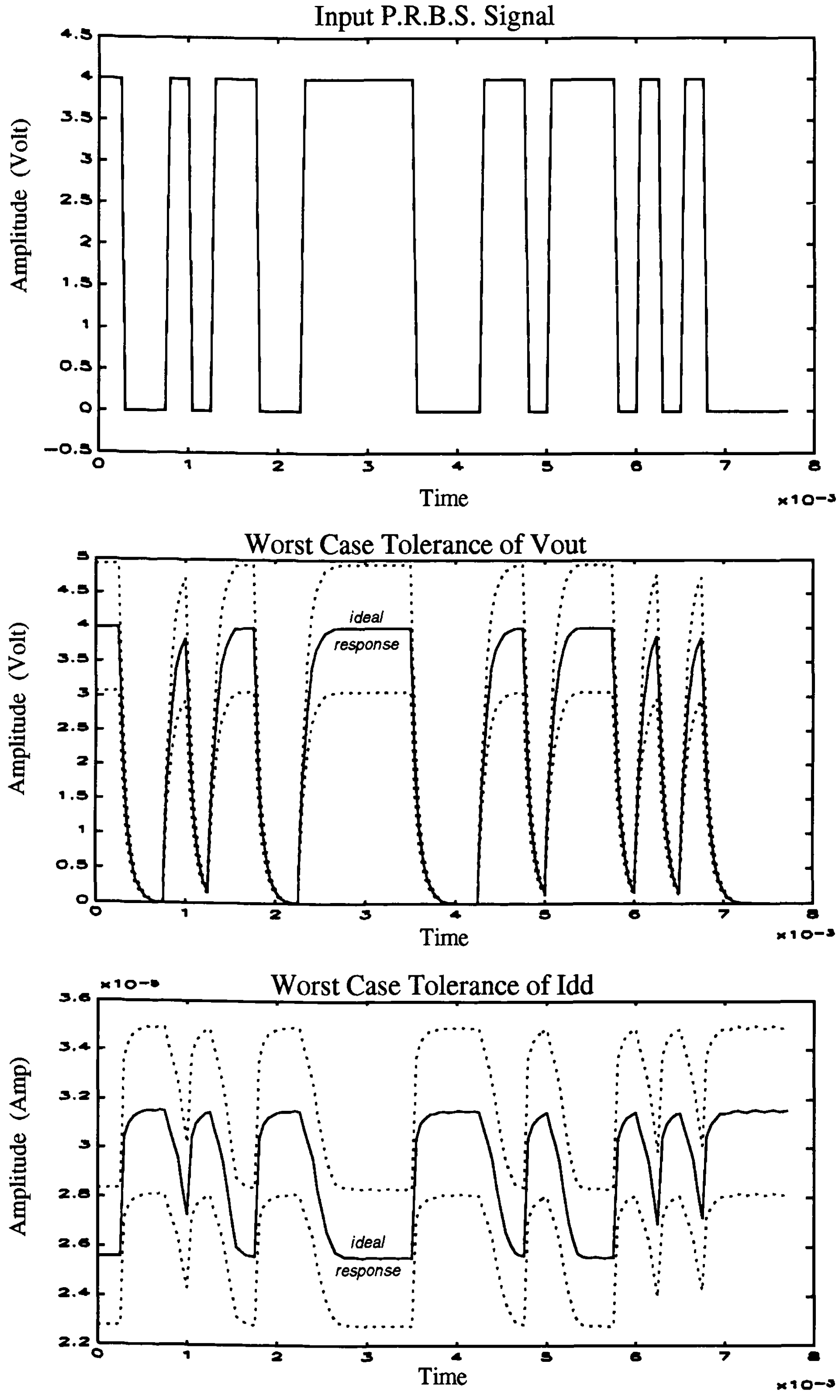


Figure 5.7: Input PRBS and Toleranced Responses Vout & Idd of First-Order LPF

To determine whether Vout or Idd detected faults with better confidence the bar charts in Figure 5.8 and Figure 5.9 were plotted. The charts in Figure 5.8 show that the number of instances at which a fault was detected by a particular type of measurement (i.e. Vout or Idd), it does not take into account the value of the difference at the instant of detection. Based on Figure 5.8, a fault is on average more detectable by Idd than Vout.

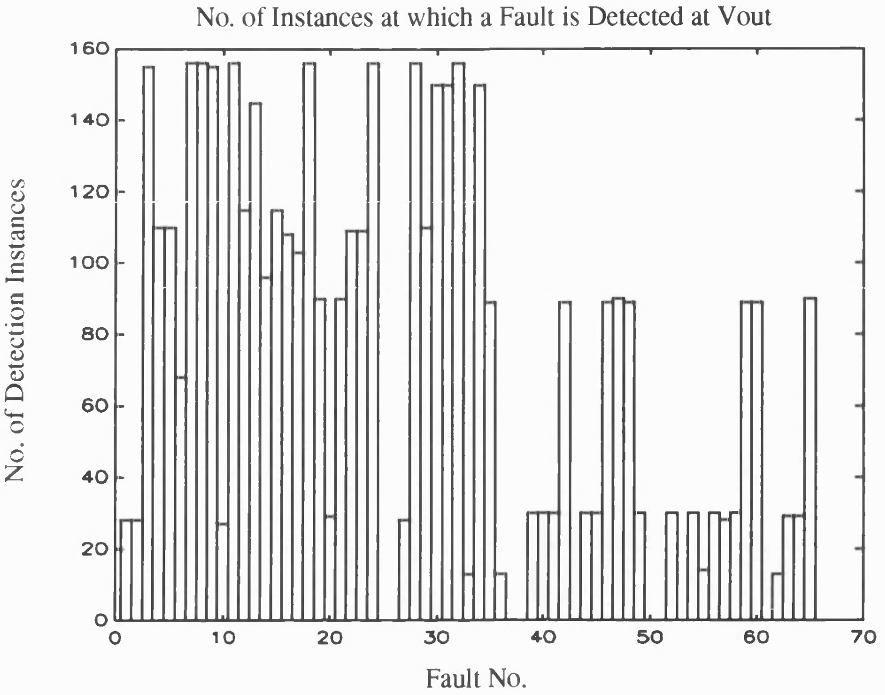
The normalised coefficients of variation (CVs) bar charts in Figure 5.9 are plotted according to Equation 5.8. The charts take into account both the number of detection instances and the percentage of deviation from the ideal fault-free response at each detection instant. This gives a measure of how similar or dissimilar the signatures of the fault-free circuit and the CUT. Figure 5.9 shows that Vout and Idd are complementary to each other, because some faults are detected with higher confidence by Vout than Idd and vice versa for some other faults. In total out of the 56 faults detected 32 are detected with higher confidence by Vout (57%), while the other 24 are detected with higher confidence by Idd (43%).

The rate of change method was then applied to Vout and Idd. The results of the program that performs the method of analysis are show in Figure 5.10, it indicates that the fault-coverage at Vout did not change (i.e. 92%), but the fault-coverage achieved by Idd increased from 92% to 98%. However, further analysis revealed that the confidence in the detection of the 4 extra faults is low because they were only detected in 3 intervals.

The ideal fault-free and toleranced auto-correlation function of Vout and cross-correlation function between Vout and the input PRBS test signal are illustrated in Figure 5.11. The auto and cross correlation functions of the faults simulated are compared with the fault-free correlation functions in Figure 5.11. As in the samples values method above, a coefficient of variation is computed for every fault in order to compare the detectability of a fault by auto and cross correlation. The results of computation show that the fault-coverage of auto and cross correlation are the same and equal 92%.

The plots of the number of detection instances, illustrated in Figure 5.12, show that the number of detection instances by auto-correlation is marginally higher than that by cross-correlation. However, the correlation normalised coefficients of variation in Figure 5.13 indicate that the detectability of faults by auto and cross correlation is virtually the same.

The application of the digitization analysis method to Vout and Idd data with 10 tests specified, resulted in a fault-coverage of 88% and 92% respectively. Figure 5.14 illustrates two plots resulting from the processing of digitized Vout data; the first one shows which test is best suited to detect a particular fault while the second one indicates the highest number of detection instances achieved for each fault. Figure 5.15 is similar to Figure 5.14 except that it resulted from processing digitized Idd data.



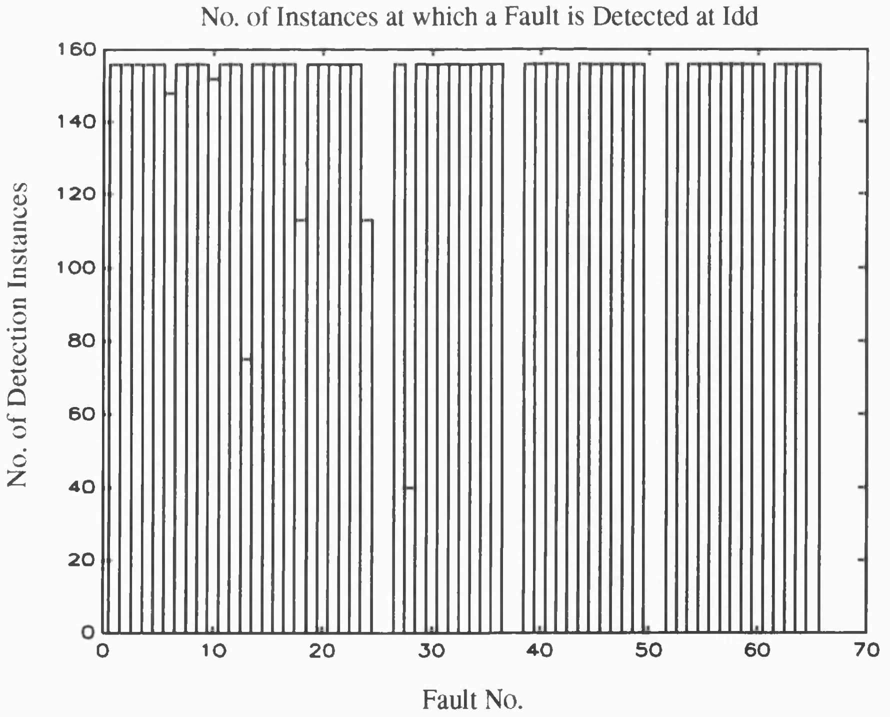
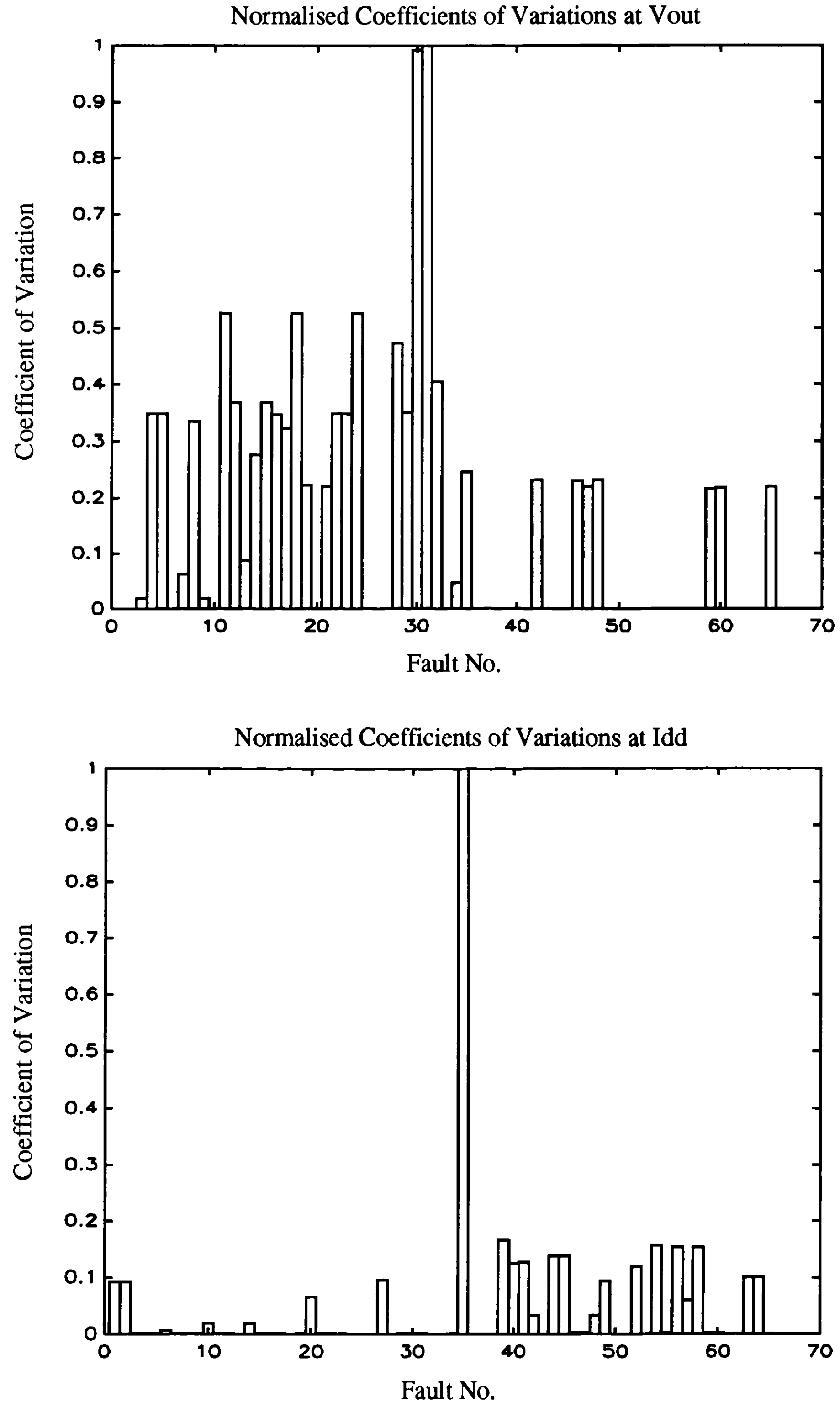


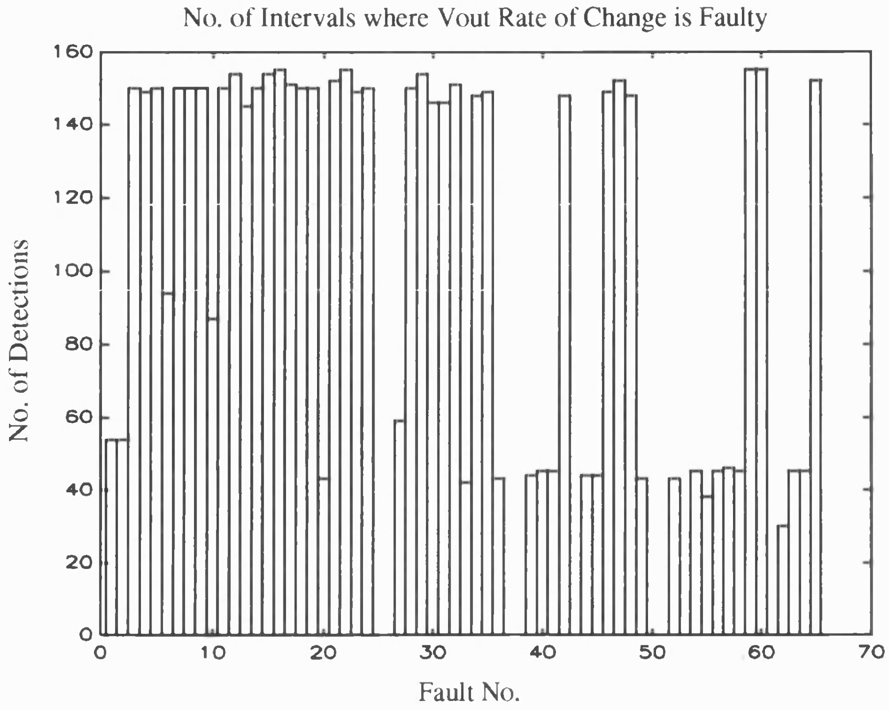
Figure 5.8: Detection Instances by Vout & Idd of First-Order LPF



Normalised Coefficients of Variations at Vout

Normalised Coefficients of Variations at Idd

Figure 5.9: Norm. Coefficients of Variation of Vout & Idd of First-Order LPF



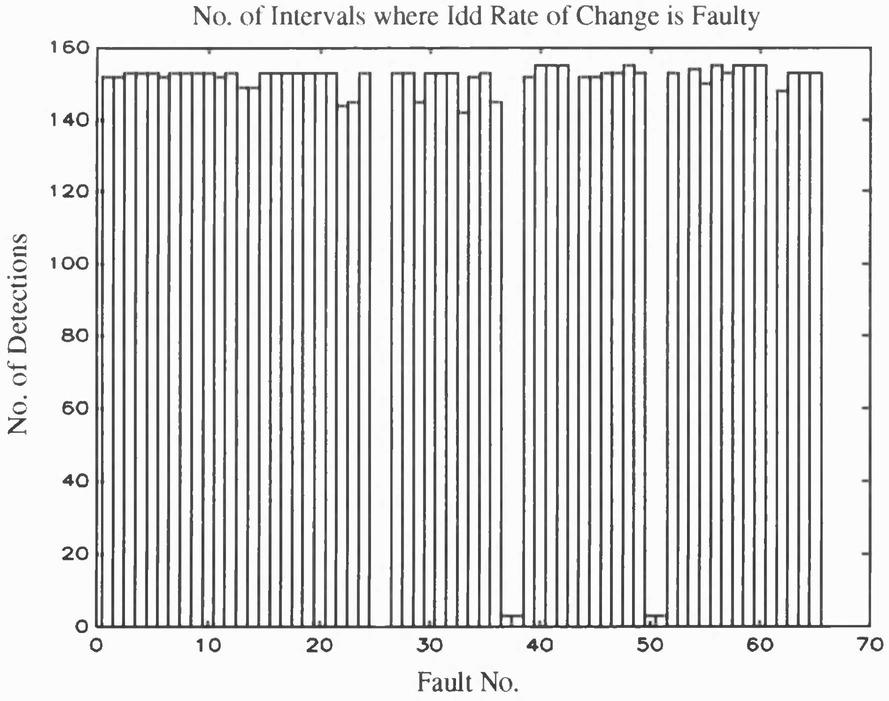
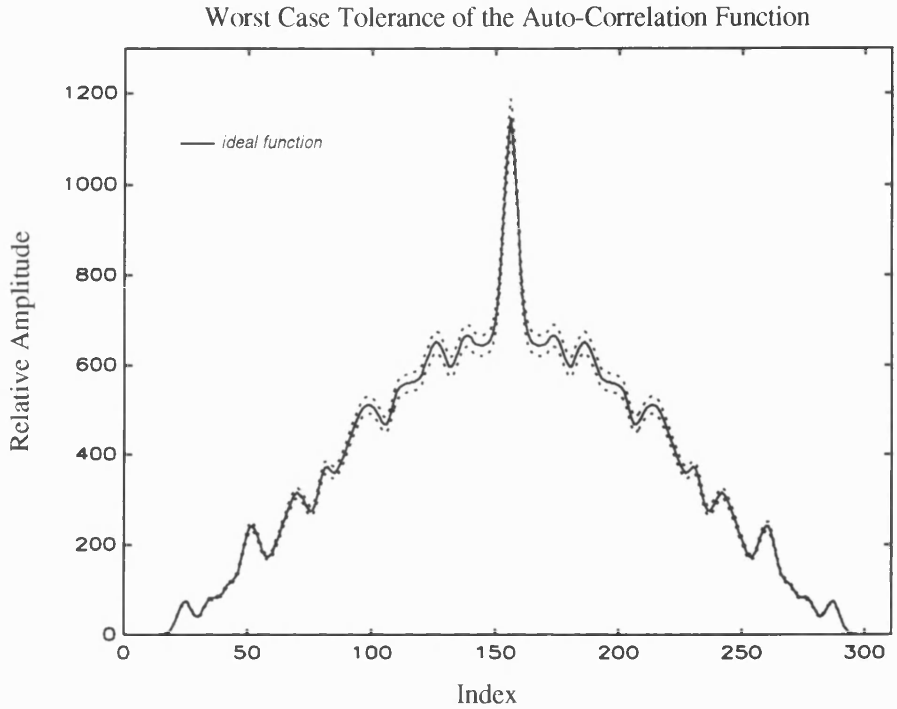


Figure 5.10: Detection Intervals by Vout & Idd Rate of Change of First-Order LPF



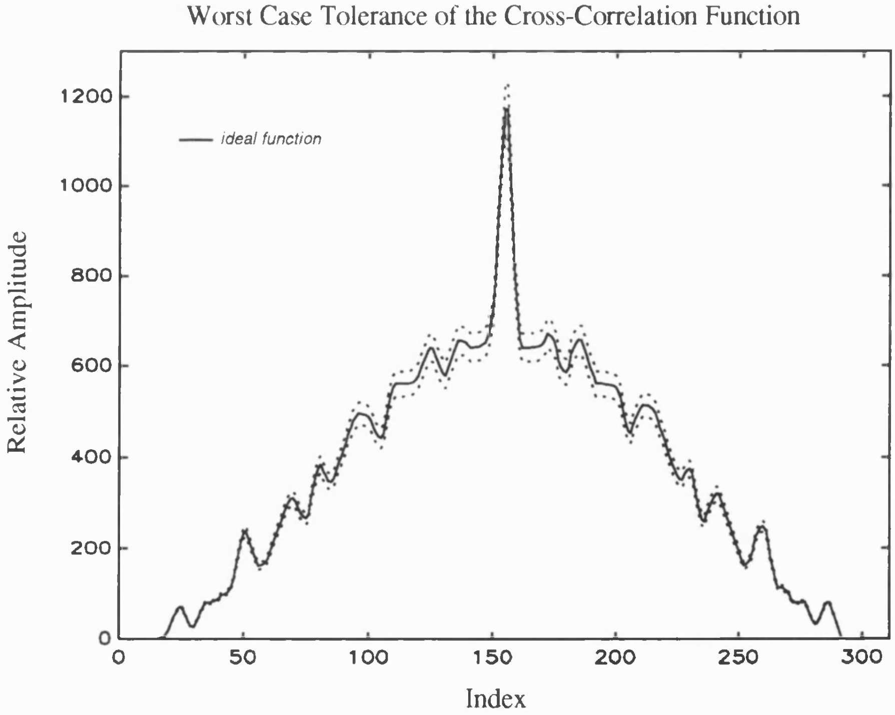


Figure 5.11: Toleranced Auto & Cross Correlat. Funct. at Vout of First-Order LPF

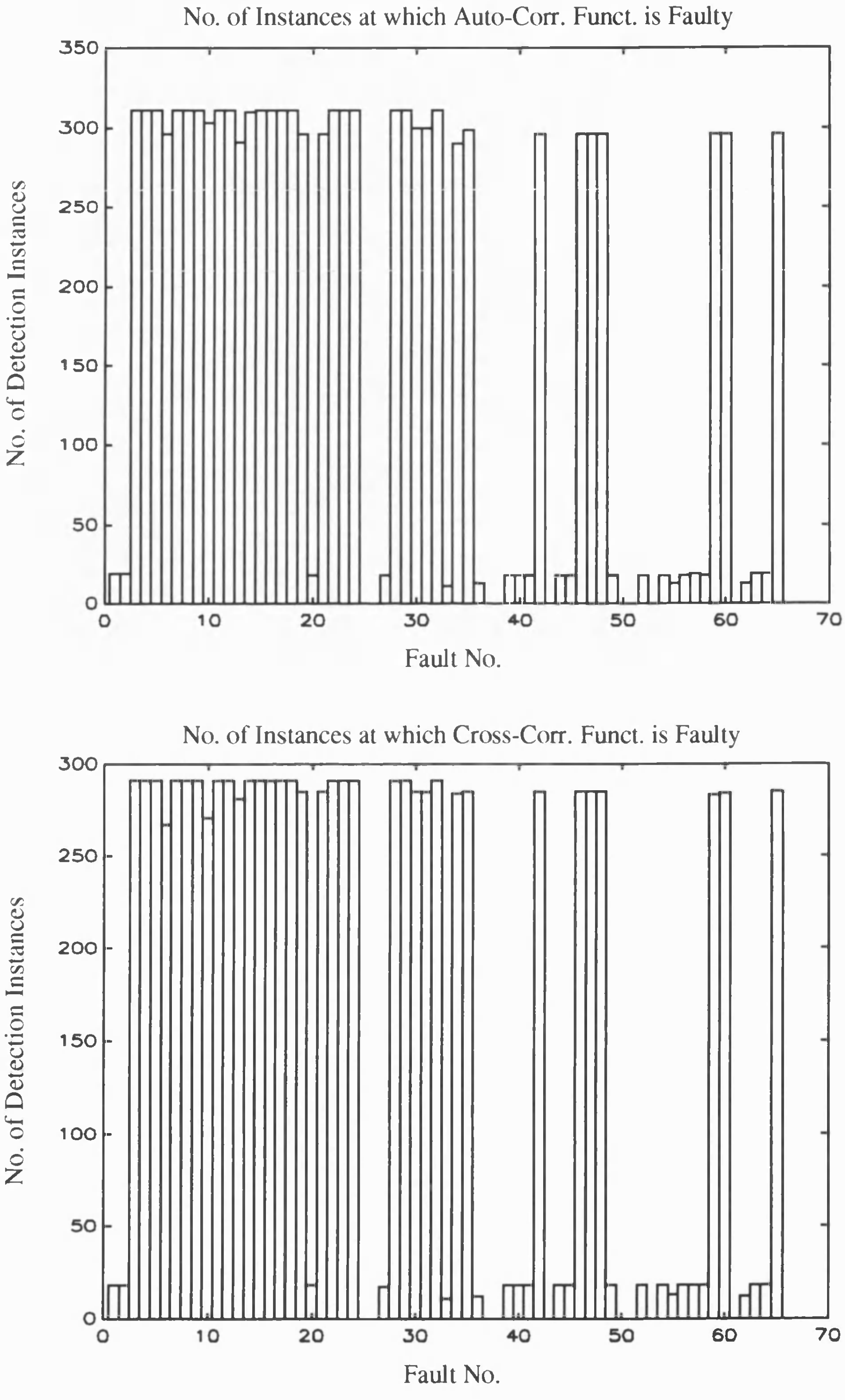
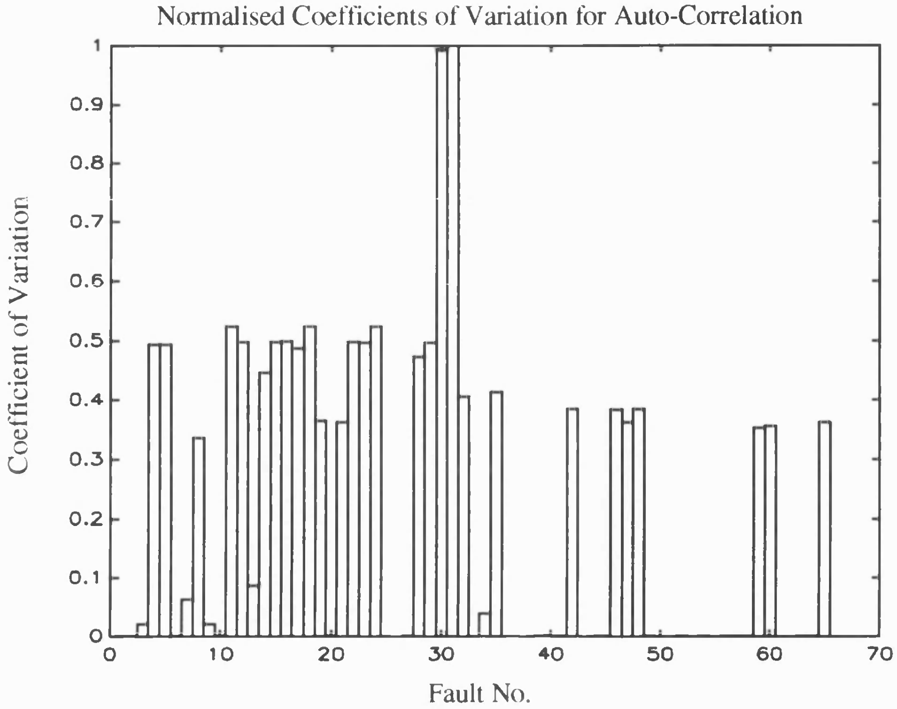


Figure 5.12: Detection Instances by Auto & Cross Correlations of First-Order LPF



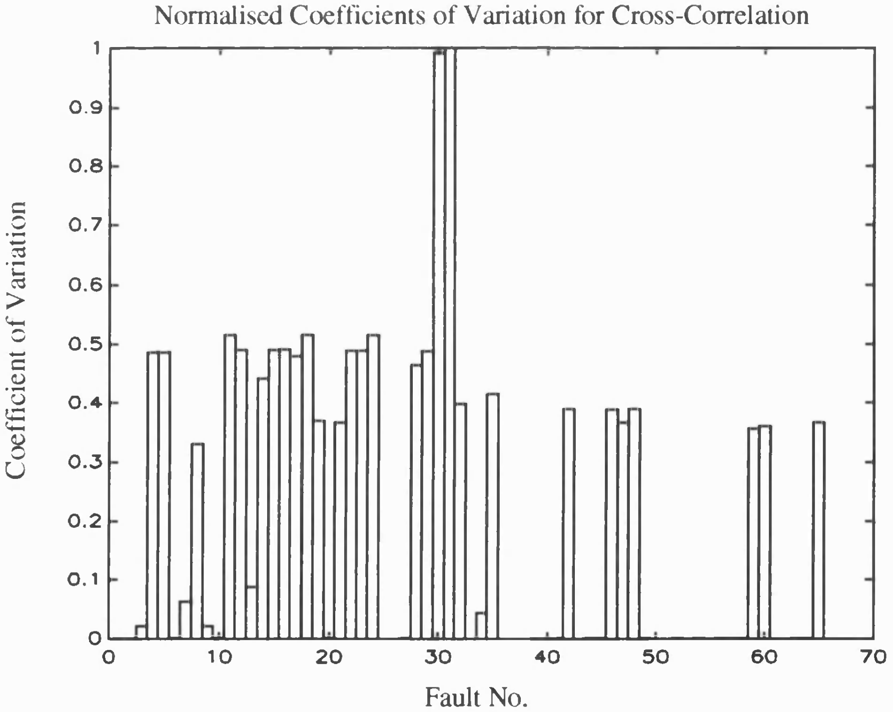
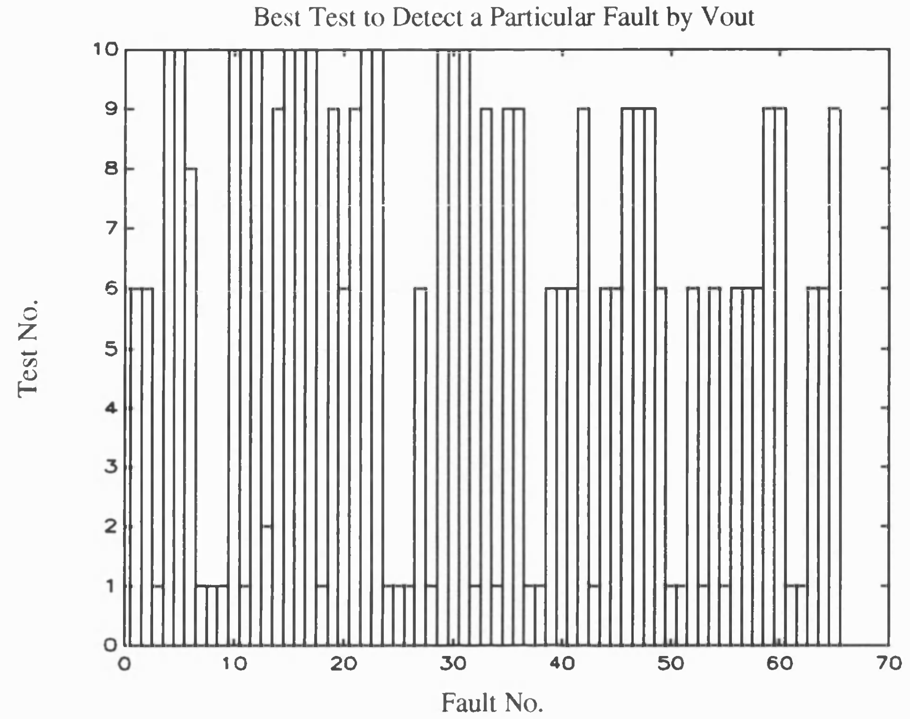


Figure 5.13: Normalised CVs of Auto & Cross Correlations of First-Order LPF



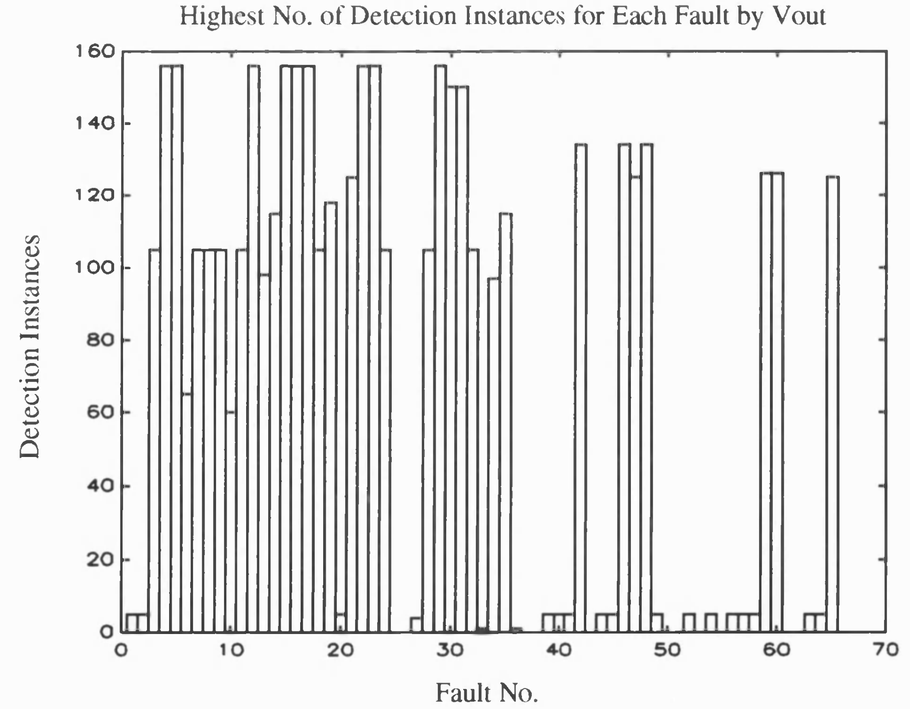
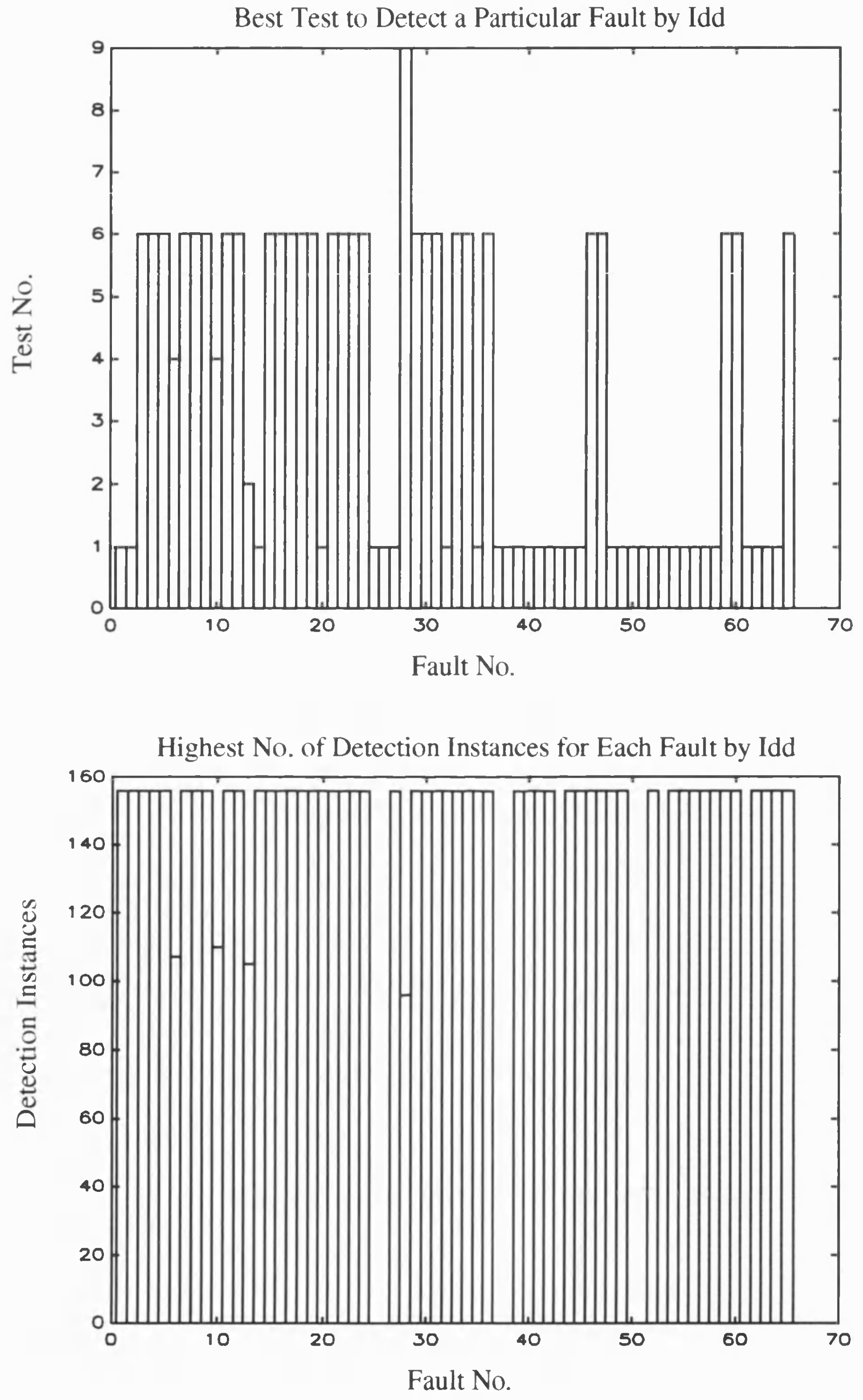


Figure 5.14: Best Test & Highest Detections by Digitized Vout of First-Order LPF



Best Test to Detect a Particular Fault by Idd

Highest No. of Detection Instances for Each Fault by Idd

Figure 5.15: Best Test & Highest Detections by Digitized Idd of First-Order LPF

5.3.2 Fourth-Order Low-Pass Filter

The second circuit example simulated is the fourth-order low-pass filter, with a 3-dB bandwidth of 10-KHz, depicted in Figure 5.16. The objective is to investigate the ability of the time-domain technique to test analogue circuits larger than the one in the previous example. The op-amps used in Figure 5.16 are the same as the one in Figure 5.6.

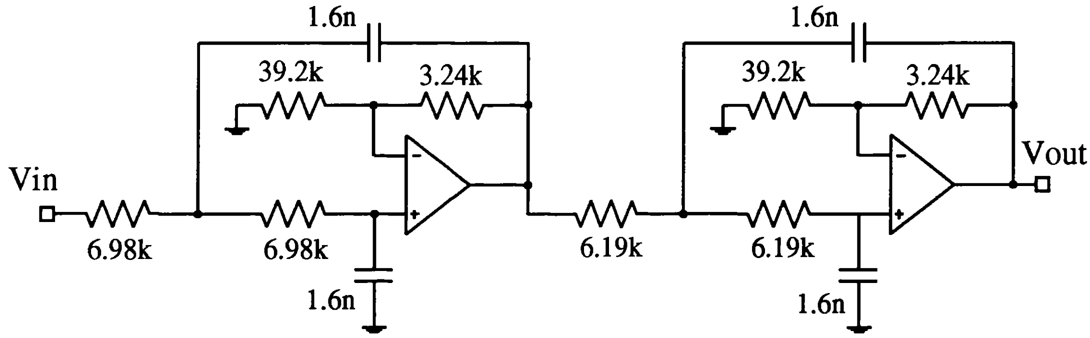
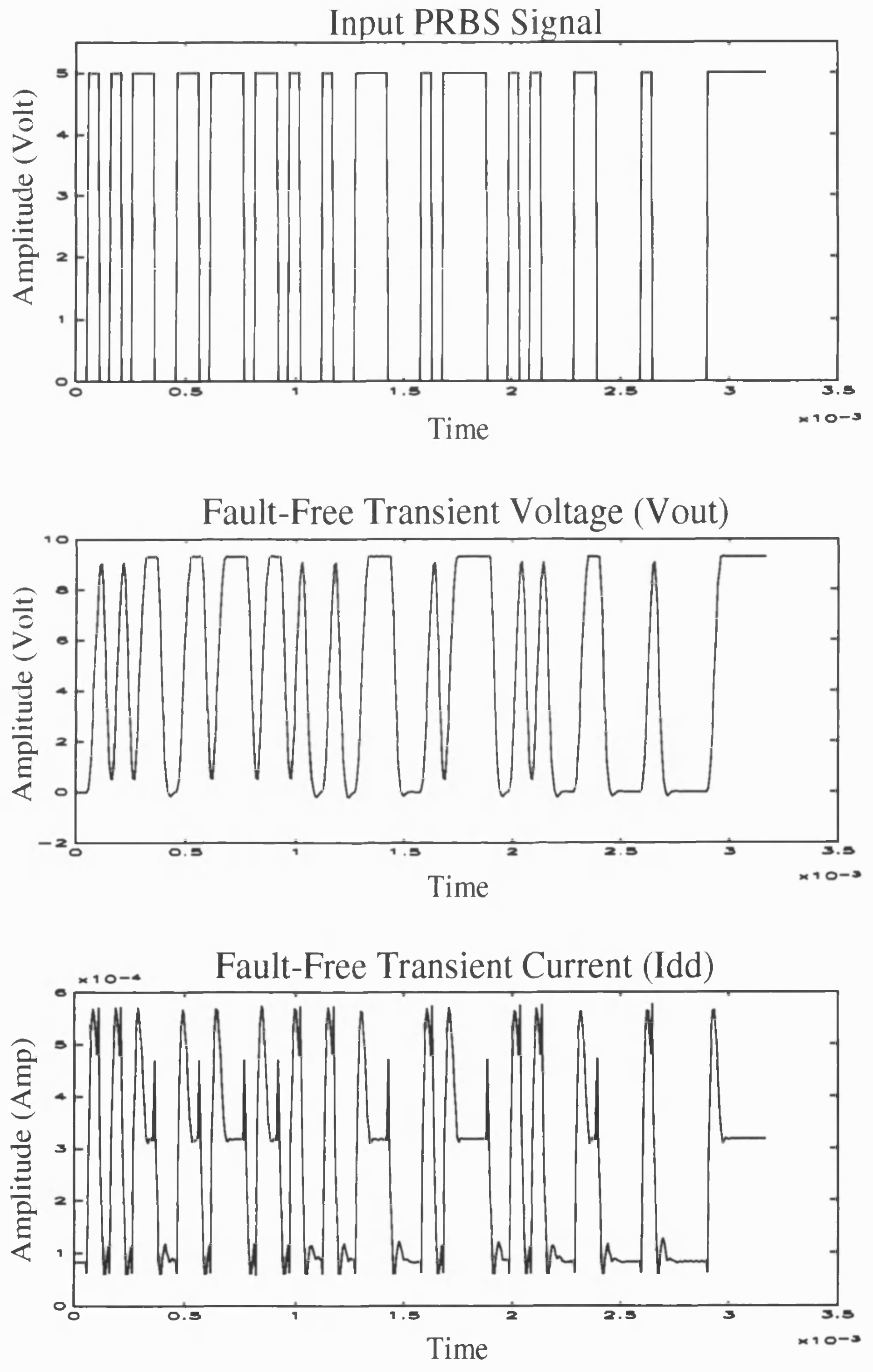


Figure 5.16: Fourth-Order Low-Pass Filter

The circuit in Figure 5.16 was tested by applying a 63-bits long PRBS test signal, having a period of 63T. The bit interval T is 50.8 μsec. The transient response of the circuit was sampled at 8 times the bit interval, resulting in 504 samples per period for each signal measured. The input PRBS test sequence and the fault-free transient response waveforms of Vout and Idd are illustrated in Figure 5.17.

Seventy single fault conditions were introduced to the network. Of these faults 60 were catastrophic faults in the MOS op-amps, and 10 soft faults in the resistive and capacitive components of the network. The soft faults ranged in variations between ± 25% and ± 50% of a component fault-free value.



Input PRBS Signal

Fault-Free Transient Voltage (Vout)

Fault-Free Transient Current (Idd)

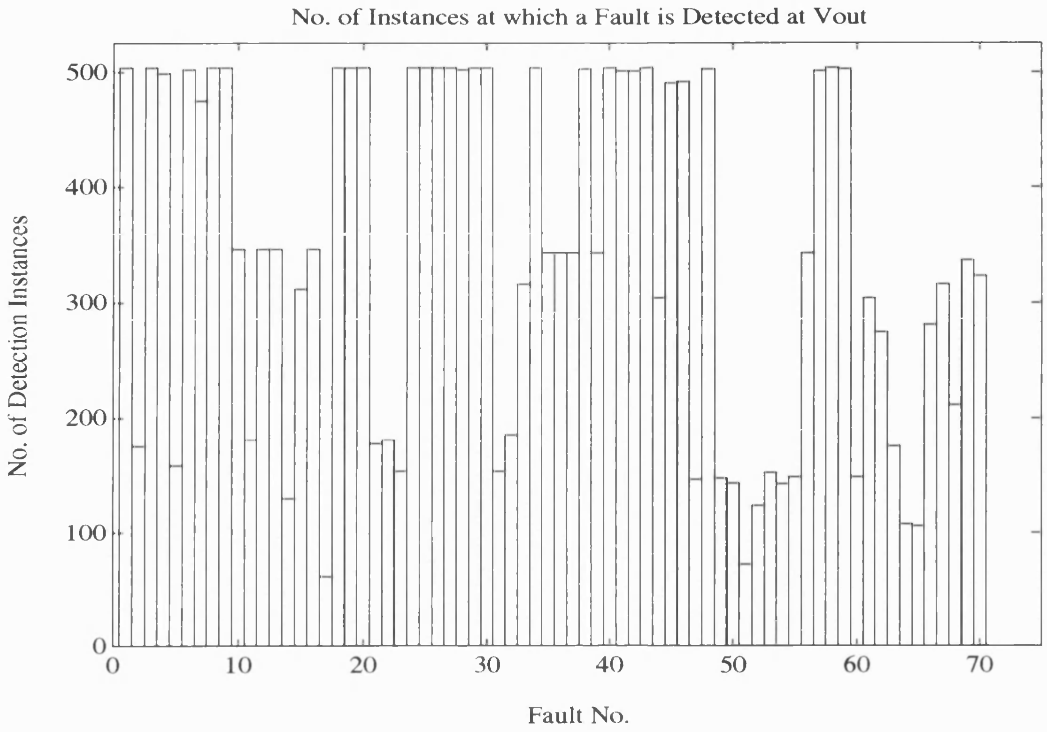
Figure 5.17: Input PRBS and Fault-Free Vout & Idd of Fourth-Order LPF

The results of the samples values method, illustrated in Figure 5.18, indicate that both Vout and Idd achieve an equal fault-coverage of 100% (i.e. all faults introduced were detected). When the normalised coefficients of variation for Vout and Idd were computed and plotted as shown in Figure 5.19, it turned out that 50 faults (i.e 71.43%) are best detected by Vout while the other 20 faults (i.e. 28.57%) are best detected by Idd.

The rate of change for both Vout and Idd, as in the first example, was calculated. The bar charts in Figure 5.20 show that the fault-coverage of both type of measurement is equal to 100%.

Calculations of the auto and cross correlation functions indicate that in terms of fault-coverage both functions are equivalent, each detects 65 faults (i.e. 92.86) as shown in Figure 5.21. To compare the detectability of faults, as in the previous example, the coefficients of variation (CVs) for both auto and cross correlation were calculated and plotted in Figure 5.22. The results of processing the CVs revealed that 54 faults are best detected by auto-correlation, and the remaining 11 faults are best detected by cross-correlation. Further analysis of the results indicated that the difference in detectability is small, therefore both auto and cross correlation are almost equivalent in terms of faults detectability.

The digitization method of analysis was then applied to Vout and Idd with 10 tests specified. The results of calculations indicate that both Vout and Idd achieved a fault-coverage of 100% each. Plots of the best test to detect a particular fault and the highest number of detection instances for each fault are illustrated in Figure 5.23 and Figure 5.24 for Vout and Idd respectively.



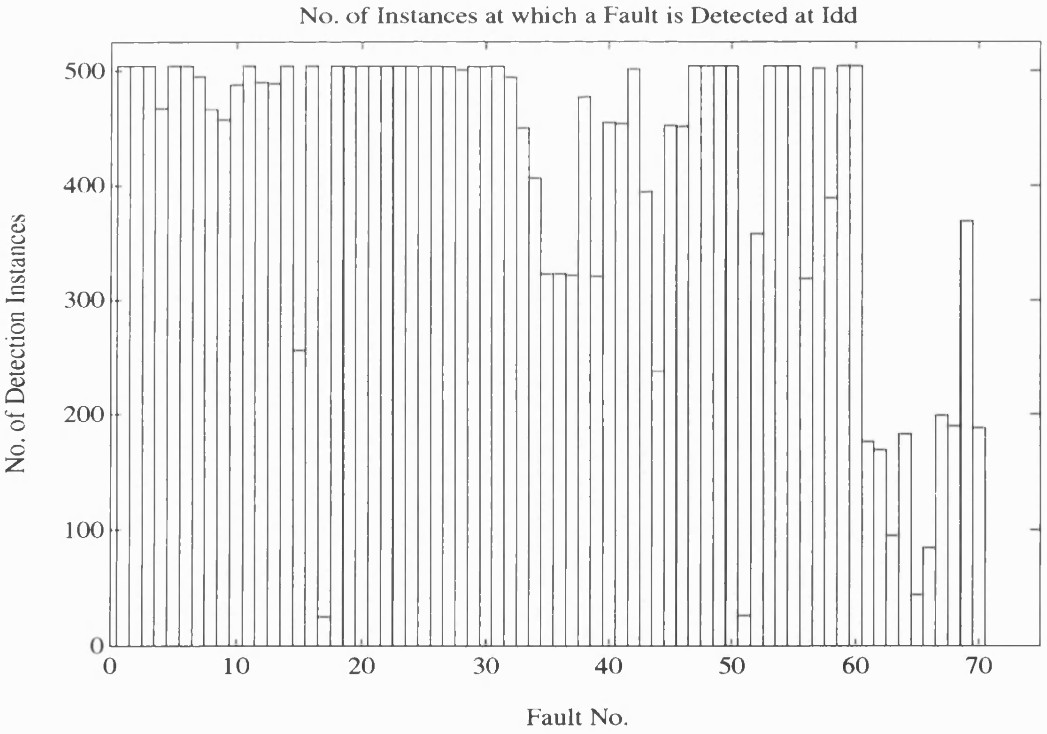
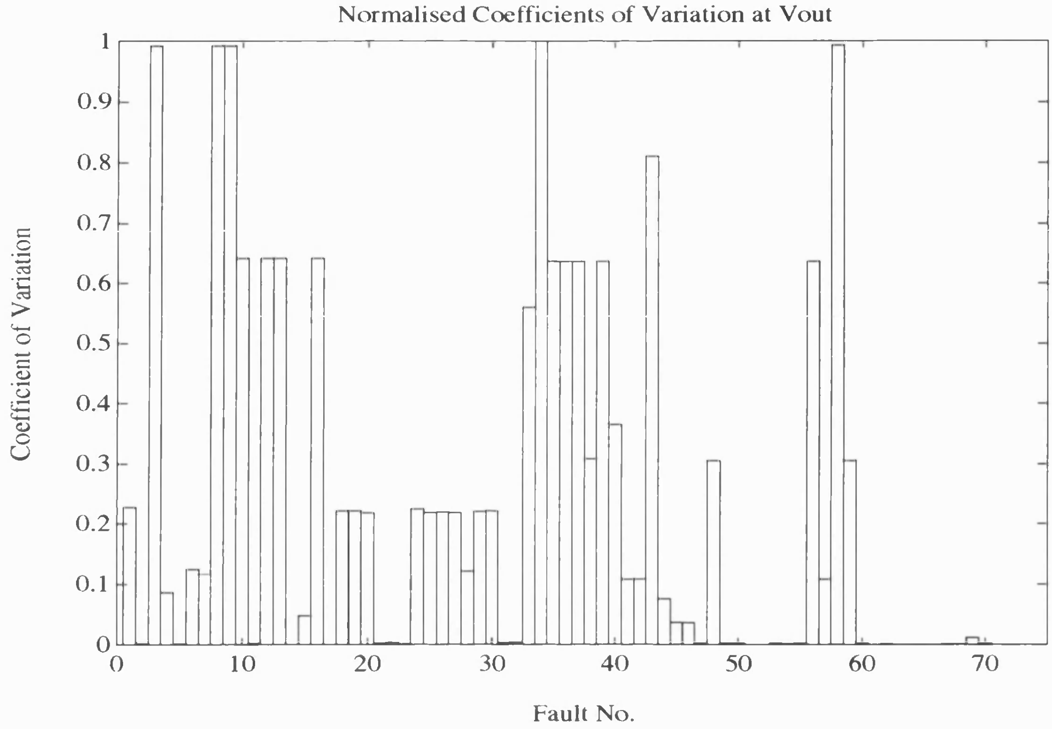


Figure 5.18: Detection Instances by Vout & Idd of Fourth-Order LPF



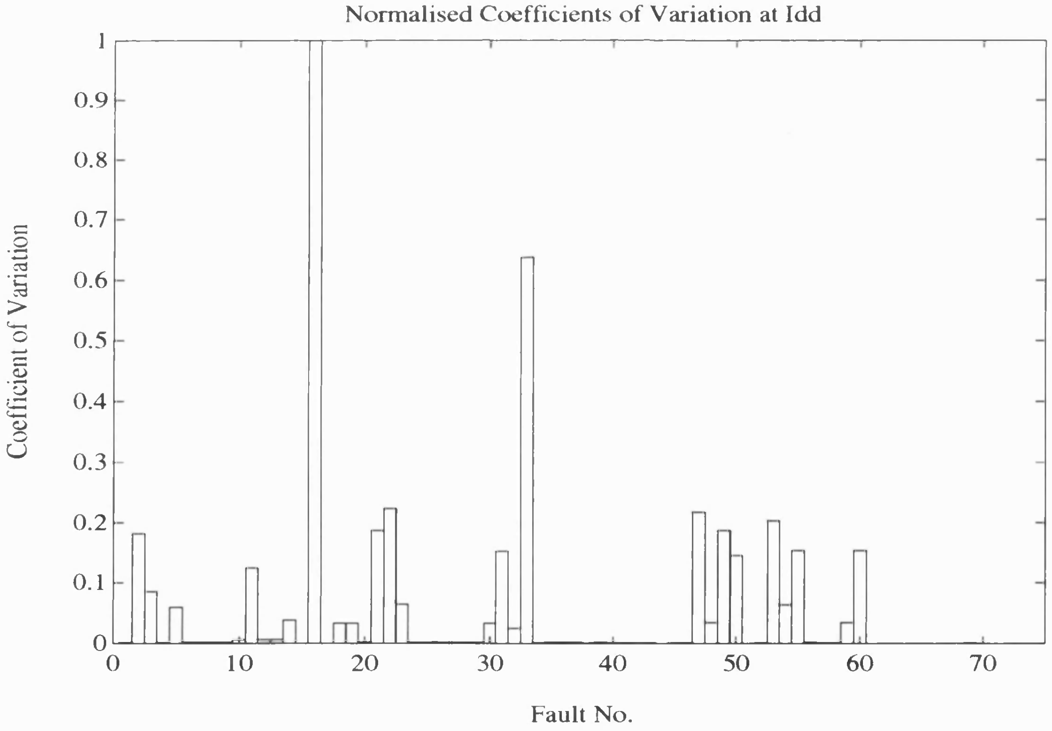
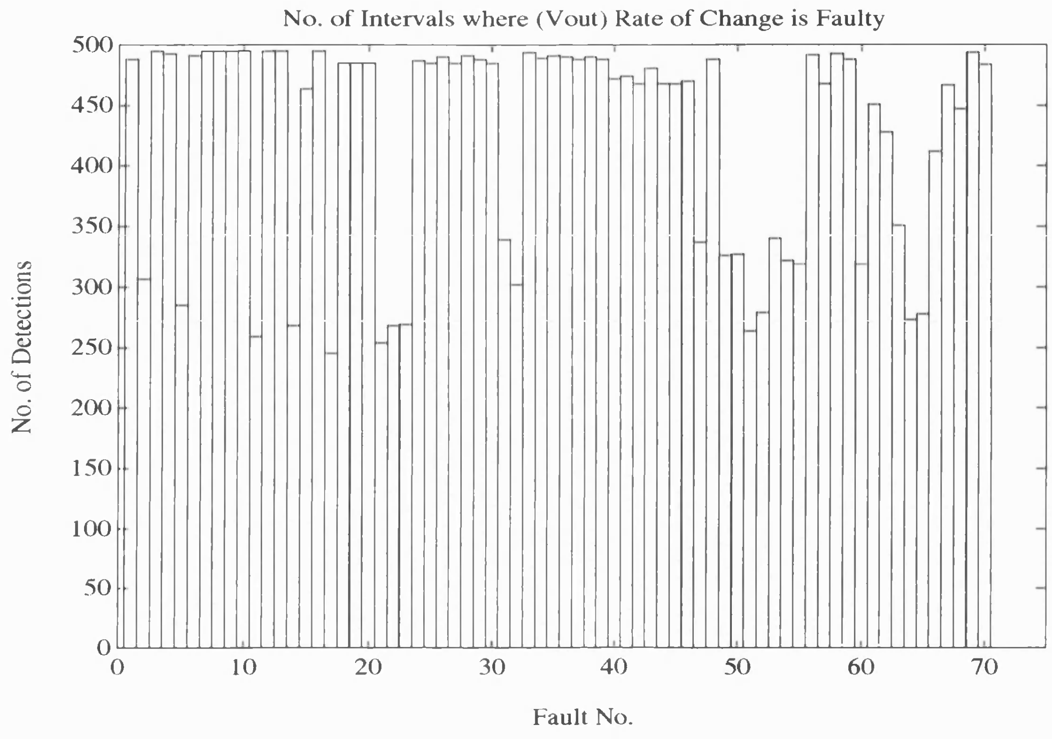


Figure 5.19: Norm. Coefficients of Variation of Vout & Idd of Fourth-Order LPF



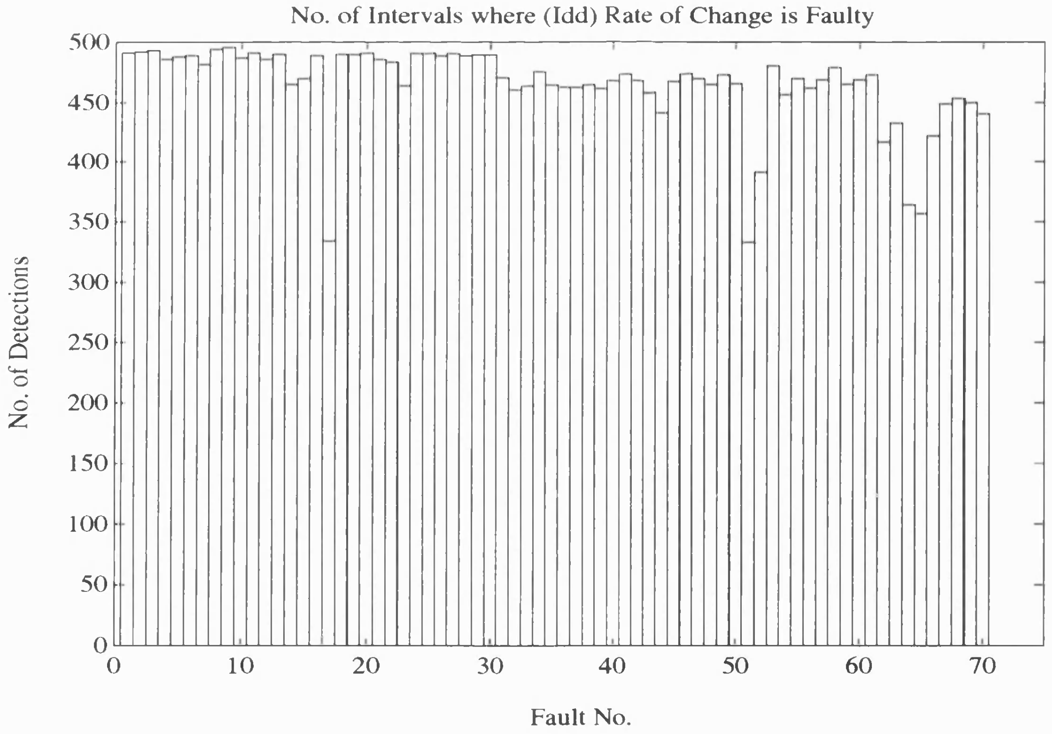
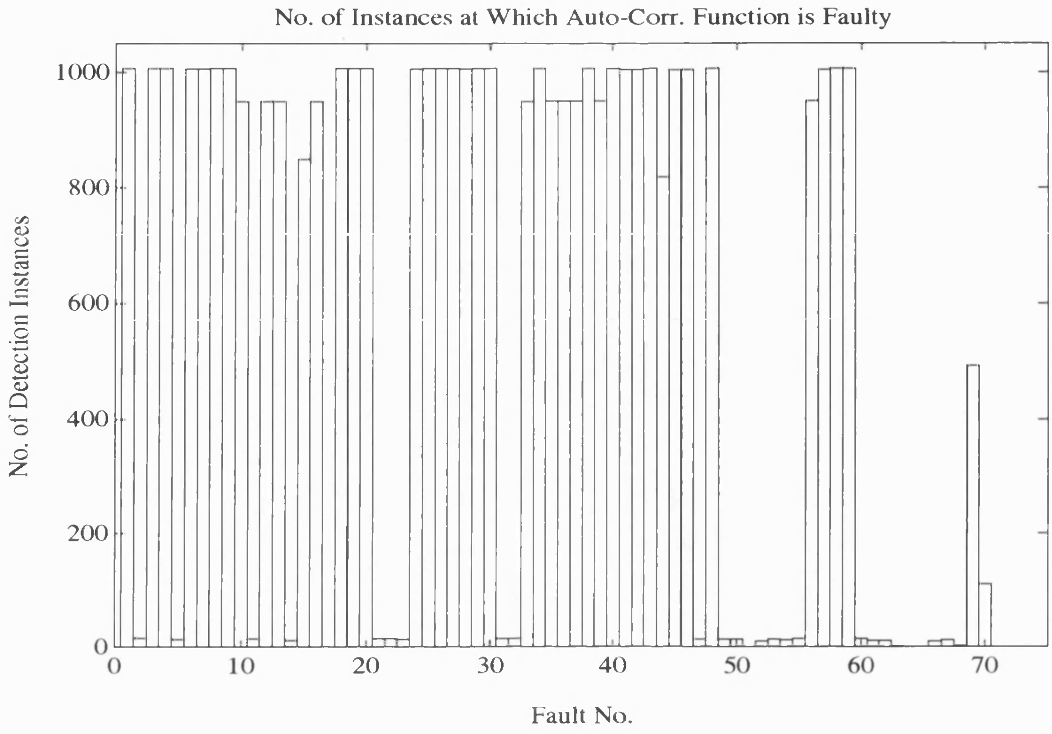


Figure 5.20: Detect. Intervals by Vout & Idd Rate of Change of Fourth-Order LPF



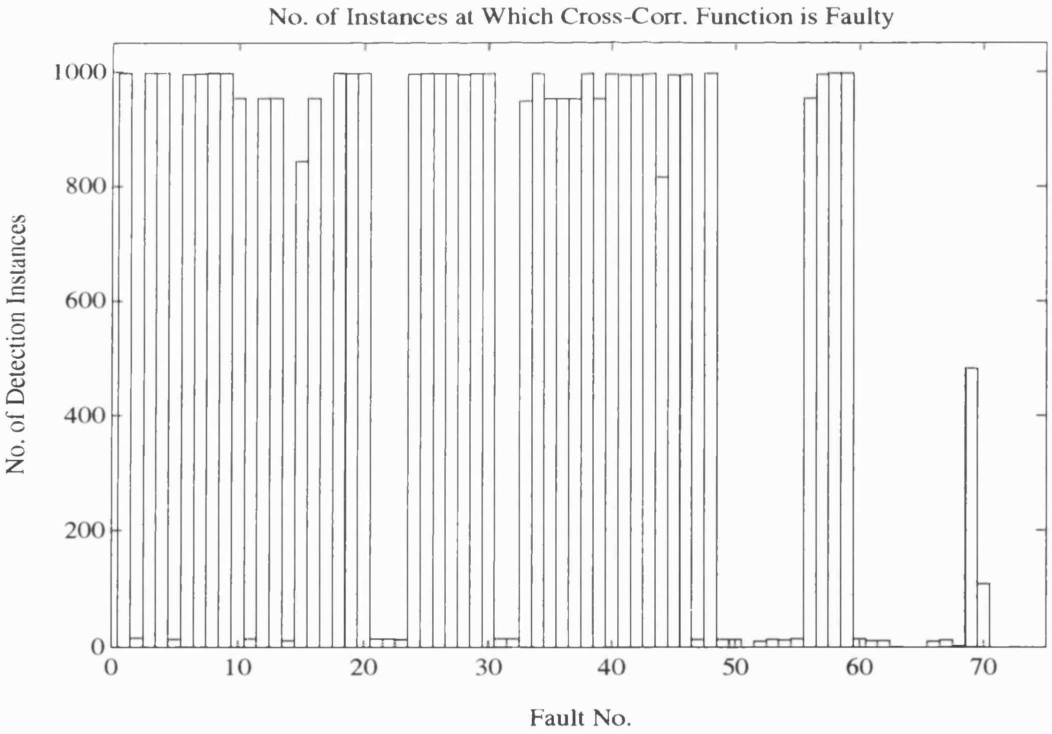
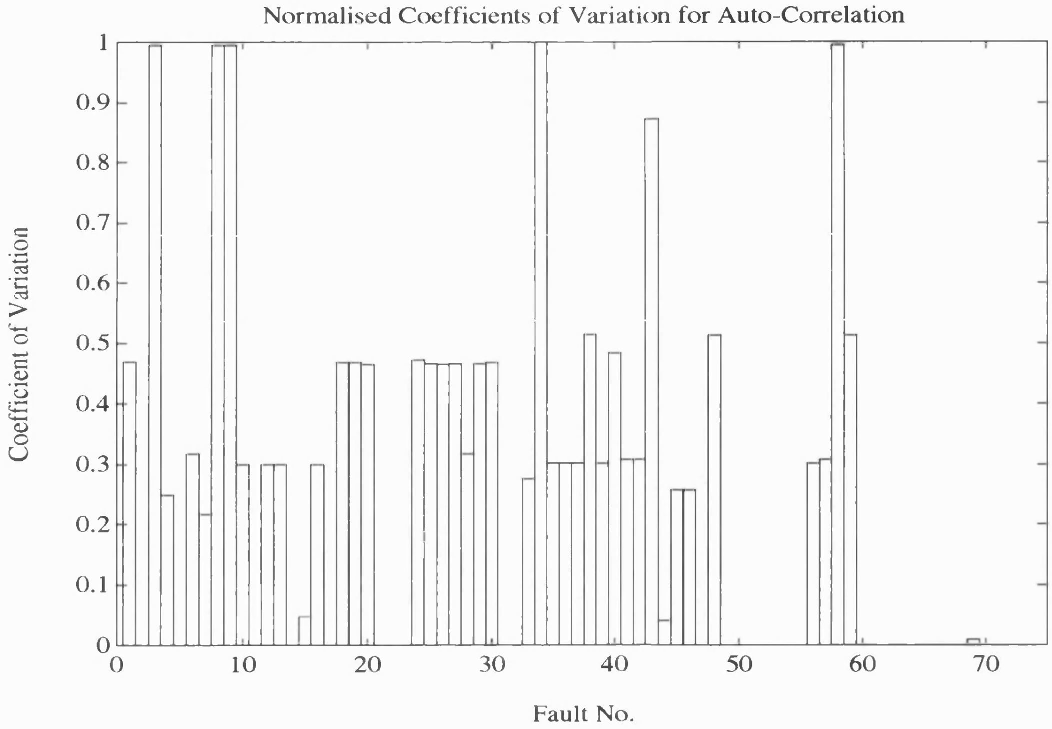


Figure 5.21: Detect. Instances by Auto & Cross Correlations of Fourth-Order LPF



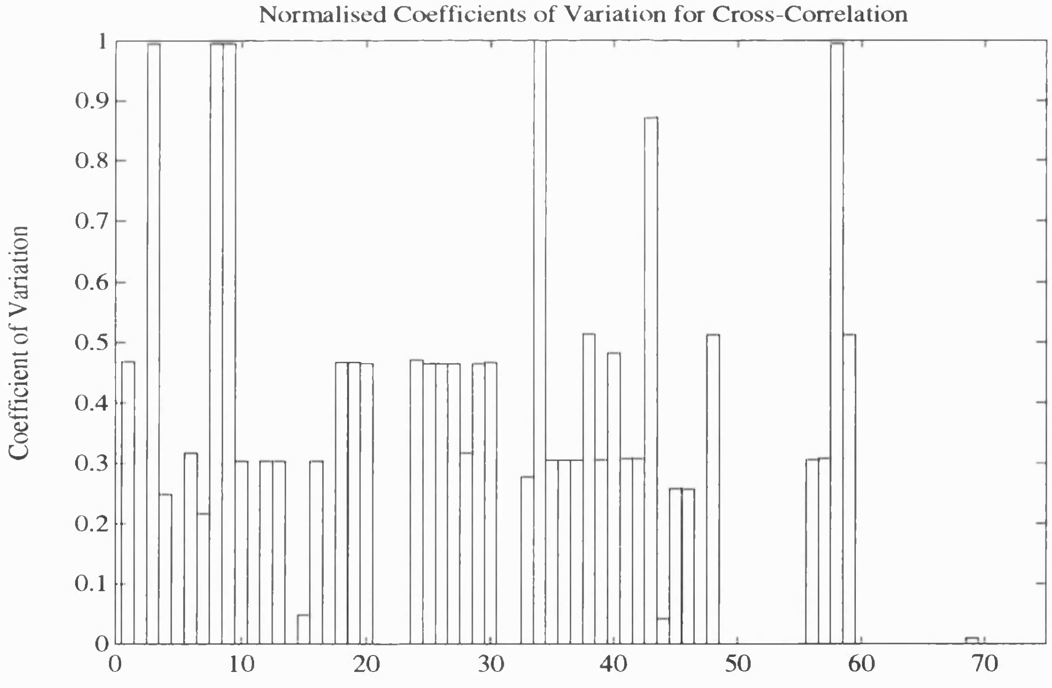
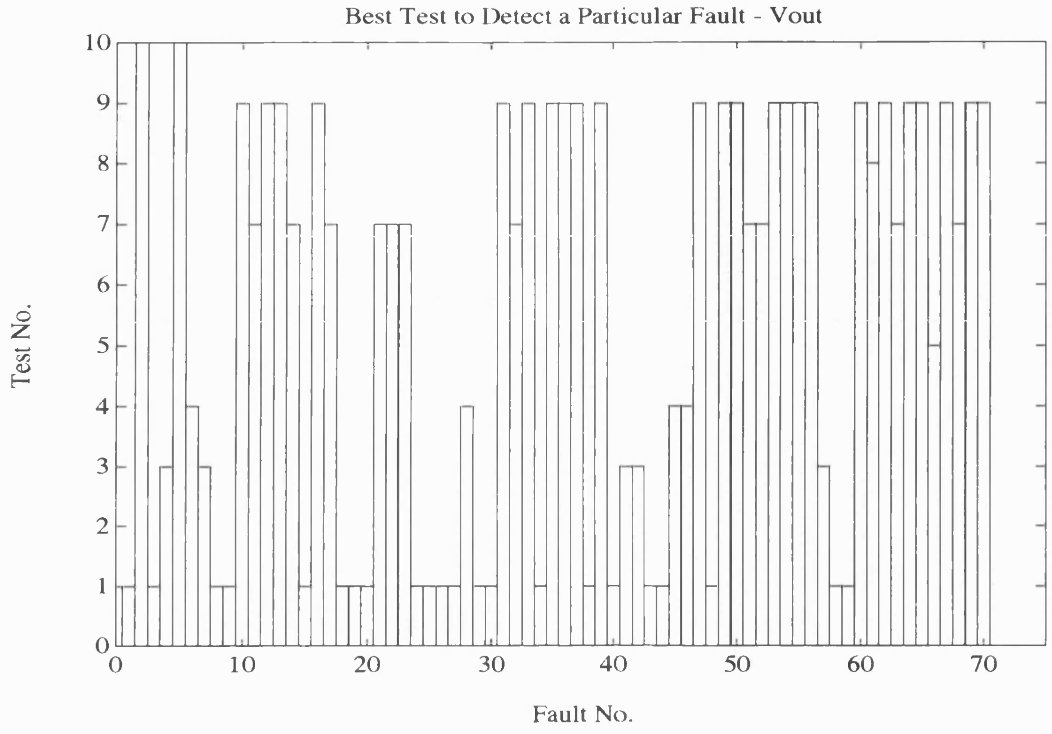


Figure 5.22: Normalised CVs of Auto & Cross Correlations of Fourth-Order LPF



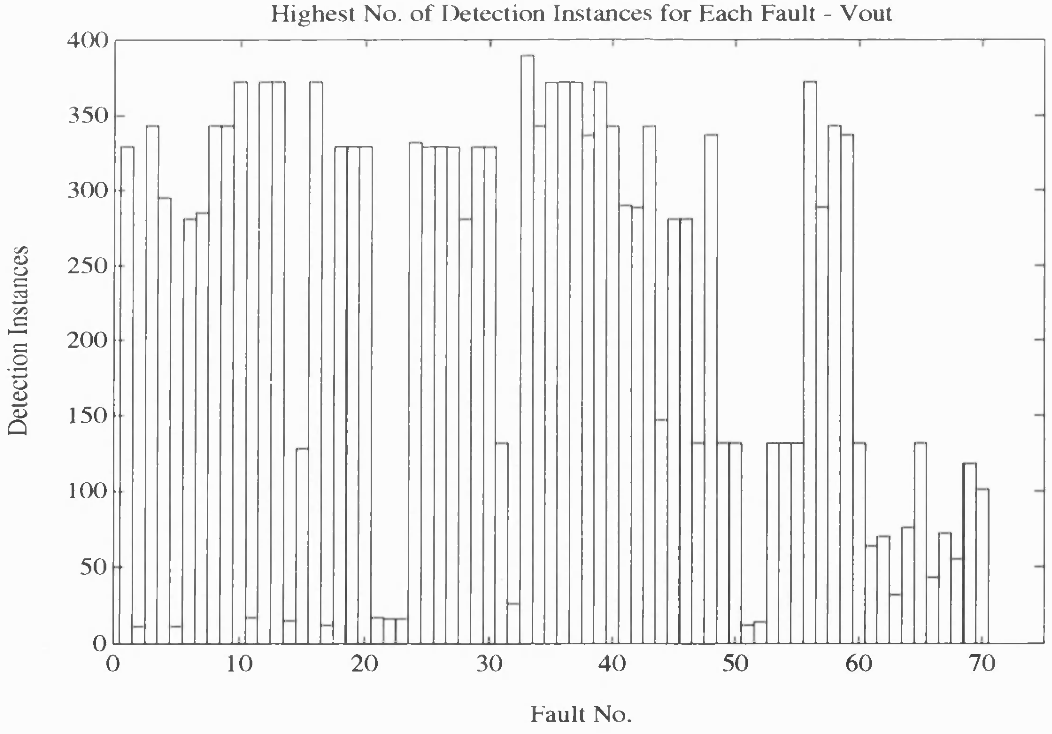
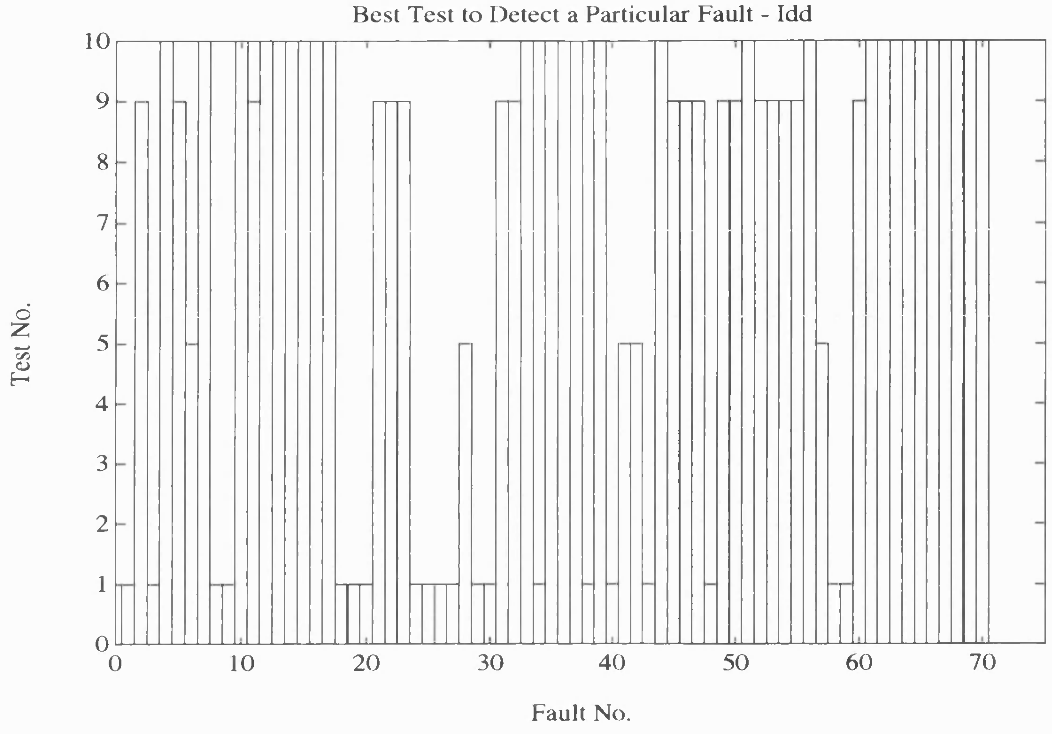


Figure 5.23: Best Test & Highest Detect. by Digitized Vout of Fourth-Order LPF



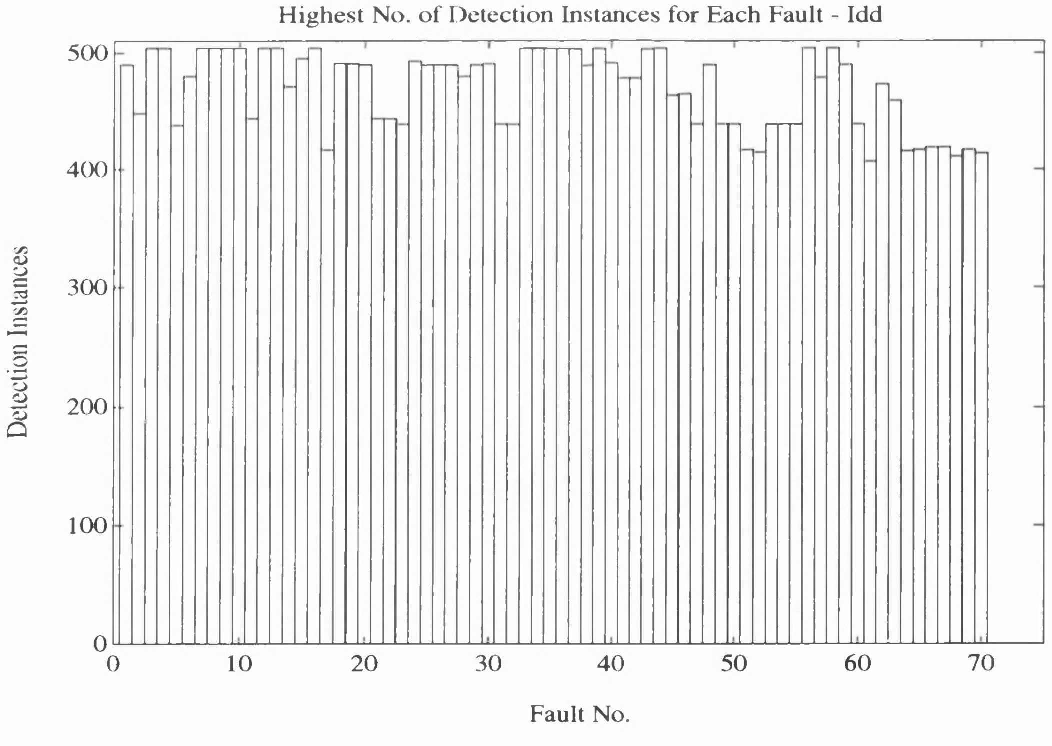


Figure 5.24: Best Test & Highest Detections by Digitized Idd of Fourth-Order LPF

5.3.3 Mixed-Signal Circuit

To demonstrate the effectiveness of the time-domain approach and the analysis methods in testing mixed-signal ICs the circuit shown in Figure 5.25 was simulated and tested. The circuit consists of four modules: a low-pass filter (LPF) with a 3-dB bandwidth of 2-KHz, a sample and hold (SH) circuit, a 2-bits analogue-to-digital converter (ADC), and a full-adder digital logic network. The op-amp used in the LPF module is the same one used in the two previous examples and depicted in Figure 5.6. The schematics of the comparator, analogue switch and individual logic gates are illustrated in Figure 5.26, Figure 5.27 and Figure 5.28 respectively.

The mixed-signal circuit in Figure 5.25 was tested by injecting a 15-bits PRBS test sequence at the LPF input (Vin). The PRBS has a period of 15T, where T is equal to 250 μsec. The transient voltage responses at Vs and Vc, and the supply transient current Idd were sampled every 10 μsec, resulting in 375 samples for each waveform. The PRBS input signal and the fault-free transient responses at Vs, Vc and Idd are illustrated in Figure 5.29.

A total of 140 single fault conditions were simulated. Of these faults 115 were catastrophic faults in the MOS transistors of the various modules, 5 soft faults in the resistive and capacitive components of the LPF, SH and ADC, and 20 stuck-at (s-a-1 and s-a-0) faults at the terminals of the digital logic gates.

Analysis of Vs, Vc and Idd transient response data based on samples values, depicted in Figure 5.30, shows that the three measurements achieve 100% fault-coverage each. To determine which one of the measurements is best at detecting a particular fault the normalised CVs of Vs, Vc and Idd were calculated and plotted in Figure 5.31. The normalised CVs indicate that Vs, Vc and Idd are best at detecting 72, 32 and 36 faults respectively.

Figure 5.32 illustrates the results of applying the rate of change analysis method to Vs, Vc and Idd. It shows that the fault-coverage achieved by each one of the three measurements is 100%.

Due to the presence of two output nodes (Vs and Vc), four correlation functions were calculated; an auto and cross correlation functions for Vs, and an auto and cross correlation functions for Vc. The cross-correlation functions are between the voltage of the respective output node (i.e. Vs or Vc) and the input node voltage (Vin). Based on the instances of detection plotted in Figure 5.33 and Figure 5.34 for Vs and Vc respectively. The fault-coverage of the auto and cross correlation of Vs are equal 85.71% (i.e. 120 fault detected) each, and the correlations of Vc are also equivalent and equal 100% each. The normalised CVs of the four correlation functions are plotted in Figure 5.35 and Figure 5.36 to determine the function best suited to detect a particular fault. Analysis of the CVs in both figures indicate that auto-Vs, cross-Vs, auto-Vc and cross-Vc are each best at detecting 14, 19, 33 and 72 fault respectively, while 2 faults are equally detectable by all functions.

The application of the digitization method of analysis to Vs, Vc and Idd, with 10 tests specified, resulted in a fault-coverage of 74.29%, 72.86% and 100% respectively. The plots of the test best suited to detect particular faults and the corresponding highest number of detection instances for Vs, Vc and Idd are illustrated in Figure 5.37, Figure 5.38 and Figure 5.39 respectively.

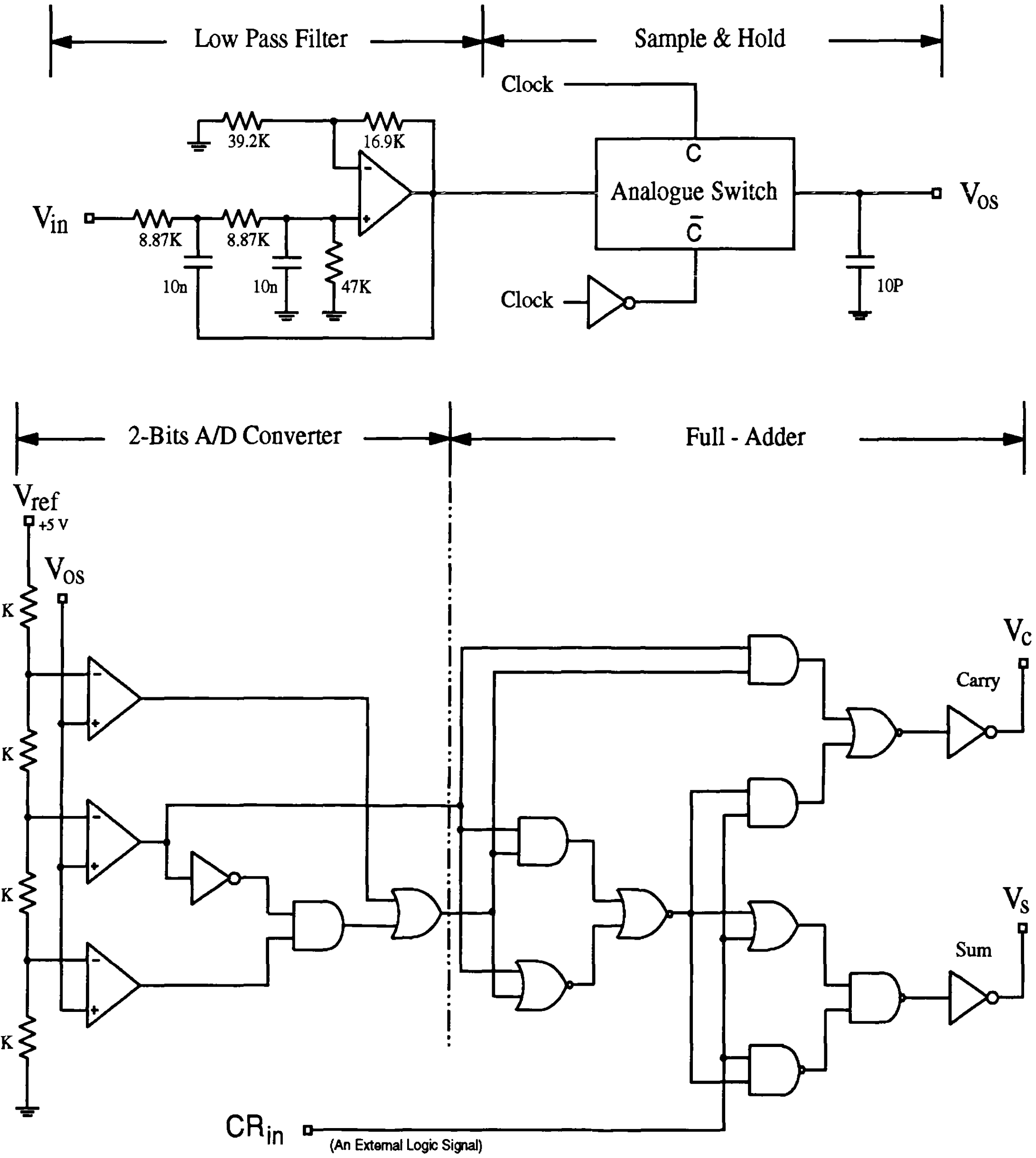


Figure 5.25: Mixed - Signal Circuit

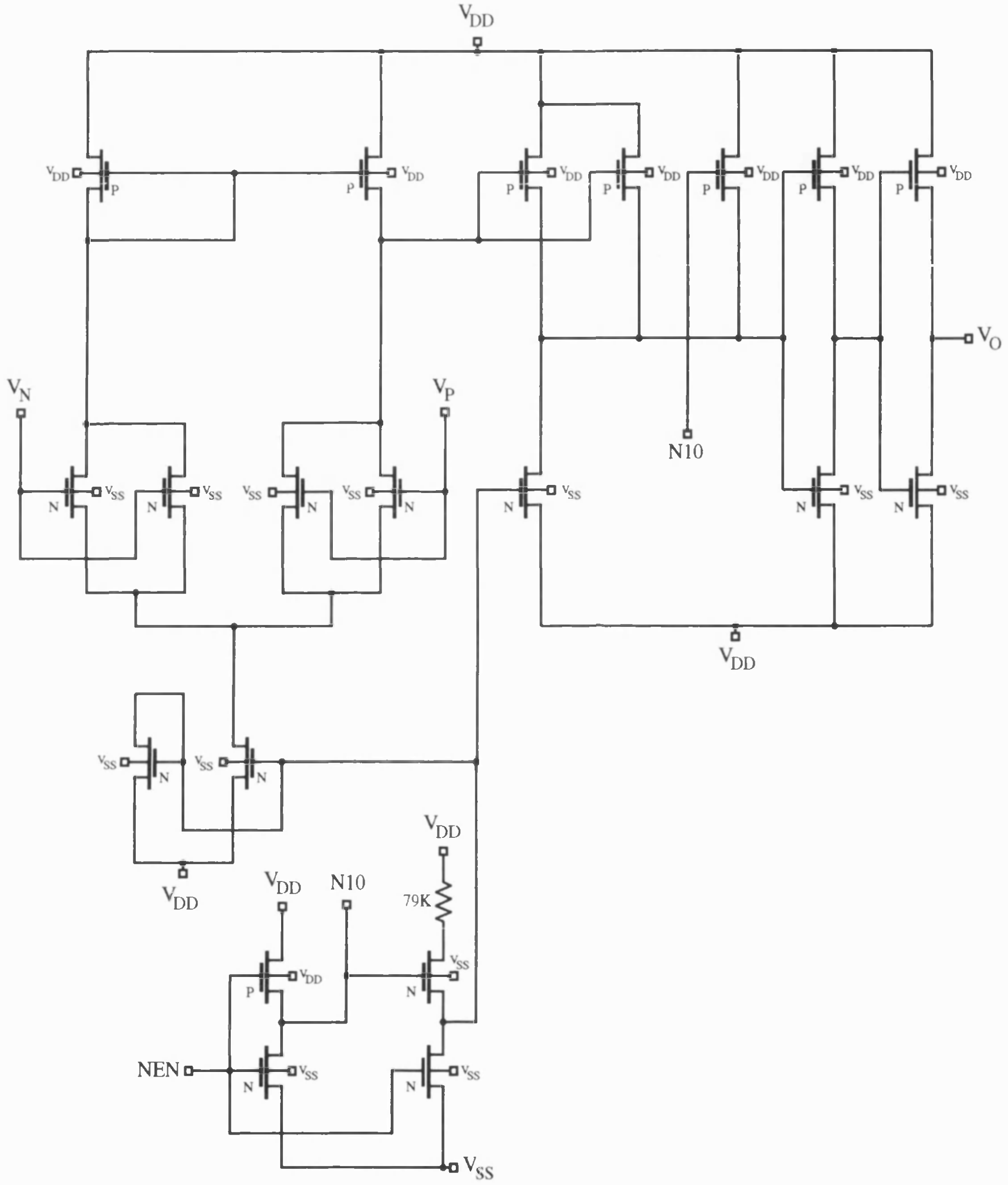


Figure 5.26: MOS Comparator Circuit Used in the Mixed-Signal Circuit [17]

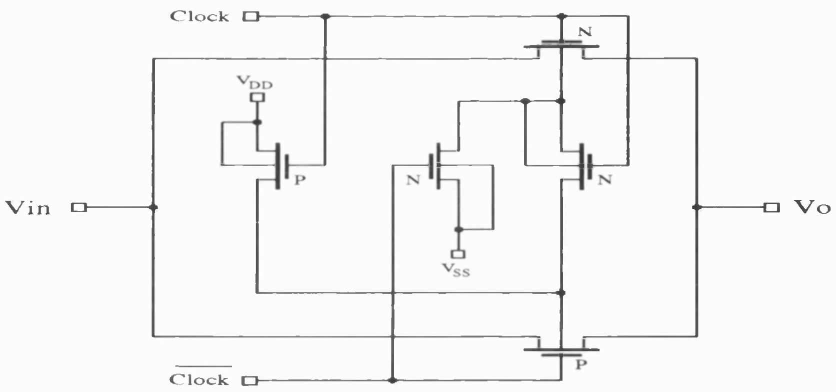


Figure 5.27: CMOS Analogue-Switch Circuit [18]

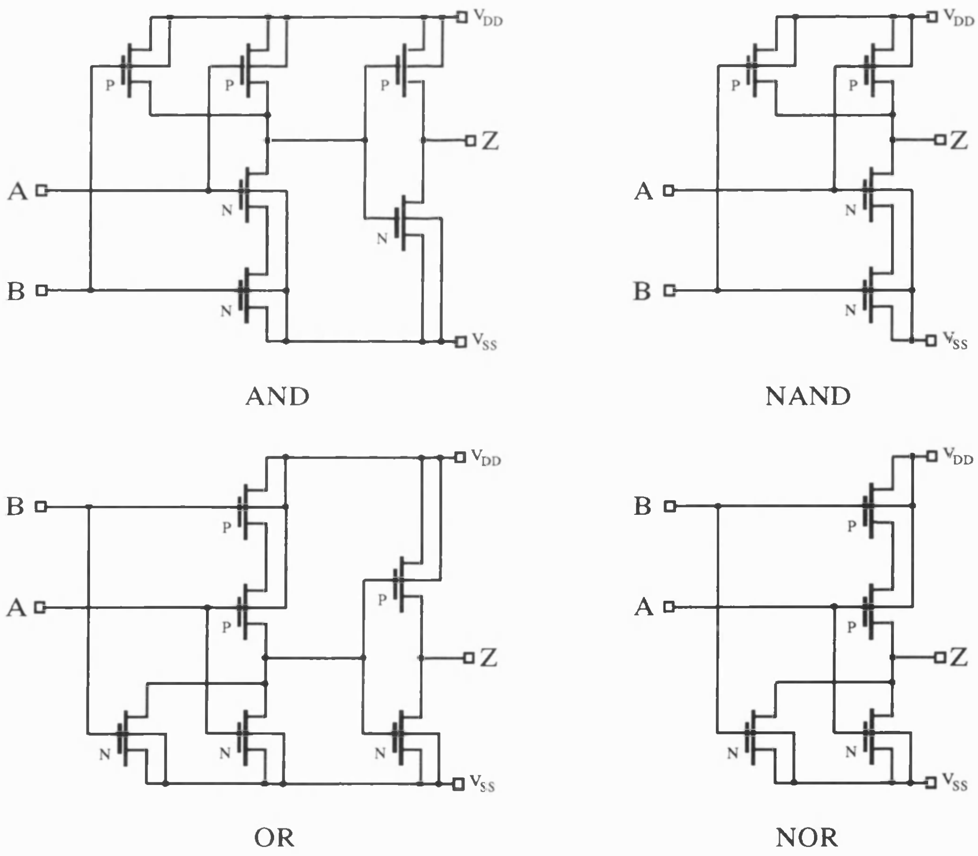


Figure 5.28: Schematics of AND, NAND, OR and NOR CMOS Logic Gates

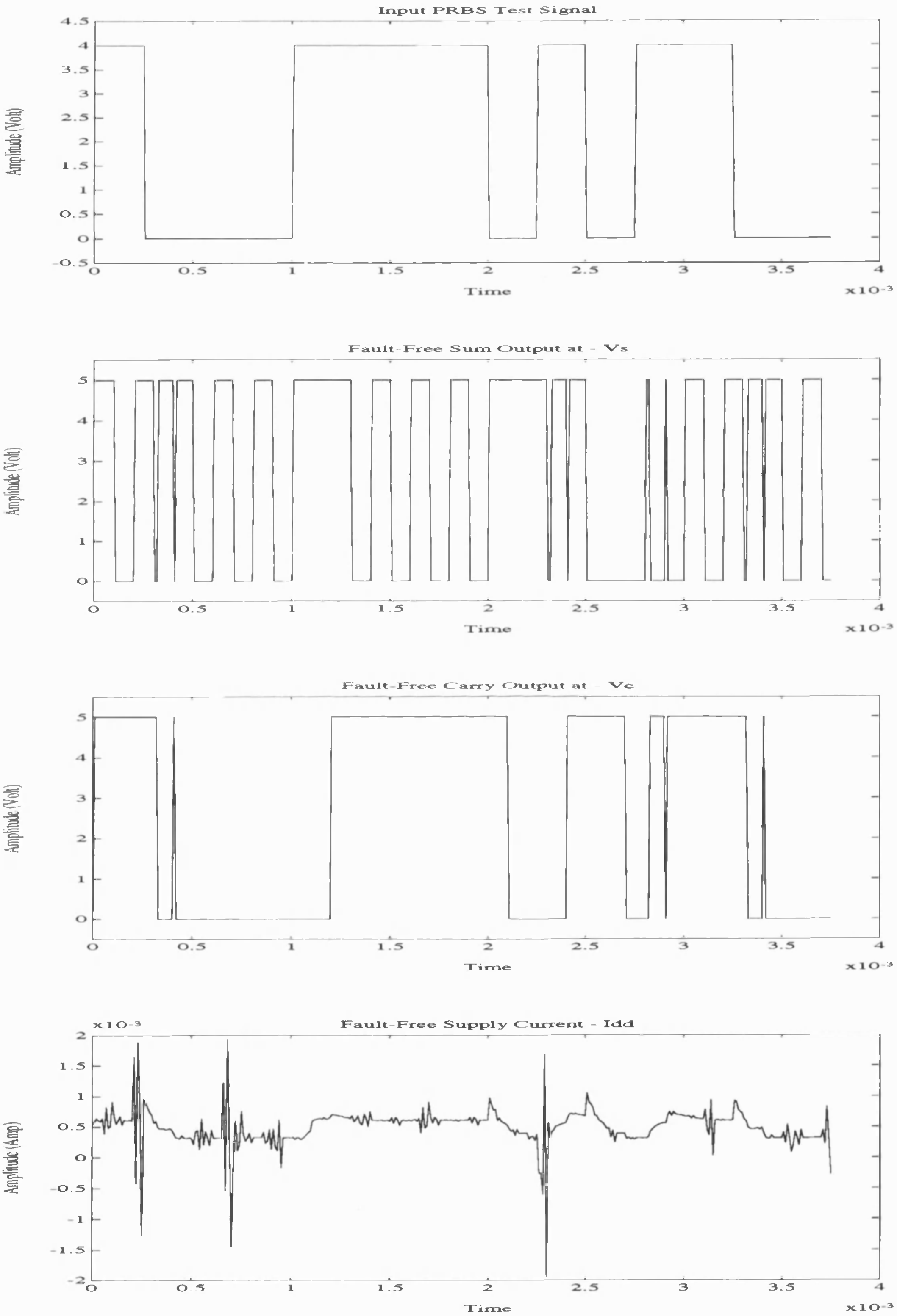


Figure 5.29: Input PRBS and Fault-Free Responses Vs, Vc & Idd

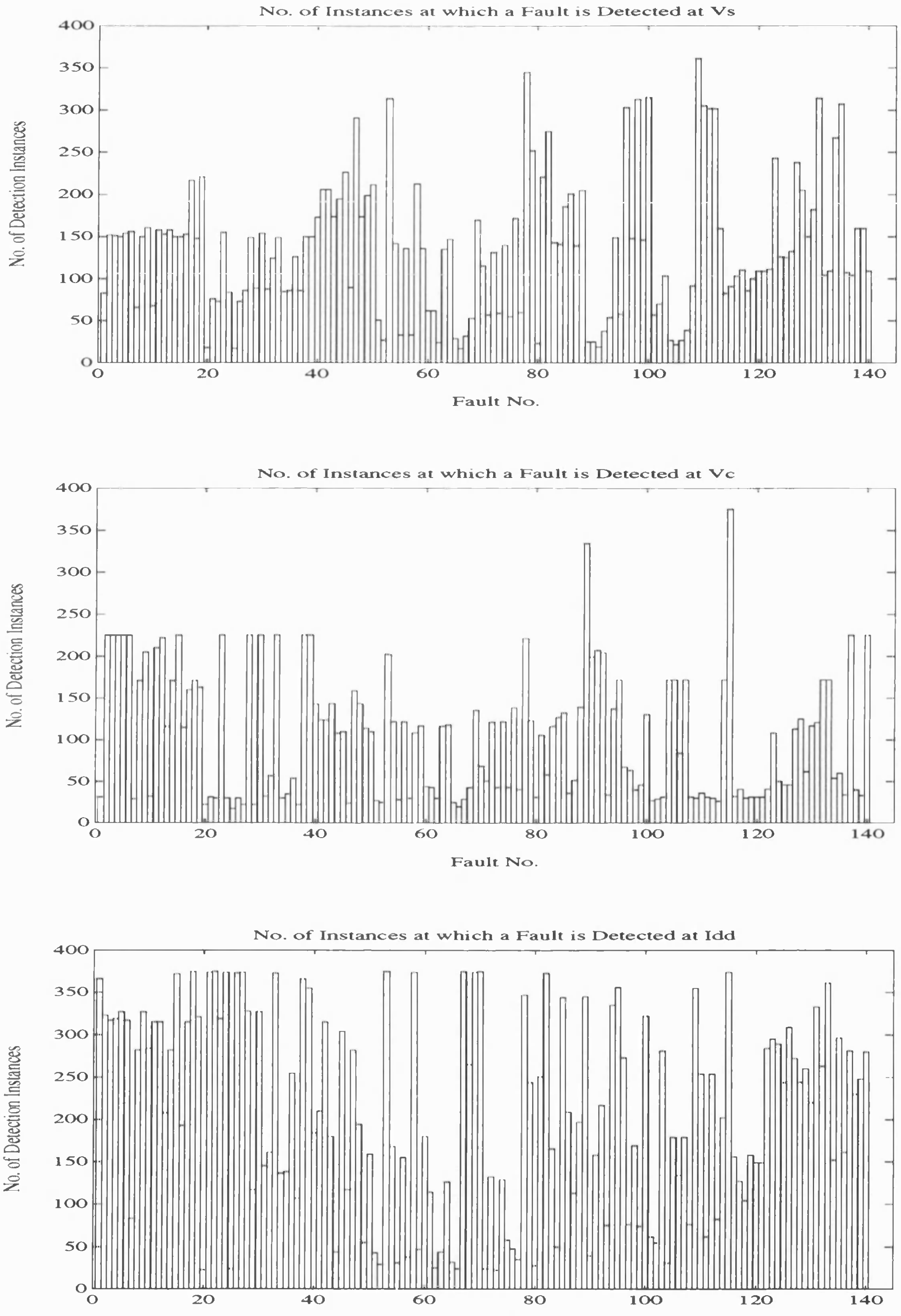


Figure 5.30: Detection Instances by Vs, Vc & Idd of Mixed-Signal Circuit

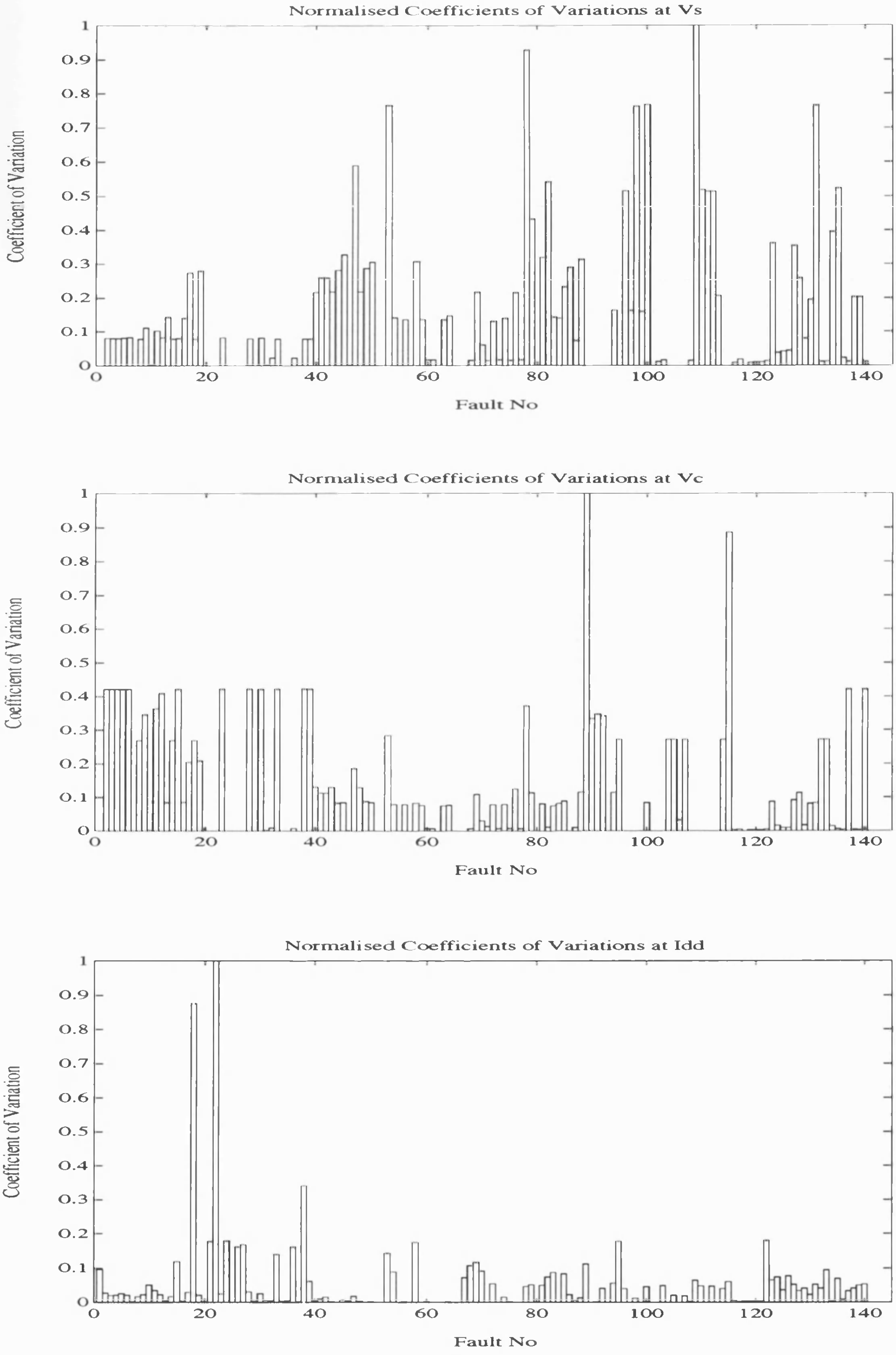


Figure 5.31: Normalised CVs of Vs, Vc & Idd of Mixed-Signal Circuit

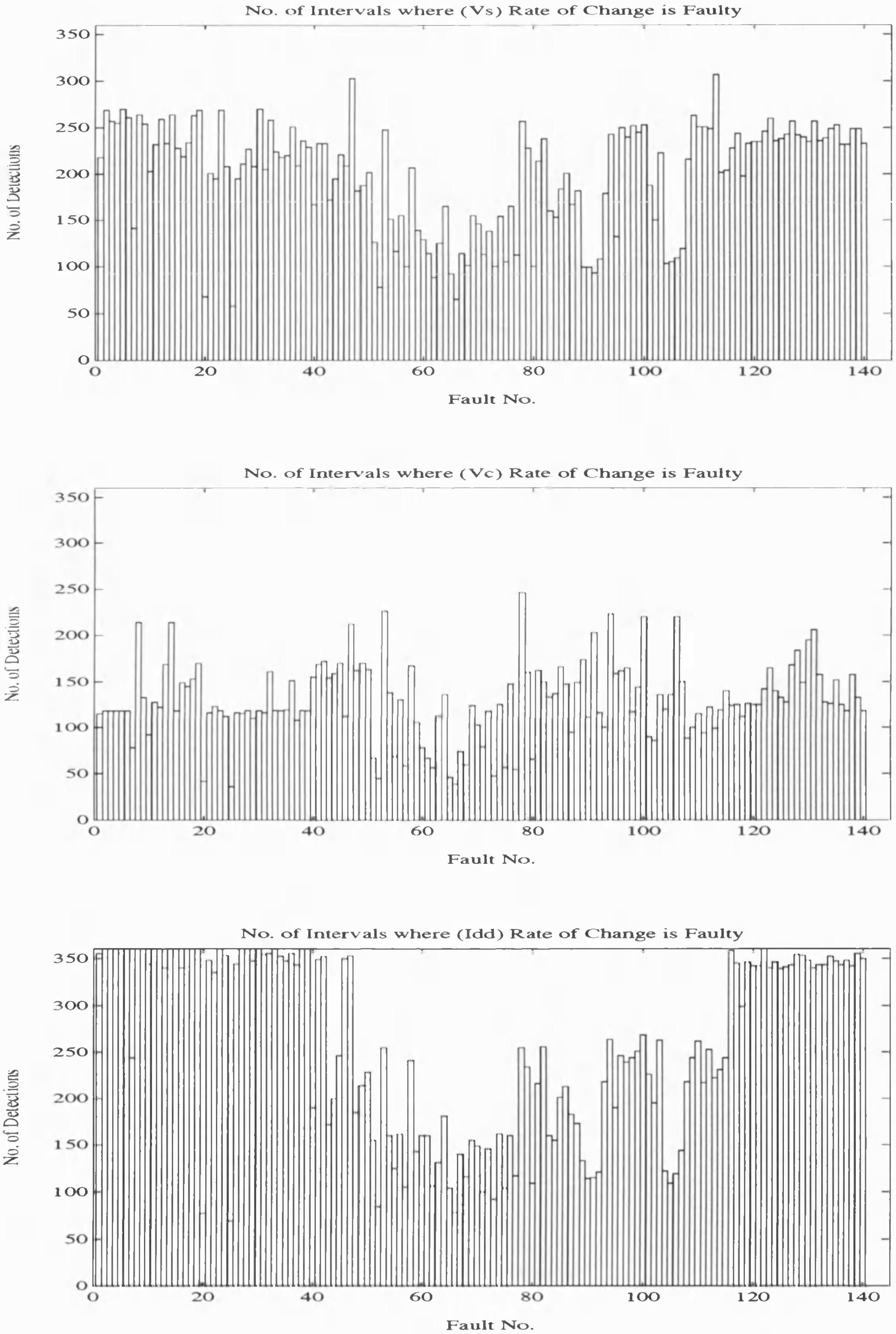
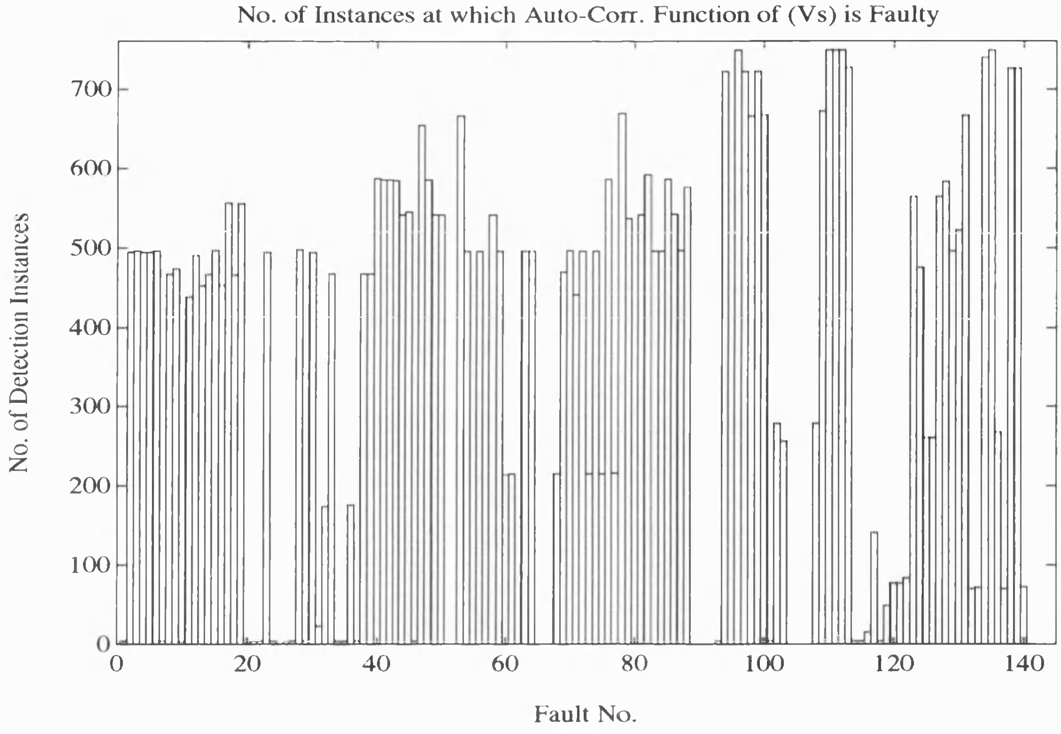


Figure 5.32: Det. Intervals by Vs, Vc & Idd Rate of Change of Mixed-Signal Cct.



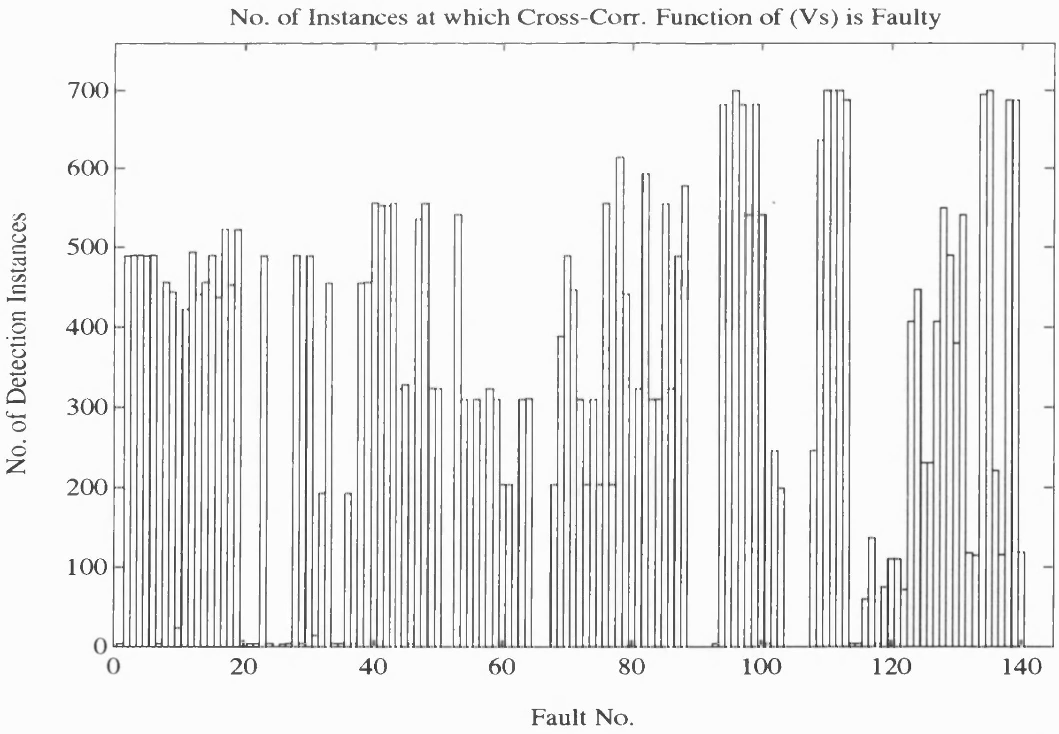
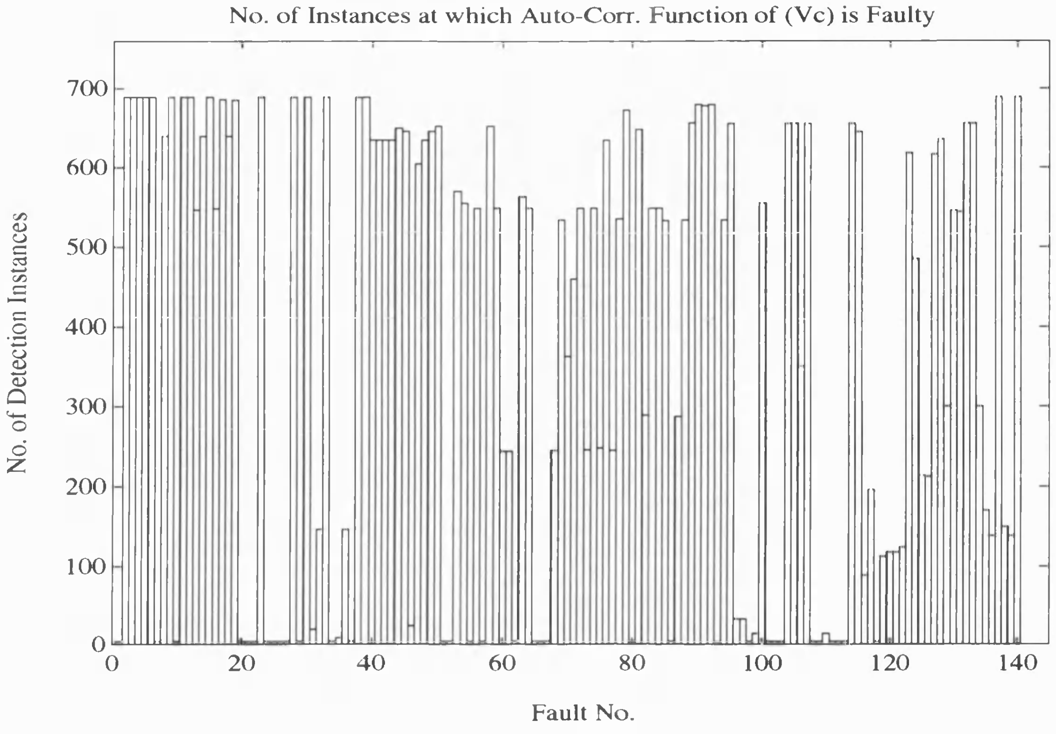


Figure 5.33: Det. Instan. by (Vs) Auto & Cross Correlations of Mixed-Signal Cct.



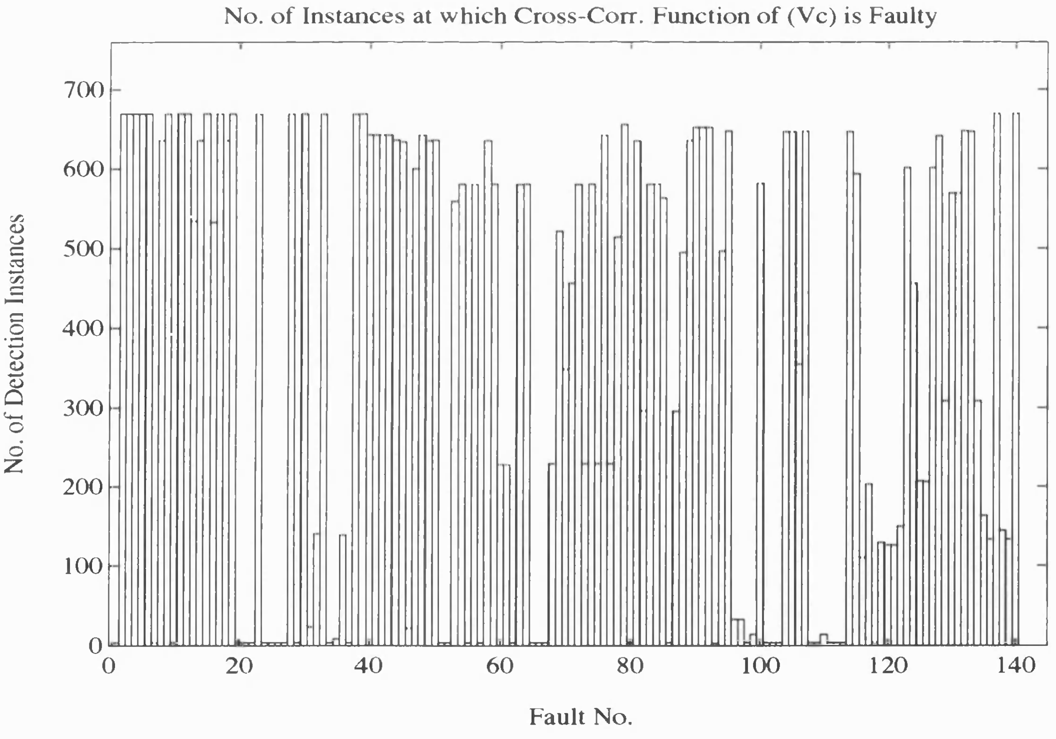
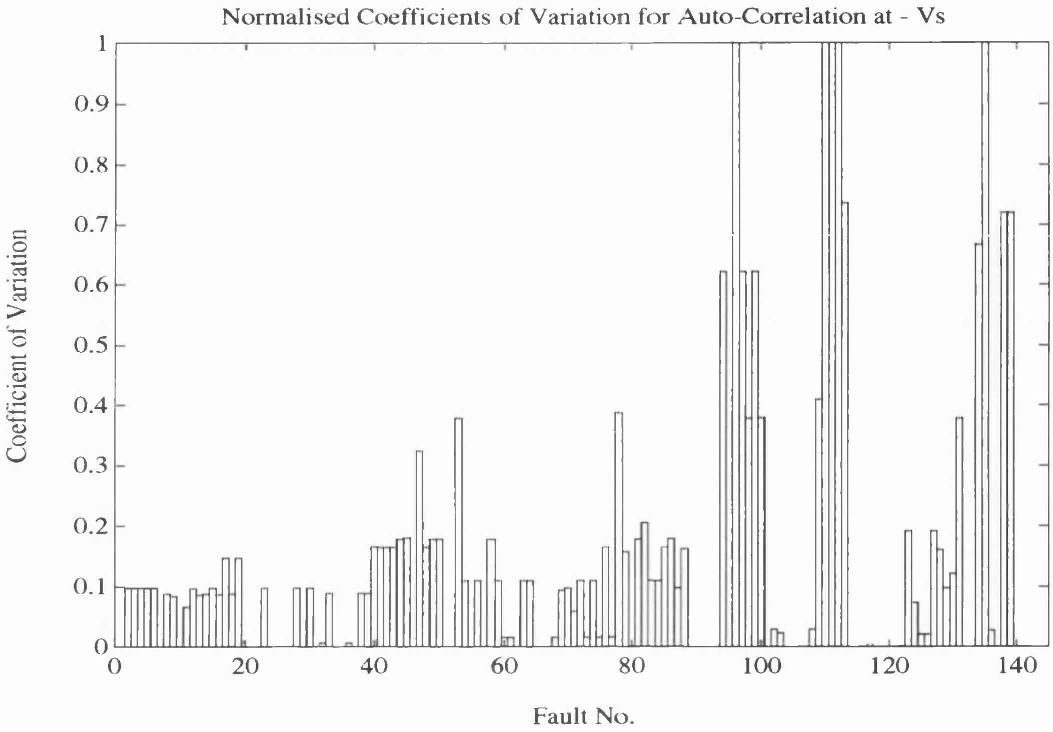


Figure 5.34: Det. Instan. by (Vc) Auto & Cross Correlations of Mixed-Signal Cct.



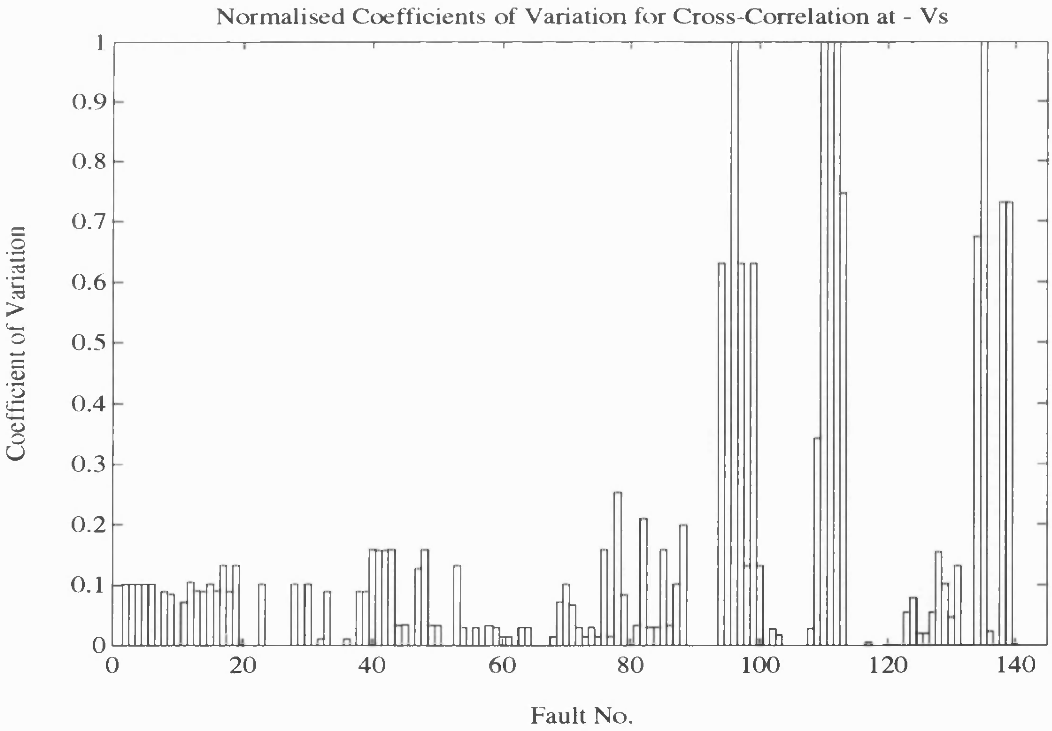
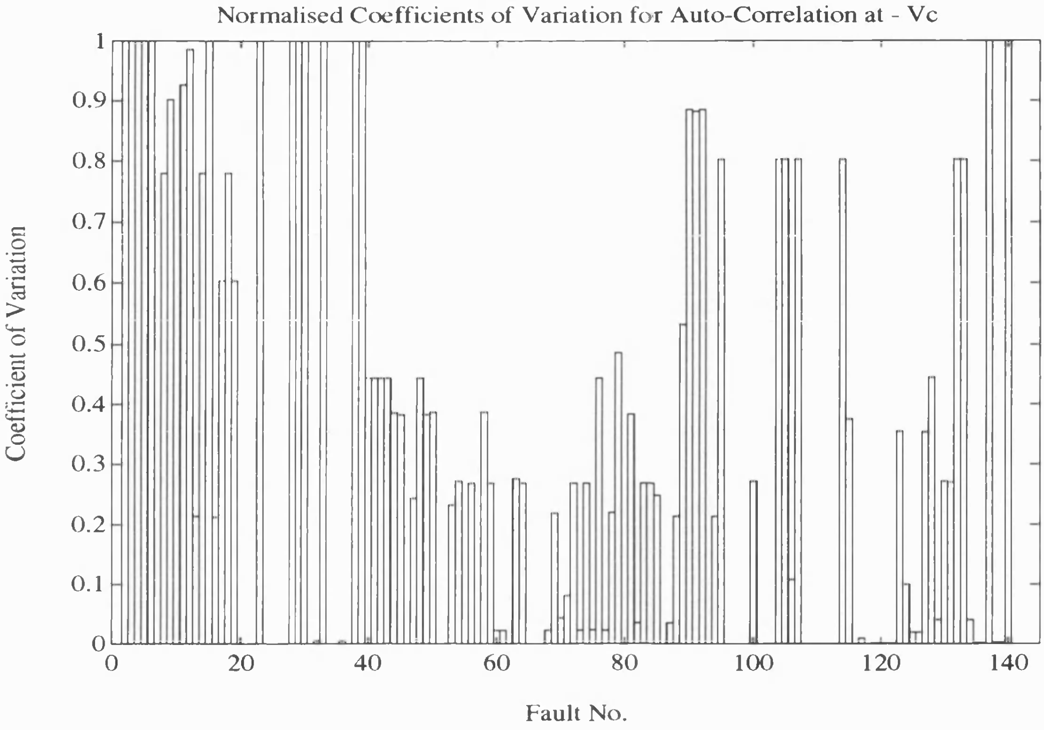


Figure 5.35: Norm. CVs of (Vs) Auto & Cross Correlations Mixed-Signal Cct.



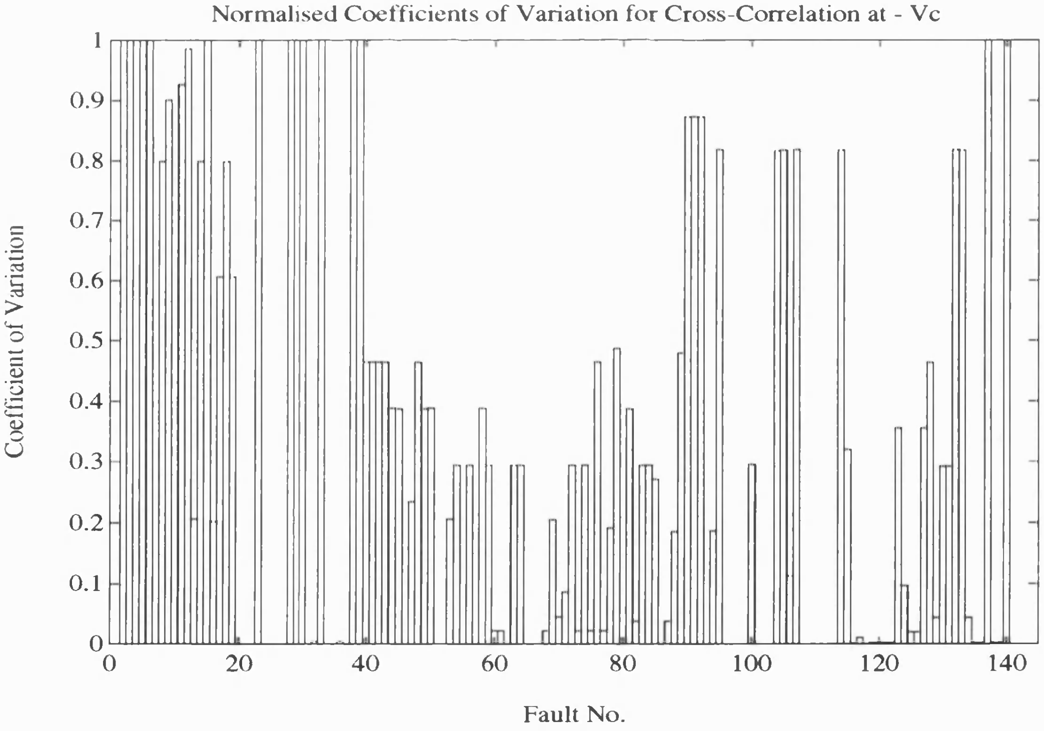


Figure 5.36: Norm. CVs of (Vc) Auto & Cross Correlations of Mixed-Signal Cct.

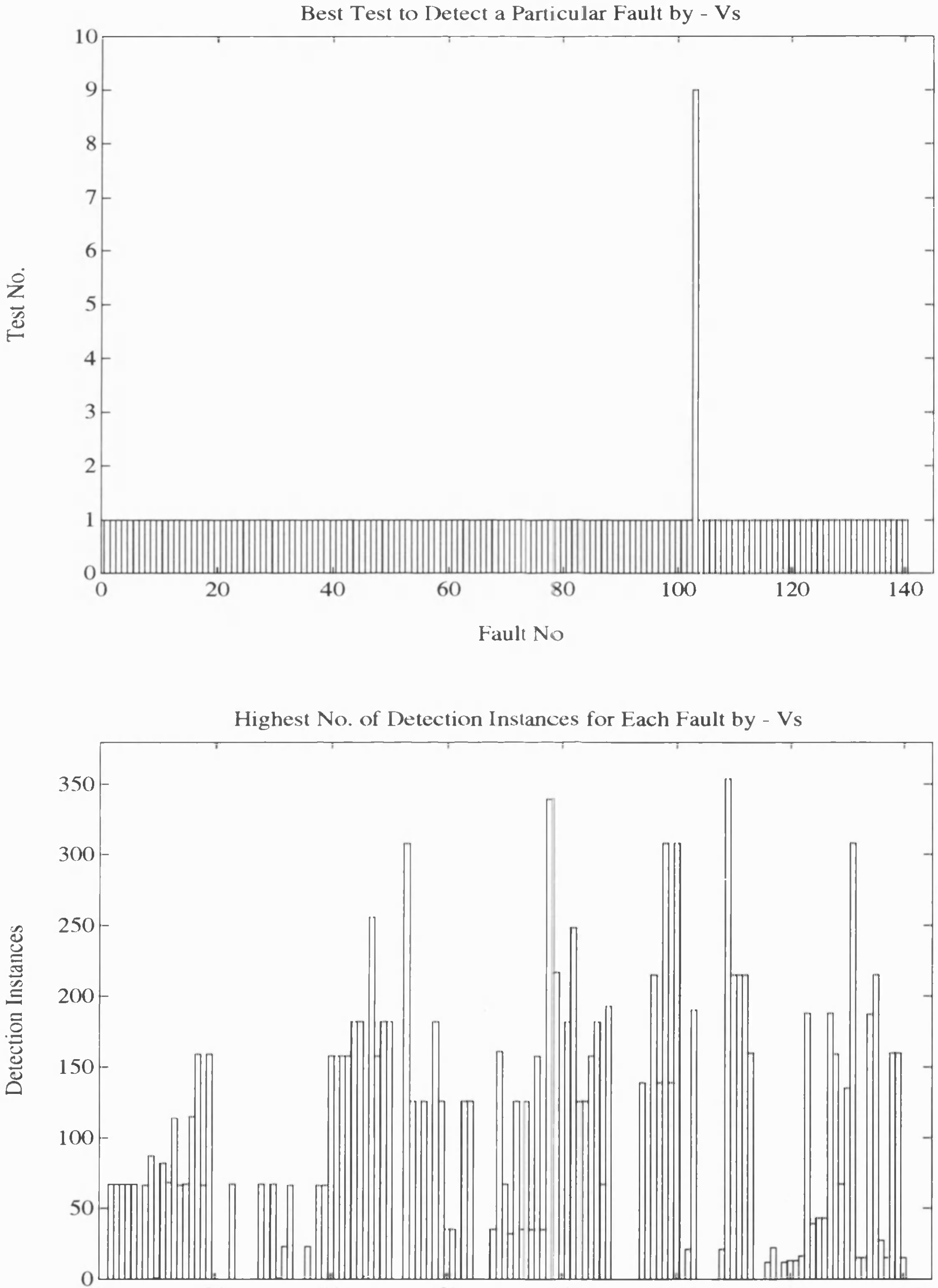
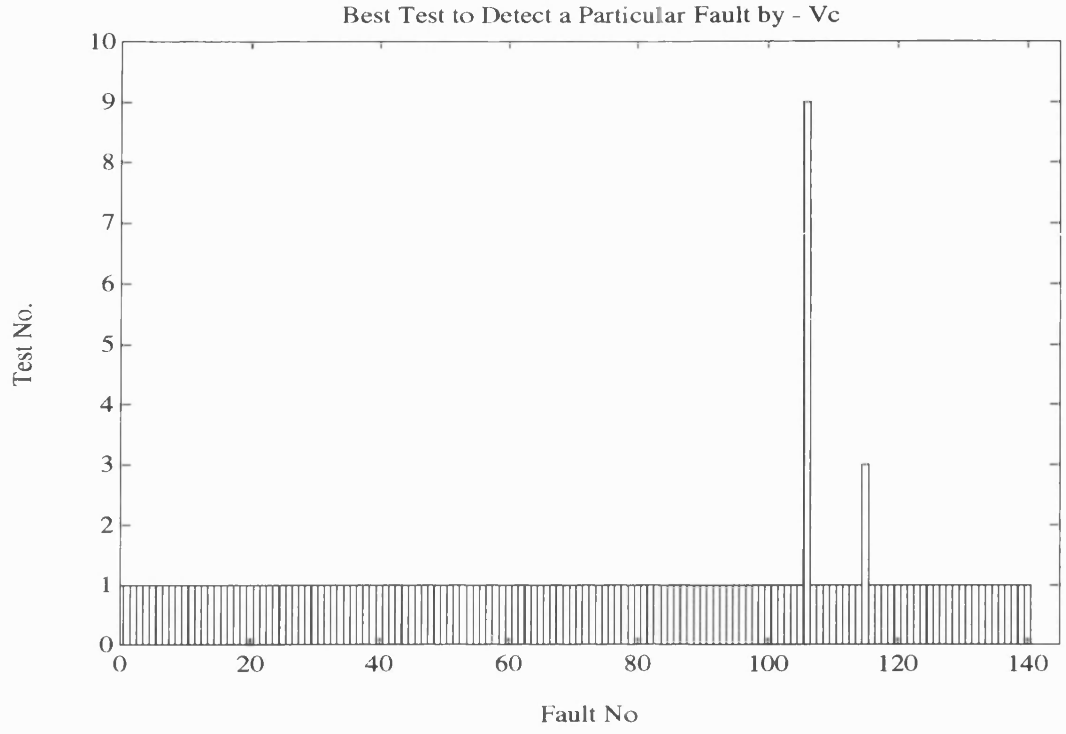


Figure 5.37: Best Test & Highest Detections by Digitized Vs of Mixed-Signal Cct.



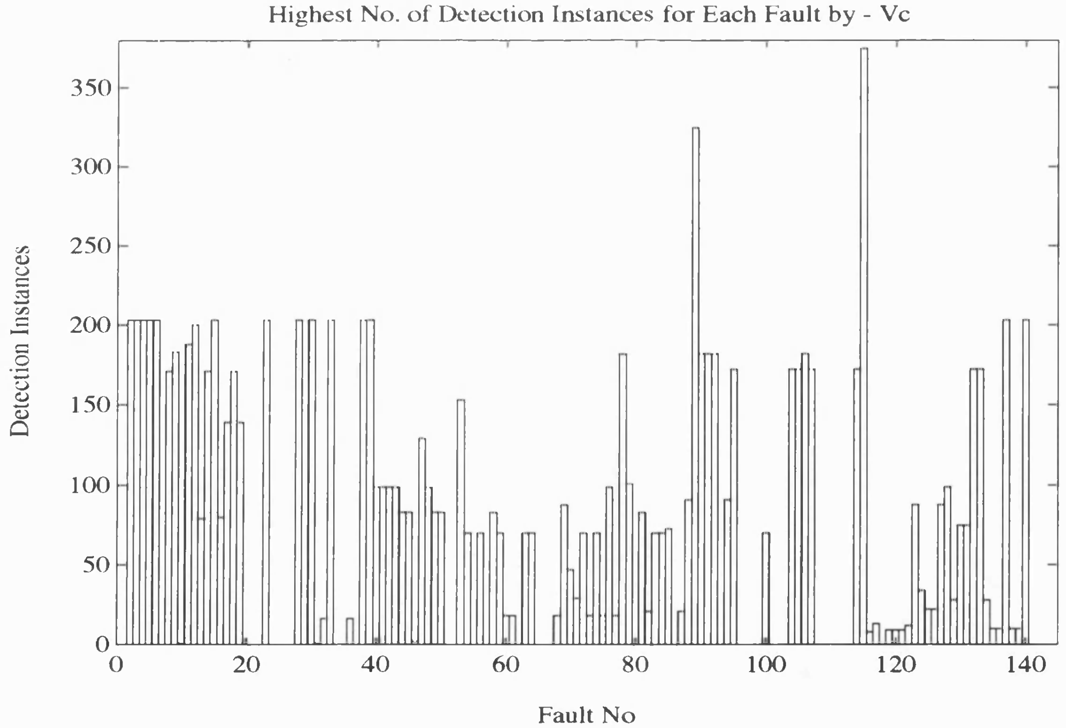
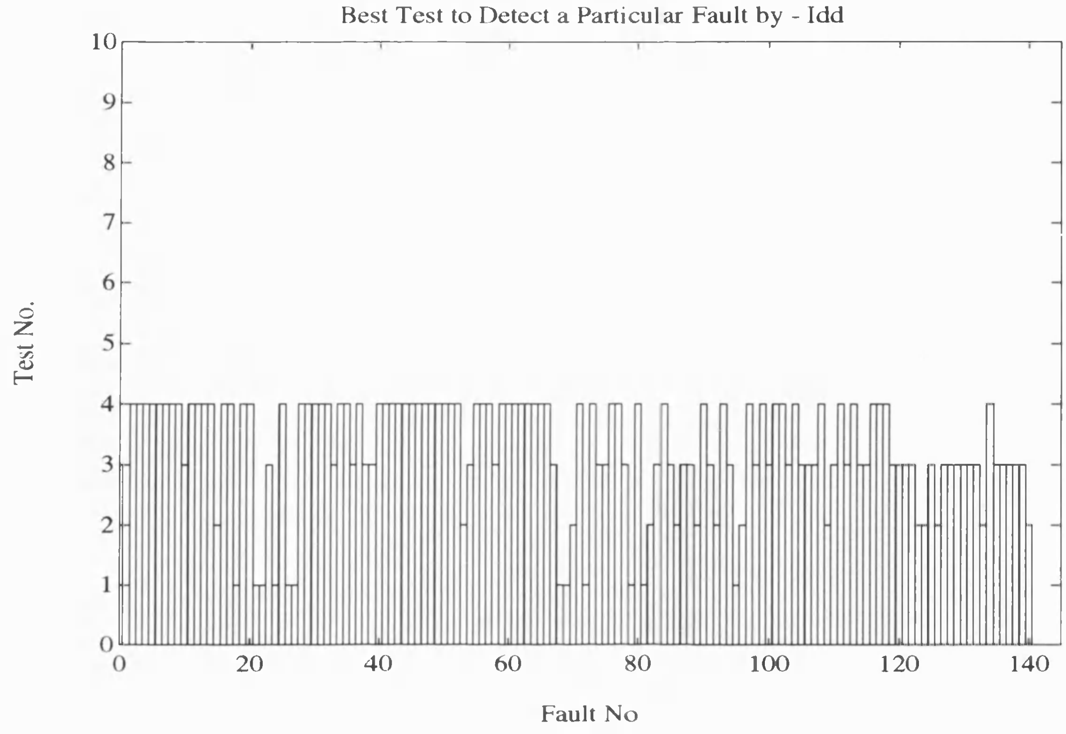


Figure 5.38: Best Test & Highest Detections by Digitized Vc of Mixed-Signal Cct.



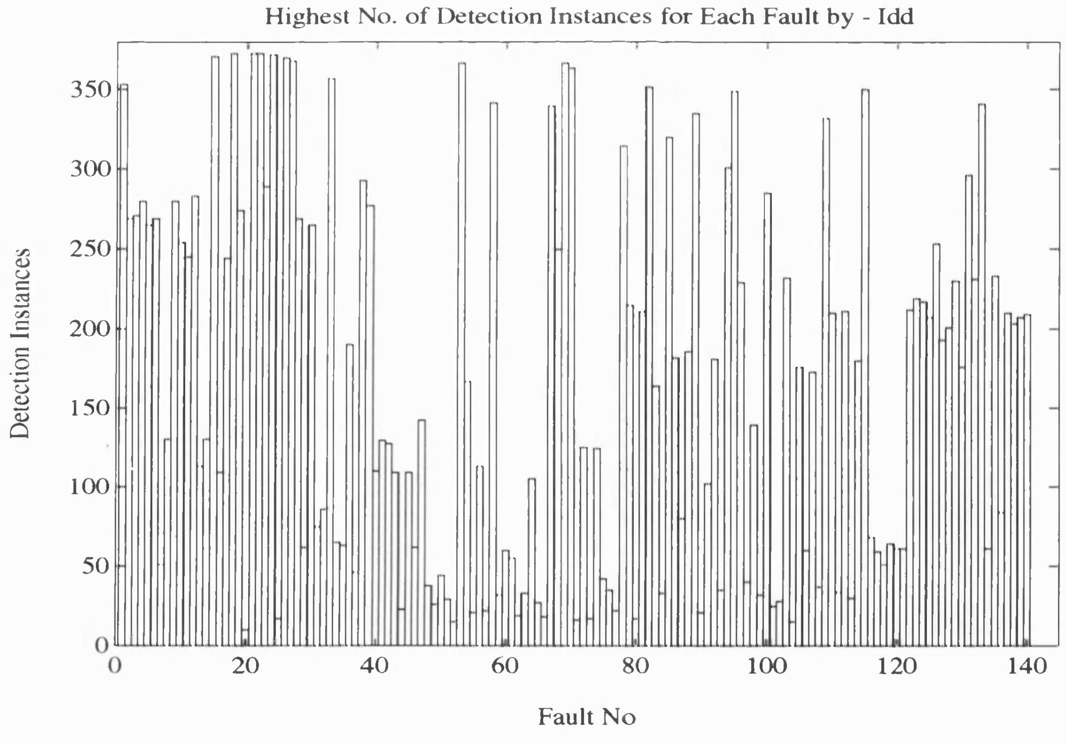


Figure 5.39: Best Test & Highest Detections by Digitized Idd of Mixed-Signal Cct

5.4 Summary

The simulation results of the analogue and mixed-signal circuits demonstrated that the time-domain testing technique is an effective unified approach for testing both types of circuits without the need for partitioning. For all the circuits simulated and tested, the results of the four methods of analysis indicate a high percentage of fault-coverage is achieved by the time-domain technique.

Analysis of the results indicate that both transient voltage and current measurements are complementary in terms of achieving a high percentage of fault-coverage with a high degree of confidence. Of the four methods of data analysis, the response digitization method is the most efficient in terms of computation, because it eliminates the need for floating point computation. The method can also be implemented on a digital tester, hence resulting in saving in both the time and cost of testing mixed-signal integrated circuits.

The simulation results detailed in this chapter are discussed in more detail in Chapter 7.

5.5 References

[1] M.A. Al-Qutayri, P.S. Evans and P.R. Shepherd, "Testing Mixed Analogue/Digital Circuitry Using Transient Response Techniques", 7th European Design for Testability Workshop, Segovia - Spain, June 1990.

[2] P.S. Evans, M.A. Al-Qutayri and P.R. Shepherd, "On the Development of Transient Testing Techniques for Mixed-Mode IC's", Journal of Semicustom ICs, Vol. 8, No. 2, pp. 34-38, September 1990.

[3] P.R. Shepherd, M.A. Al-Qutayri and P.S. Evans, "Testing Mixed-Signal Integrated Circuits", IEE Colloquium on Design and Test of Mixed Analogue/Digital ICs, Savoy Place, London, 15 November 1990.

[4] P.S. Evans, M.A. »Al-Qutayri and P.R. Shepherd, "A Novel Technique for Testing Mixed-Signal IC's", Proc. European Test Conference, Munich, Germany, pp. 301-6, April 1991.

[5] R. Fritzemeier, J. Soden, R. Treece and C. Hawkins, "Increased CMOS IC Stuck-At Fault Coverage with Reduced IDDQ Test sets", International Test Conference, pp. 427-435, September 1990.

[6] P. Nigh and W. Maly, "Test Generation for Current Testing", IEEE Design and Test of Computers, pp. 26-38, February 1990.

[7] I. Bell, D. Camplin, G. Taylor and B. Bannister, "Supply Current Testing of Mixed Analogue and Digital ICs", Electronic Letters, Vol.27, pp. 1581-83, August 1991.

[8] H. Schreiber, "Fault Dictionary Based upon Stimulus Design", IEEE Trans. Circuits and Systems, Vol. CAS-26, No. 7, pp. 529-537, July 1979.

[9] S. Golomb, Digital Communications with Space Applications, Peninsula Publishing, Los Altos, California, 1964.

[10] F. Stremler, Introduction to Communication Systems, Addison Wesley, Reading, Massachusetts, 1982.

[11] P.R. Shepherd and M.A. Al-Qutayri, "A Time-Domain Strategy for Testing Mixed-Signal IC's", 6th U.K. Design Automation Workshop, Hilton Hotel, Bath, May 1991.

[12] M.A. Al-Qutayri and P.R. Shepherd, "PRBS Testing of Analogue Circuits", IEE Colloquium on Testing Mixed-Signal Circuits, Savoy Place, London, May 1992.

[13] M.A. Al-Qutayri and P.R. Shepherd, "Go/No-Go Testing of Analogue Macros", IEE Proceedings on Circuits, Devices and Systems - Part G, Vol. 139, No. 4, pp. 534-540, August 1992.

[14] HSPICE User's Manual, Meta-Software Inc., Campell, California 1988.

[15] MATLAB User's Guide, The MathWorks Inc., South Natick, MA, 1989.

[16] J. Compton, Silicon Micro Systems Ltd, Malmesbury, U.K., Private Communications, March 1991.

[17] R. Cobley, University of Exeter, U.K., Private Communications, August 1990.

[18] P.E. Allen and D.R. Holberg, CMOS Analog Circuit Design, Holt Rinehart and Winston, New York, 1987.

CHAPTER SIX

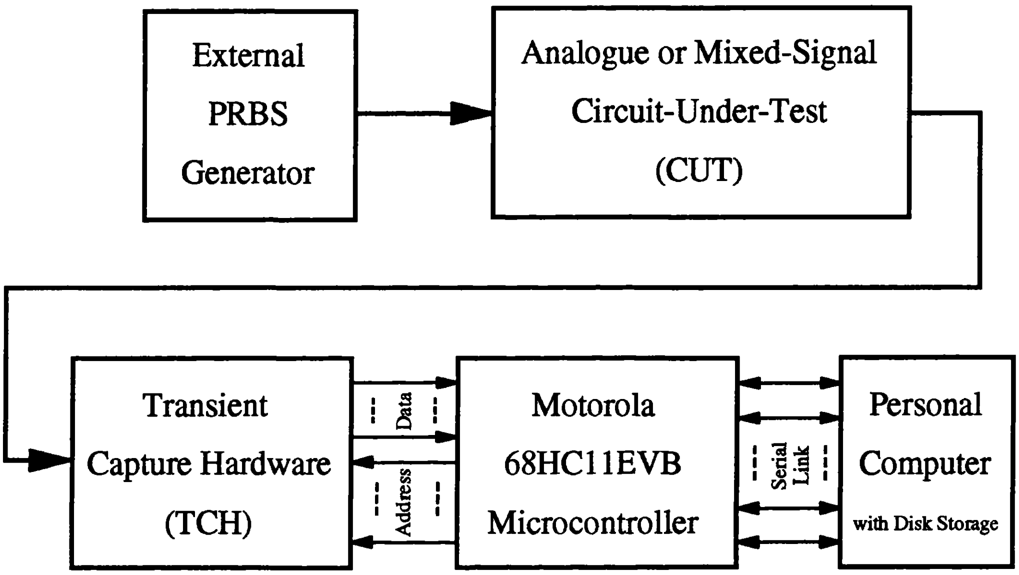
EXPERIMENTAL SYSTEM AND RESULTS

In Chapter 5 the time-domain testing technique and the four methods of analysing the data extracted from a circuit-under-test (CUT) were demonstrated by simulating analogue and mixed-signal circuits. This chapter describes the prototype experimental system, and demonstrates the effectiveness of the technique and analysis methods by performing measurements on physical analogue and mixed-signal circuits.

6.1 Experimental System

To enable the testing of physical circuits an experimental system was designed. The block diagram of the system is illustrated in Figure 6.1. The system consists of three major blocks: transient capturing hardware (TCH), a personal computer (PC) and an interface between the PC and TCH. The interface was implemented by the 68HC11EVB microcontroller from Motorola. Upon detecting the transient response signal from the CUT, the TCH unit performs the transient capturing process and stores that transient signal in its RAM (random-access-memory), then the microcontroller interface unit signals to the PC that the transient response data is ready. The microcontroller then uploads the data stored in the TCH unit RAM to the PC when a request for that is issued by the PC. Once the data is uploaded to the PC, it gets converted to a format compatible with MATLAB to allow the application of the data analysis method required. Details of the design and implementation of the prototype experimental system in Figure 6.1 are in [1].

The following sections give details of the experimental results of testing an analogue and a mixed-signal circuit. Only transient voltage measurements are performed because the experimental system is not capable of performing transient current (Idd) measurements. Therefore, all the results and data analysis will be based on the CUT transient voltage output measurements. For each test run only 1000 data samples are up loaded to the PC due to physical design limitations and the excessive length of time it takes the PC to process a high number of data samples.



External PRBS Generator
Analogue or Mixed-Signal Circuit-Under-Test (CUT)
Transient Capture Hardware (TCH)
Data
Address
Motorola 68HC11EVB Microcontroller
Serial Link
Personal Computer with Disk Storage

Figure 6.1: Block Diagram of the Experimental System

As the purpose of the experiment is to detect faults within an integrated circuit, ideally what is required is a series of ICs in which there are faulty circuits, and in which the origin of the fault is known. This set of samples would be very difficult to come by in the normal course of events, and the facilities to manufacture a series of test circuits with deliberately introduced processing faults was not available. Therefore, both the analogue and mixed-signal circuits tested were breadboarded using discrete components. This has enabled the introduction of soft faults by introducing resistors and capacitors with values different from the nominal values, and catastrophic faults by open and short circuiting particular nodes. In the case of the integrated circuit components used (e.g. opamps, comparators and logic gates), the faults introduced were confined to the terminals of these devices because it is physically not possible to introduce faults to the internal transistors of such ICs. Although clearly this method of intoducing faults is not an ideal solution, as it does not exactly match the sort of faults occurring in IC processes, it does go some way to proving the validity and usefulness of the measurement technique.

The effects of the PRBS bit interval (T) and sequence length (N) on the detectability of faults are studied later on in the chapter by performing actual measurements on a faulty analogue circuit.

6.2 Analogue Circuit

The analogue circuit tested is shown in Figure 6.2, the same as the one in Chapter 5. It is a fourth-order low-pass filter with a 3-dB bandwidth of 10KHz.

The circuit in Figure 6.2 was tested by applying a PRBS test sequence 63-bits long and having a bit interval of 50μsec (i.e. 20KHz). The transient response of the circuit at the output (Vo) was sampled at 200KHz. The 20KHz bit frequency of the PRBS enables the placement of 63 Fourier components of high amplitude around the 10KHz, cut-off frequency of the filter, which will be sensitive to faults.

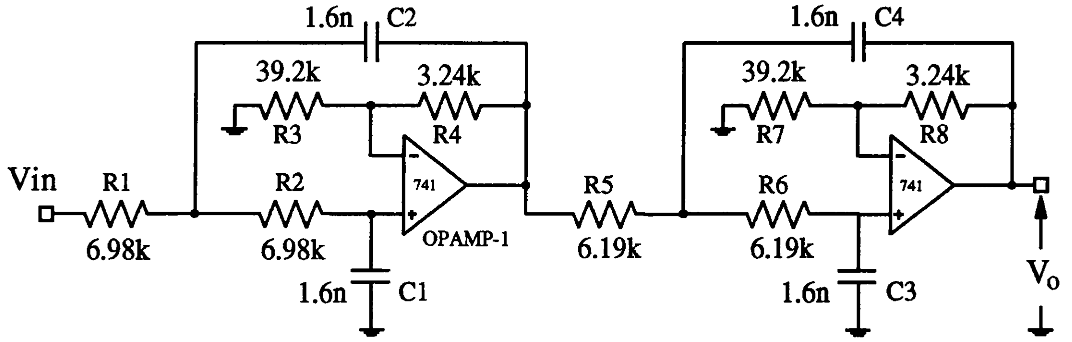


Figure 6.2: Fourth-Order Low-Pass Filter Circuit Tested Experimentally

As for the choice of the PRBS length and the sampling frequency, they are a compromise that is dictated by the number of samples that can be uploaded. High sampling rates lead to high resolutions and the ability to detect small variations between the fault-free response and that of the CUT. Long PRBS sequences are closer to white noise characteristics and hence improve fault detectability as will be explained in a later section.

A total of 20 single fault conditions, listed in Table 6.1, were introduced to the filter circuit in Figure 6.2. The faults include soft and catastrophic faults in the resistive and capacitive components, and catastrophic faults associated with the terminals of the opamps.

The transient responses at (Vo) of the CUT under fault-free and the faulty conditions were processed by the sample values, rate of change, correlation and digitization methods of data analysis presented in Chapter 5. All the methods resulted in a fault-coverage of 100% (i.e. all 20 faults were detected).

Results of applying the samples values and rate of change methods are shown in Figure 6.3 and Figure 6.4 respectively. Both plots indicate a high detection rate.

The auto and cross correlation results are depicted in Figure 6.5. The number of detection instances shows that the correlation functions are virtually equivalent in their ability to detect faults. This result is supported by the correlation normalised coefficients of variations plots in Figure 6.6.

Application of the response digitization method resulted in the best test and the corresponding highest instances of detection bar charts depicted in Figure 6.7. The tests are determined as in section 5.2.4. The digitization highest number of detection instances chart shows that on average the number of detections is slightly less than that achieved by the samples values and rate of charge methods. This is due to the reduction in resolution when the digitization process is applied. To achieve a higher resolution the number of digitization tests would need to be increased.

Table 6.1: Faults Introduced to the Circuit in Figure 6.2

Fourth-Order Low-Pass Filter Faults List	
Fault No.	Description of Fault Condition
F-1	C1: 1.6nF → 3.3nF
F-2	C2: 1.6nF → 0.82nF
F-3	C3: 1.6nF → 1.0nF
F-4	C3: 1.6nF → 10pF
F-5	C4: 1.6nF → 3.9nF
F-6	C1: 1.6nF → Open
F-7	R1: 6.98KΩ → 10KΩ
F-8	R2: 6.98KΩ → 3KΩ
F-9	R3: 39.2KΩ → 100KΩ
F-10	R4: 3.24KΩ → Short
F-11	R5: 6.19KΩ → 1.6KΩ
F-12	R6: 6.19KΩ → 10KΩ
F-13	R7: 39.2KΩ → Open
F-14	R7: 39.2KΩ → 10KΩ
F-15	R8: 29.4KΩ → 100KΩ
F-16	OpAmp-1: (+) & (-) Shorted with 250Ω
F-17	OpAmp-1: (+) & (Out) Shorted with 2.5KΩ
F-18	OpAmp2: (-) Shorted to GND with 1.0KΩ
F-19	OpAmp-2: (+) & (Out) Shorted with 10KΩ
F-20	OpAmp-2: (-) & (+) Shorted with 100Ω

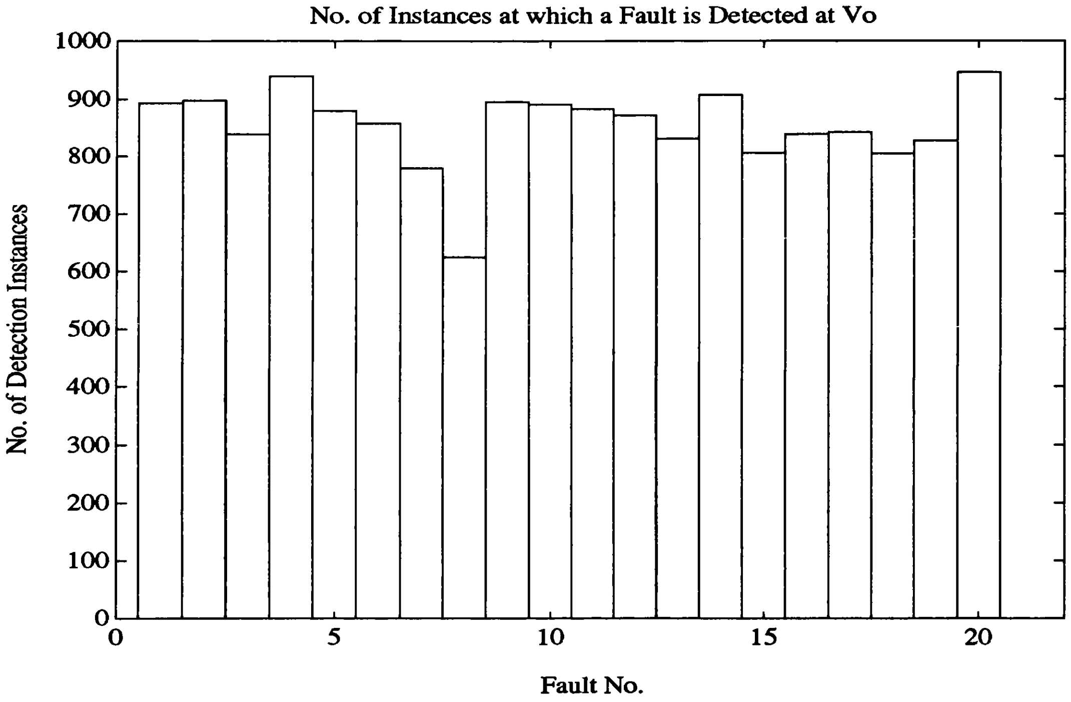


Figure 6.3: Detection Instances at (Vo) of the Fourth-Order LPF

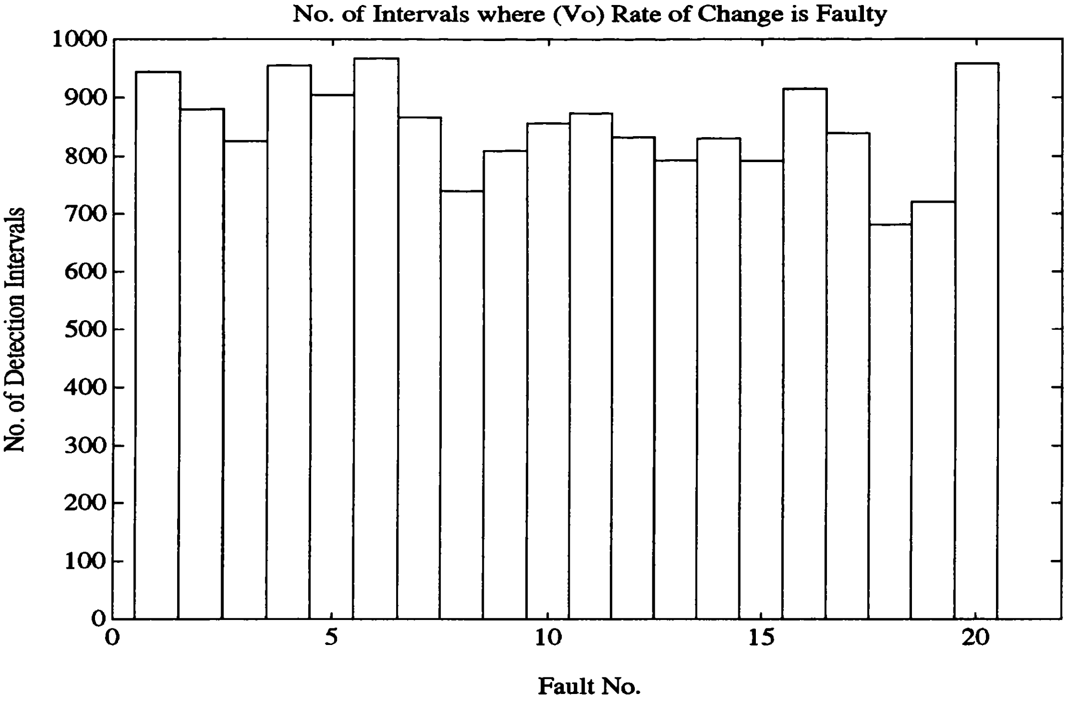
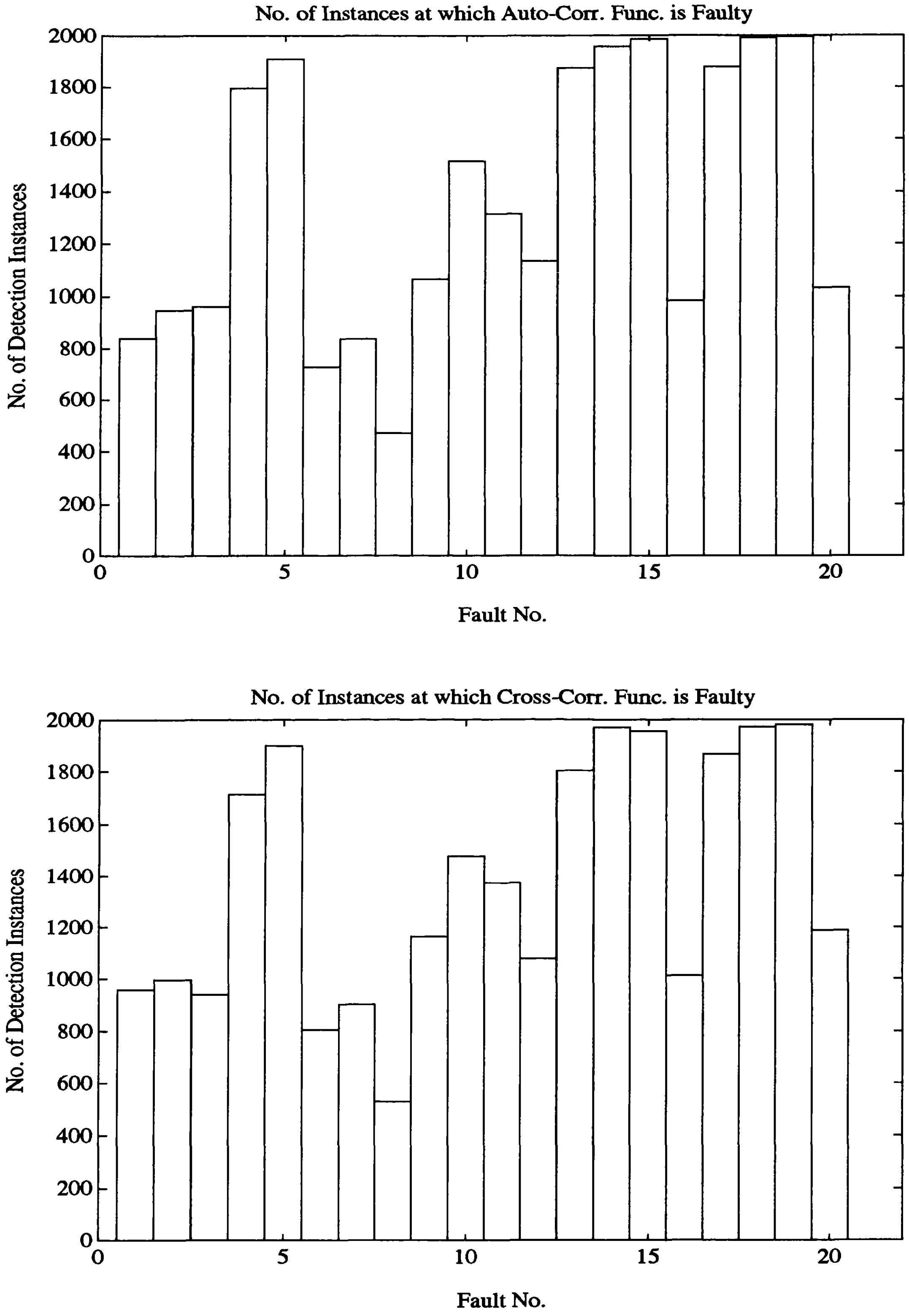


Figure 6.4: Detection Intervals at (Vo) of the Fourth-Order LPF



No. of Instances at which Auto-Corr. Func. is Faulty

No. of Instances at which Cross-Corr. Func. is Faulty

Figure 6.5: Detection Instances by Auto & Cross Corr. of Fourth-Order LPF

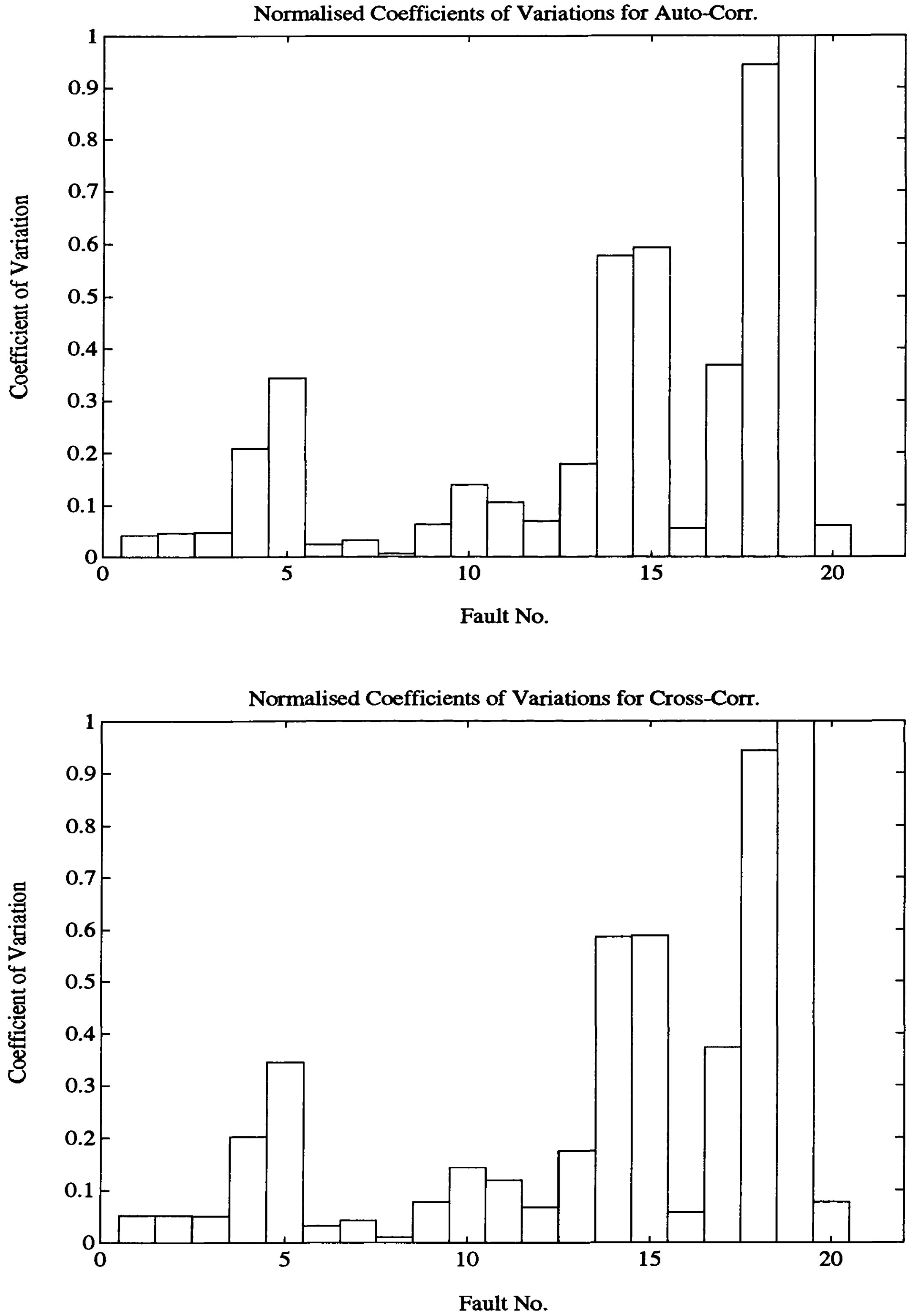
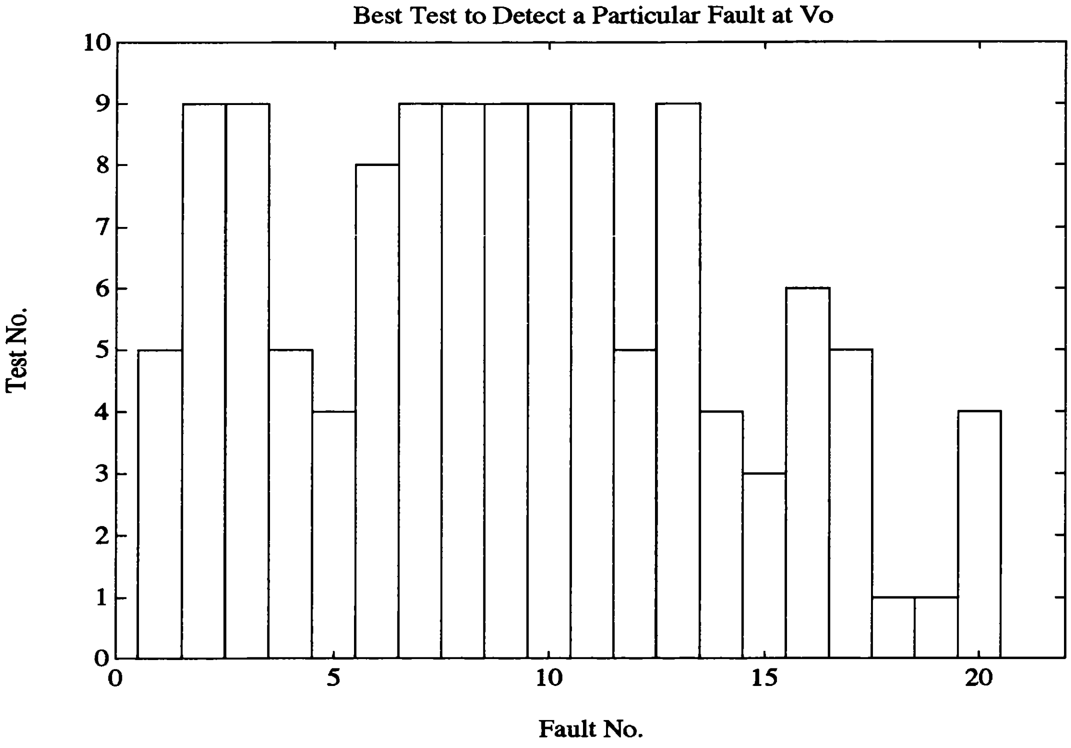


Figure 6.6: Normalised CVs of Auto & Cross Correlations of Fourth-Order LPF



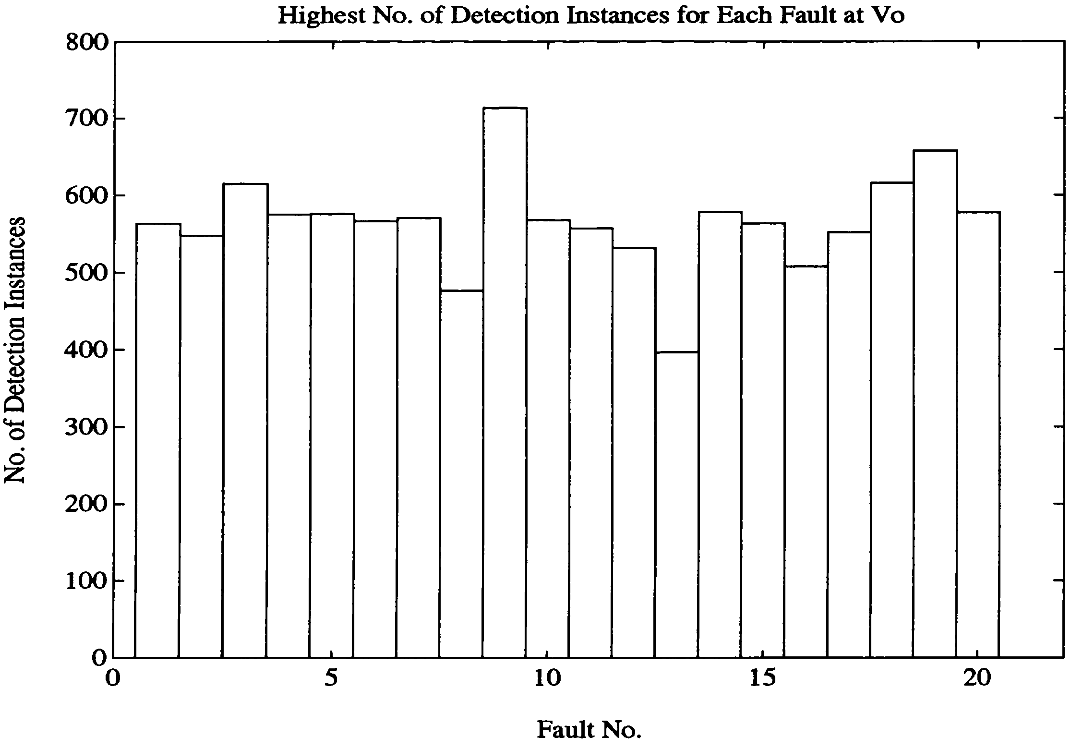


Figure 6.7: Best Test & Highest Detect. by Digitized (Vo) of Fourth-Order LPF

6.3 Mixed-Signal Circuit

The mixed-signal circuit shown in Figure 6.8 was implemented using discrete components and tested to demonstrate the mixed-signal case. The low-pass filters at the input and output are identical and each has a 3-dB bandwidth of 2KHz.

To test the mixed-signal circuit, a PRBS test signal with a 250μsec (i.e. 4KHz) bit interval and 63 bits long was applied. The circuit transient response (Vo) was sampled at a rate of 32KHz (i.e. 8 samples per bit interval).

The 20 fault conditions introduced to the circuit in Figure 6.8 are listed in Table 6.2. The faults include soft and catastrophic faults in the resistive and capacitive components, catastrophic faults associated with the terminals of the opamps and comparators, and stuck-at and bridging faults in the digital logic gates.

As in the previous section, the transient response data of the mixed-signal circuit was processed by the four methods of analysis to detect the presence of a fault. The four methods resulted in an equal fault-coverage of 100% (i.e. all faults were detected).

The number of detections for each fault as a result of applying the samples values and rate of change methods are illustrated in Figure 6.9 and Figure 6.10 respectively. The figures show that a high detection rate was achieved for all the faults.

The application of auto and cross correlation method of analysis resulted in the detection plots in Figure 6.11. The plots, as in the previous case, indicate a high degree of detection and that auto and cross correlation functions are almost equivalent in their detection capabilities. The near equivalence of auto and cross correlation in terms of fault detection is again demonstrated by the normalised coefficients of variations plots in Figure 6.12.

The response digitization method also achieved a high detection rate, despite its inherent lower resolution compared with the other methods of analysis. The bar charts of the best test and highest number of detection instances for each fault are depicted in Figure 6.13.

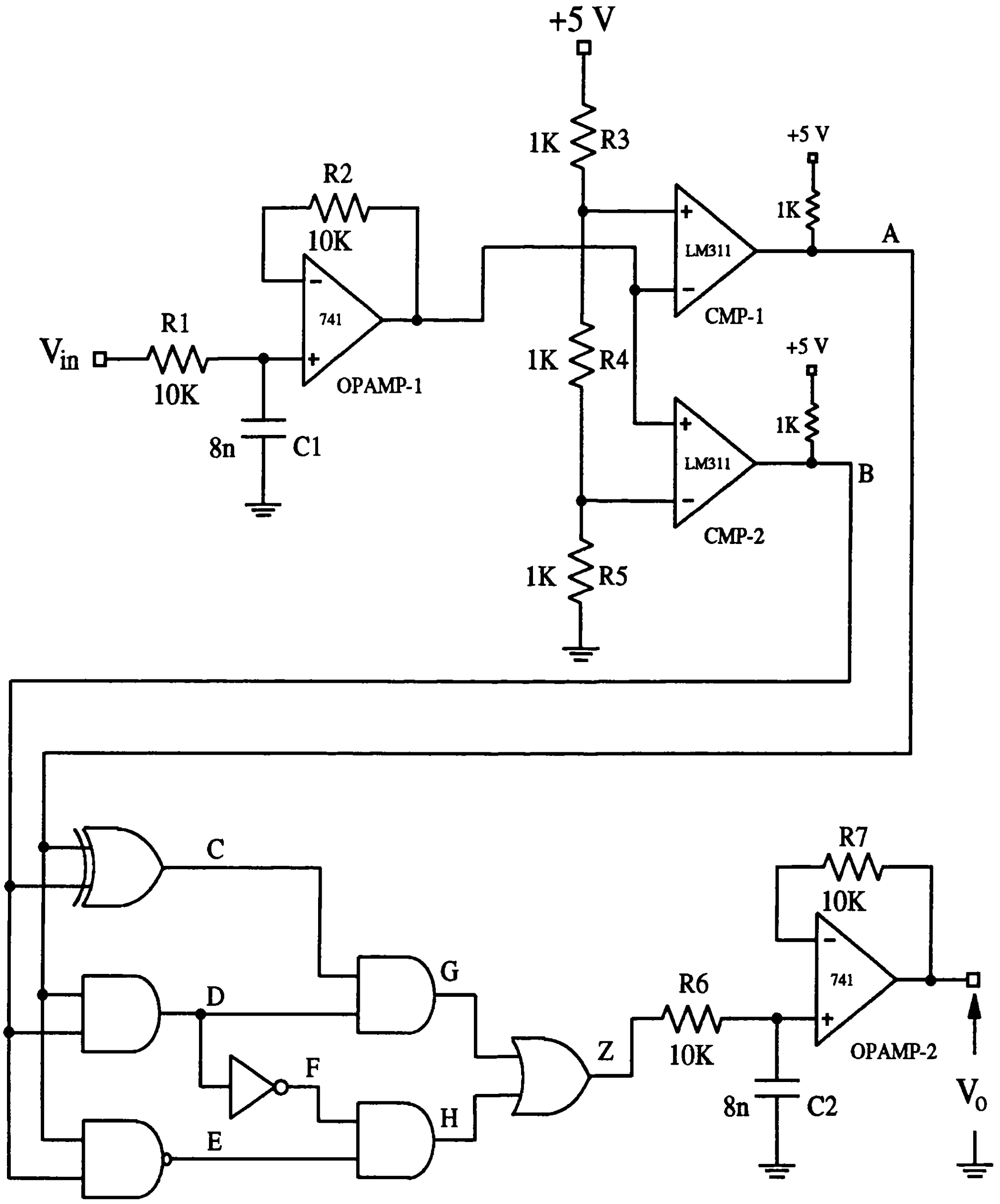


Figure 6.8: Mixed-Signal Circuit Tested Experimentally

Table 6.2: Faults Introduced to the Mixed-Signal Circuit

Mixed-Signal Circuit Faults List	
Fault No.	Description of Fault Condition
F-1	R1: 10KΩ → 20KΩ
F-2	C1: 8nF → 3.9nF
F-3	R2: 10KΩ → 100KΩ
F-4	R3: 2KΩ → Short
F-5	R4: 2KΩ → 6KΩ
F-6	R5: 1KΩ → 10KΩ
F-7	R6: 10KΩ → 1KΩ
F-8	C2: 8nF → 15nF
F-9	R7: 10KΩ → 1KΩ
F-10	OpAmp-1: (-) Shorted to GND by 250Ω
F-11	OpAmp-2: (-) & (+) Shorted by 150Ω
F-12	CMP-1: (+) Shorted to GND by 150Ω
F-13	CMP-2: (+) Floating
F-14	Line A s-a-1
F-15	Line C s-a-0
F-16	Line D s-a-1
F-17	Lines E s-a-1
F-18	Lines B & E Shorted
F-19	Line F s-a-1
F-20	Line G s-a-0

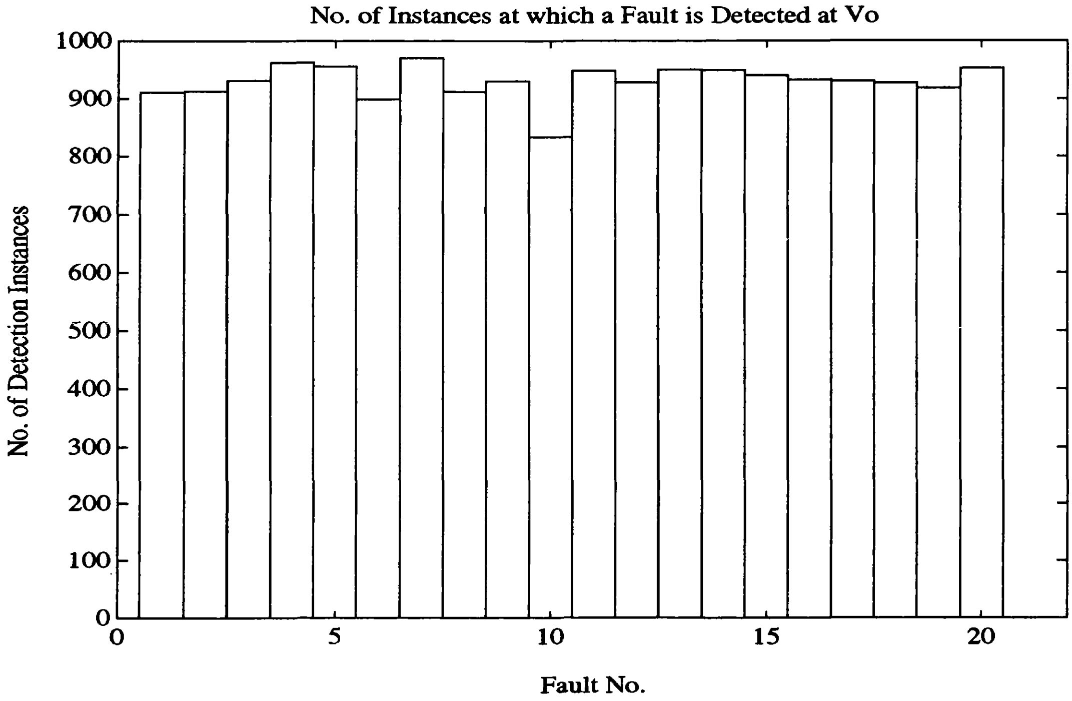


Figure 6.9: Detection Instances at (Vo) of the Mixed-Signal Circuit

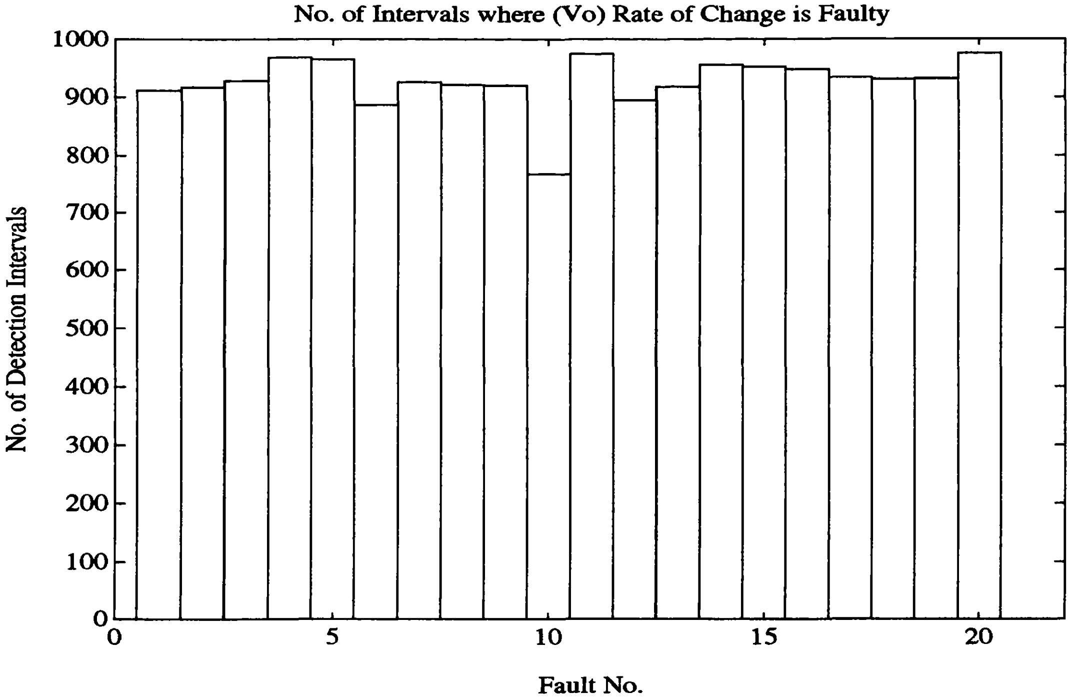


Figure 6.10: Detection Intervals at (Vo) of the Mixed-Signal Circuit

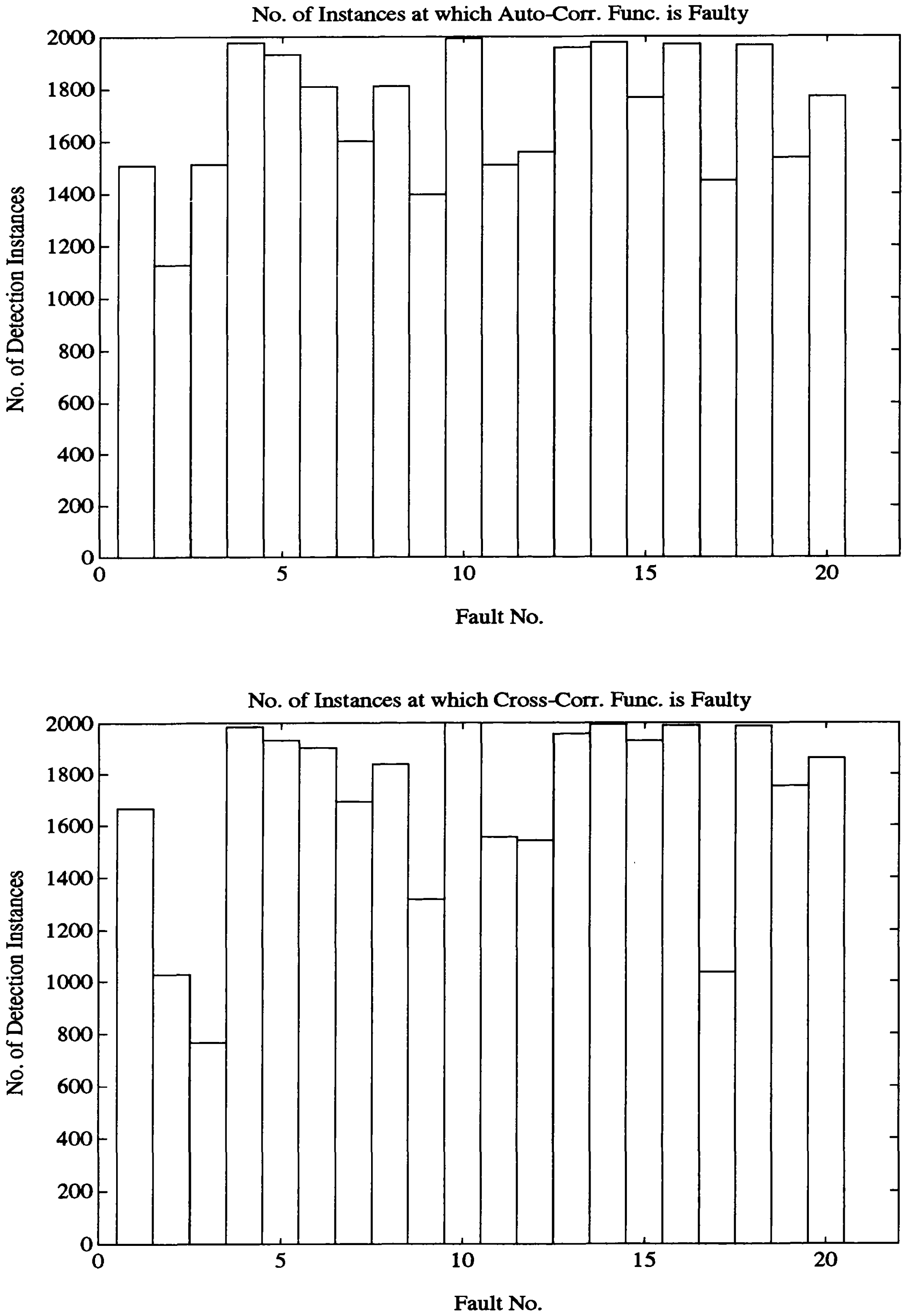
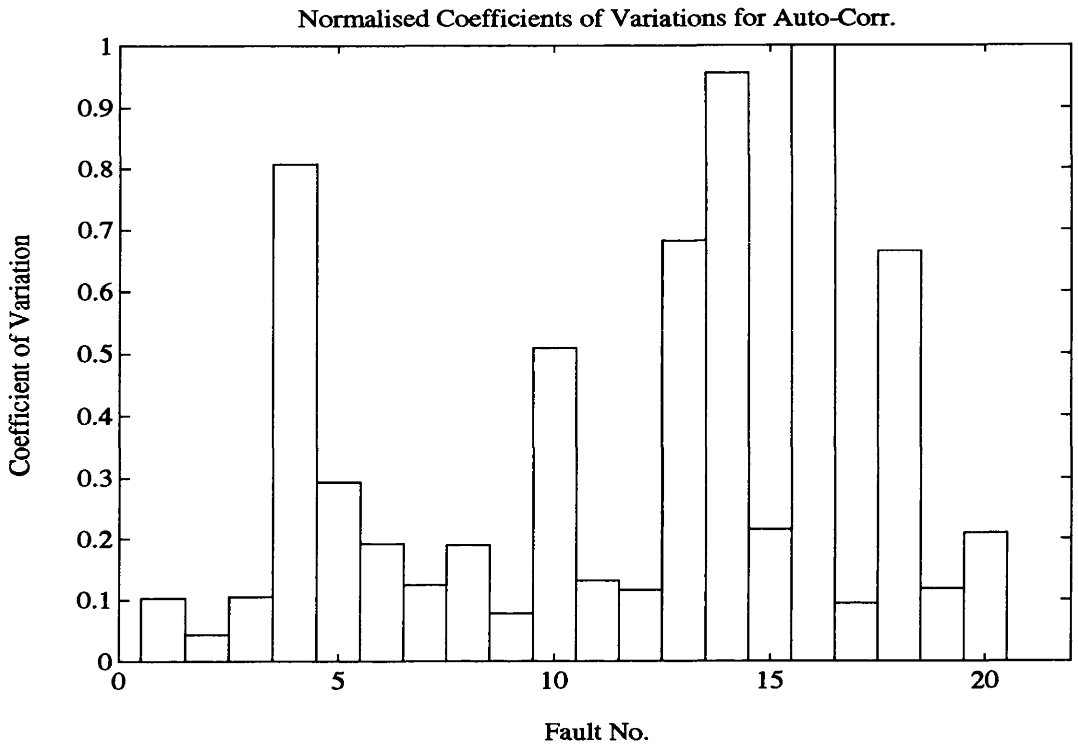


Figure 6.11: Detection Instances by Auto & Cross Corr. of Mixed-Signal Circuit



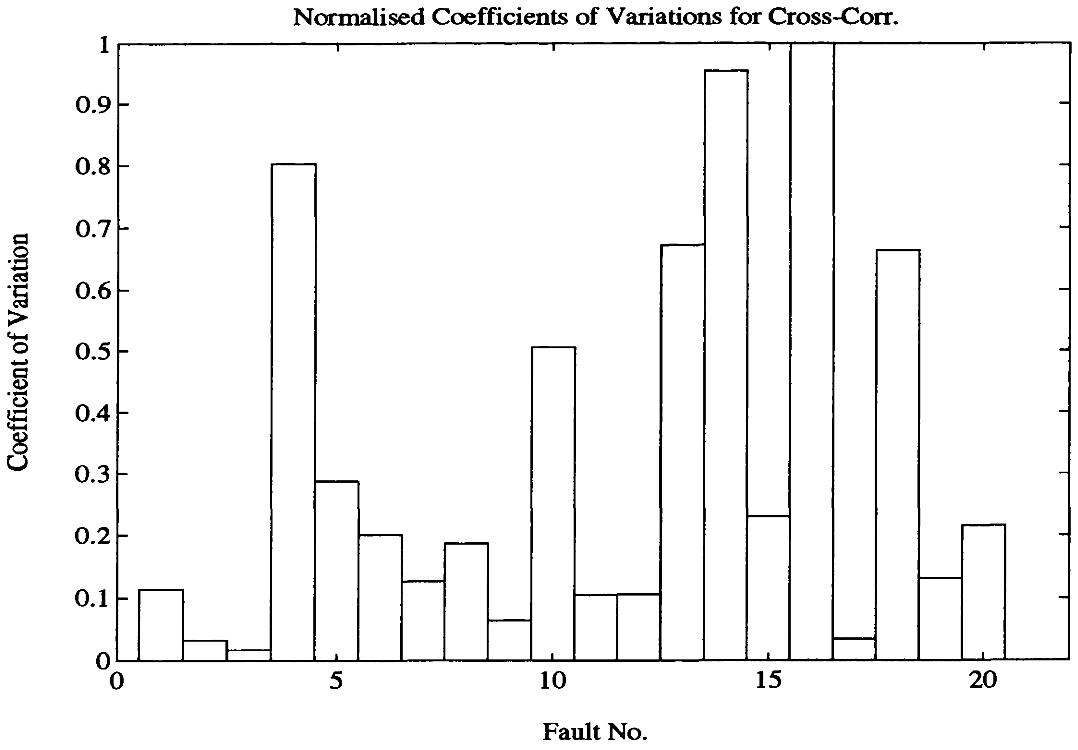


Figure 6.12: Normalised CVs of Auto & Cross Corr. of Mixed-Signal Circuit

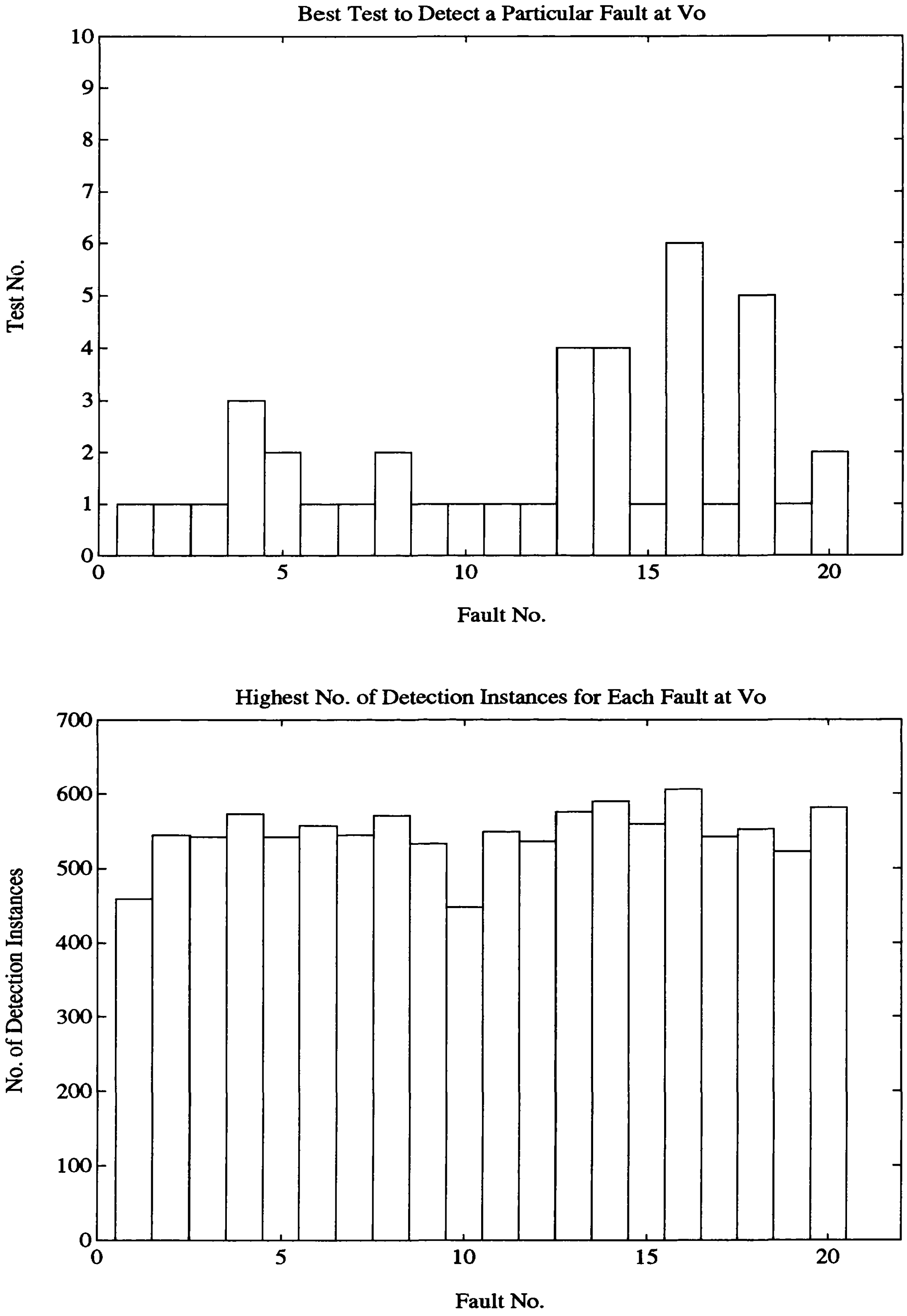


Figure 6.13: Best Test & Highest Detect. by Digitized (Vo) of Mixed-Signal Cct.

6.4 Effect of PRBS Bit Frequency

This section studies the effect of a PRBS test signal bit frequency (i.e. bit interval T) on the detectability of a fault.

To demonstrate the effect of T, the analogue circuit in Figure 6.2 was used. The fault-free circuit was injected with a PRBS test signal 63 bits long. The process of excitation and measurement of the transient response at the circuit output was repeated 10 times by only varying the PRBS bit frequency from 5KHz to 50KHz in steps of 5Khz. The sampling frequency was always 10 times that of the bit frequency in order not to affect the resolution of the measurements. The same test conditions and procedure were repeated for the same circuit, but this time fault F-1 (C1 changed from 1.6nF to 3.3nF) from Table 6.1 was introduced to the circuit.

For each test run the fault-free and faulty responses, at a particular PRBS bit frequency, were compared by the sample values method to determine the number of detection instances and the coefficients of variation. The plot of the normalised coefficients of variations for the same fault at the 10 PRBS bit frequencies is shown in Figure 6.14. The figure indicates that the best detectability is achieved at 25KHz with the 20KHz run very close to it. These results demonstrate that the value of T should be selected to place Fourier components of high amplitude in the sensitive regions of the CUT response.

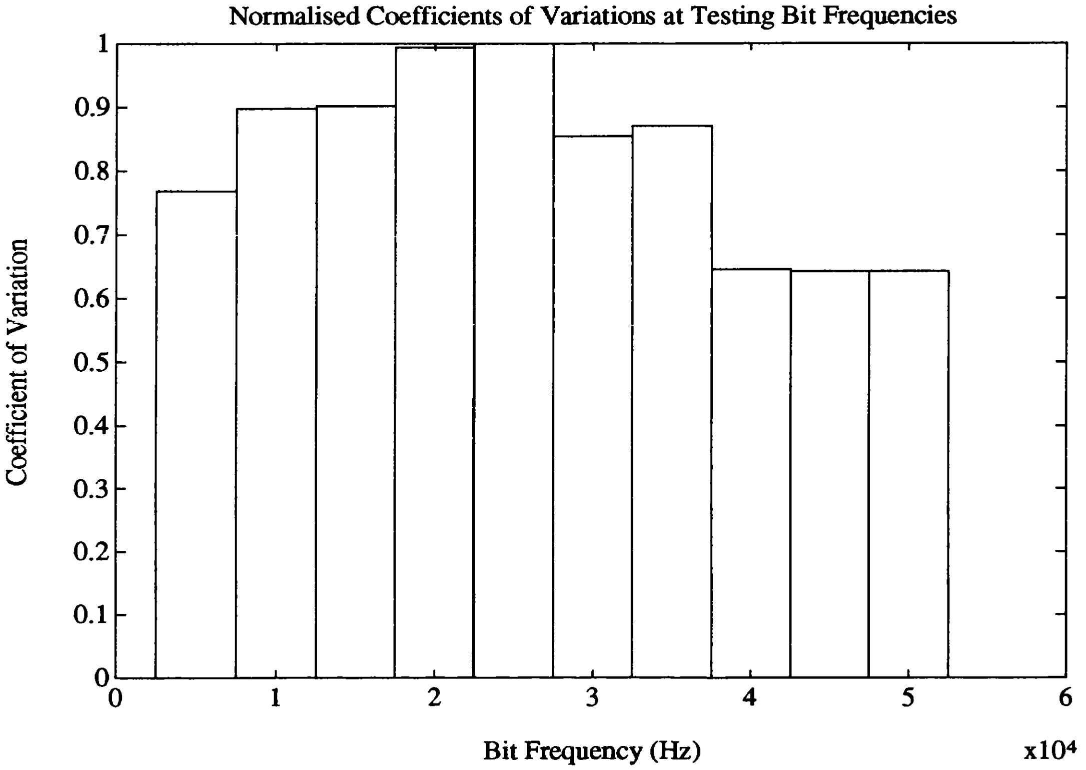


Figure 6.14: Effect of PRBS Bit Frequency on Fault Detectability

6.5 Effect of PRBS Length

In this section the PRBS bit frequency is fixed, while the length N of the PRBS is varied to demonstrate its effect on the detectability of a faults. The circuit in the previous section, Figure 6.2, with the same fault condition is used to study the effect of N.

The PRBS bit frequency was fixed at 20KHz, and the transient response of the fault-free circuit to PRBS test signals 15, 31, 63, 127 and 255 bits long was measured. The same procedure was repeated with fault F-1 (C1 changed from 1.6nF to 3.3nF) present in the CUT. The data of the fault-free and faulty responses were then analysed by the sample values method. The results of the analysis are summarised in Table 6.3. They indicate that the longer the PRBS the higher the number of detection instances and hence the better the detectability of a fault.

As indicated in Table 6.3, the sampling frequency had to be reduced as the length of the PRBS is increased. This was necessary to fit at least one complete PRBS period within the 1000 samples limit imposed by the experimental system hardware. This limitation resulted in less of an increase in the number of detection instances than expected as N was increased. However, the trend of improvement in detectability as N is increased is evident in Table 6.3. This confirms that the longer the PRBS the closer it gets to mimicking white noise characteristics and hence improves detectability.

Table 6.3: Effect of the PRBS Length (N) on Fault Detectability

Fourth-Order LPF with F-1 Present		
PRBS Length (N)	Sampling Frequency (KHz)	No. of Detection Instances
15	500	822
31	200	899
63	200	902
127	80	911
255	60	937

6.6 Summary

The experimental results of both the analogue and mixed-signal circuits tested achieved a high percentage of fault-coverage, in fact all the faults introduced were detected with a good degree of confidence. These results support the simulation results discussed in Chapter 5. The extensive testing results reported in [1] are also in agreement with those in this chapter.

6.7 References

[1] A. Bertin, The Testing of Mixed-Signal Integrated Circuits - Transient Response Analyser, Final Year Report, School of Electronic and Electrical Engineering, University of Bath, Bath, U.K., May 1992.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

This chapter summarizes the work covered by this thesis, discusses the results detailed in previous chapters and suggests directions for future research in the field of testing analogue and mixed-signal integrated circuits.

To illustrate the need for testing integrated circuits and the difficulties involved, in both fault modelling and test generation, a review of the mechanisms of failure in ICs was presented in Chapter 2. The mechanisms were classified, according to the source of their causes, into three categories: electrical stress, intrinsic and extrinsic. Electrical stress failures are caused by either electrical overstress or electrostatic discharge. Both mechanisms are a result of poor design or improper handling of the IC. Modes of failure that fall under the intrinsic category include gate oxide breakdown, ionic contamination, surface charge spreading, charge effects, piping and dislocations. The extrinsic mechanisms of failure are caused by manufacturing processes and operating environmental conditions which include packaging, metallization, bonding, die attachment, particulate contamination and radiation.

Chapter 3 presented an overview of the subject of ICs testing. The various stages of testing that an IC goes through during its life cycle were outlined. This was followed by a discussion of the stuck-at, stuck-open, stuck-on, bridging, layout driven and transistor based fault models. With the exception of the last two, the other models are primarily for digital circuits. The layout driven model applies to all ICs, but the process of deriving it is elaborate and computationally expensive. The transistor based model is simpler than the layout one and is also applicable to both analogue and digital ICs. Hence, it was used to evaluate all the testing strategies investigated in the thesis.

The test generation algorithms and the design-for-testability (DFT) techniques for digital ICs were also studied in Chapter 3. The test generation algorithms studied are: D-algorithm, Boolean difference and switch-level algorithms. The first two algorithms generate tests for stuck-at faults at the gate level, while the switch-level algorithm generates tests for both gate stuck-at faults, and transistor stuck-open and stuck-on faults. The D-algorithm is the one used most in practice due to its simplicity and computational efficiency. A number of prominent DFT techniques that would enhance the testability of digital circuits, particularly sequential ones, were described. The DFT techniques described fall into three categories: ad-hoc, structured and built-in-self-test (BIST).

Chapter 3 then outlined the factors that are causing analogue ICs designing and testing to be lagging behind their digital ICs counterpart. These factors were attributed to the unstructured nature of analogue circuits, the problem of tolerance in analogue circuits, lack of adequate and efficient models for simulation and test pattern generation, and the vast number of modes of failure in analogue circuits.

The testing techniques available for testing discrete analogue circuits were reviewed. The techniques were classified into: simulation-before-test, simulation-after-test with a single test vector, and simulation-after-test with multiple test vectors. Techniques that fall within the simulation-after-test with a single test vector category are generally preferred for testing discrete analogue circuits and systems. Finally, Chapter 3 reviewed the approaches reported in the literature and used in practice to test mixed-signal ICs. All the approaches basically partition the mixed-signal circuit-under-test (CUT) to separate analogue and digital blocks, then apply mode specific tests to each block. The disadvantages of partitioning which include waste of silicon area, long test time and high production cost were discussed.

Chapter 4 investigated in detail three existing testing techniques originally devised for discrete analogue circuits. The techniques are: dc fault dictionary, digital modelling and logical decomposition. The objectives were to assess the applicability of the techniques to testing analogue and mixed-signal ICs, their test points requirements, complexity and ability to implement on a digital tester.

The study of the dc fault dictionary approach showed that:

1- It is more suitable for production testing (i.e. go/no-go) than diagnostic testing, due to the low isolation that can be achieved with a limited number of accessible nodes. This is not a disadvantage as far as testing of ICs is concerned, because finding out whether a chip is faulty or not is more important than identifying a faulty transistor in the chip.

2- High fault-coverage was achieved for the comparator and op-amp circuits. When access to all the circuit nodes and a tolerance of 300mV were assumed, the fault-coverage was 69.8% for the comparator and 91.8% for the op-amp.

3- For both the comparator and op-amp circuits two nodes, one of which is the output node, would be sufficient to achieve a fault-coverage comparable to that when all nodes were accessed were identified. The process of identifying the nodes was systemized using a sensitivity factor, which gives an indication of the effect of faults introduced on each circuit node. A node with a high sensitivity factor is considered more likely to be a test node than one with a low sensitivity factor.

4- Although simulating all the faults in the dictionary is time consuming, especially if the circuit is a complex one, the process needs to be performed only once for a particular circuit. Therefore, this testing approach has good potential for testing analogue cells.

The digital-logic equivalent approach, which falls under digital modelling, is the ideal technique as far as testing an analogue or a mixed-signal IC using a conventional digital ICs tester is concerned. However, the approach requires good knowledge of the structure and operation of the analogue block, and the nature of faults that are likely to occur in that block. The translation of that knowledge to an equivalent digital circuit that behaves like the corresponding analogue one is a difficult task, and it cannot be generalised because the design of the analogue blocks tends to be of an individual nature (i.e. depending on the designer).

An alternative to the digital-logic equivalent approach is the functional K-map digital modelling approach. In this approach the behaviour of the analogue block is mapped to a digital function by simply accessing its external nodes. The main points that resulted from assessment of the approach are summarized below:

1- The major advantage of this testing method over that of the fault dictionary is that only the input and output nodes of the CUT need to be accessed. These nodes do not increase the pin count because they are readily accessible.

2- The computational effort required by this method is much less than that required by the fault dictionary.

3- From the analysis of the comparator and op-amp circuits results, the K-map testing method generated tests which achieved the highest possible fault-coverage for dc testing. The degree of confidence in the tests is either the optimum or close to the optimum.

4- An important factor in this testing approach is the determination of the input voltages (i.e. values for test vectors) and the threshold voltage value. Changing the test and threshold voltage may result in different digital functions and subsequent variation in fault-coverage.

Both the dc fault dictionary and functional K-map approaches can be implemented on a conventional digital tester. However, both approaches are restricted to dc testing. Therefore, no information can be extracted about the dynamic behaviour of an analogue CUT and reactive components cannot be tested.

The final approach assessed in Chapter 4 is the logical decomposition approach. The approach is a general one which applies to both linear and non-linear networks. This was demonstrated by utilizing it to test an active filter and a video amplifier networks. The method places no restrictions on the test signals that can be used, hence it overcomes the limitations of the dc based methods. The ability to locate a fault in this technique is very much dependent on the degree of network decomposition and hence the number of accessible nodes.

The logical decomposition approach is efficient because the diagnosis is performed at the subnetwork level rather than the component level. This testing approach produces reliable results if the devices tested allow reasonable amounts of current flow (e.g. bipolar devices). Hence, it was shown that the strategy is not applicable to CMOS ICs due to the small amount of current supported in such devices.

A unified strategy, called the time-domain technique, for testing analogue and mixed-signal ICs was presented in Chapter 5. The fundamental concept behind the technique is the excitation of the CUT with an appropriate pseudo-random binary sequence (PRBS) test signal and the extraction of the resulting transient response at the CUT external nodes. The technique has the following advantages over the other techniques investigated in this thesis and those reported in the literature:

1- The technique can test a mixed-signal CUT as one complete entity. Hence, eliminating the requirement for partitioning the CUT to separate analogue and digital modules.

2- The PRBS test signals have very well defined properties and consist of pulses with constant amplitude. Such pulses can be readily generated by a digital tester.

3- Due to the compatibility of PRBS test signals with a digital tester, the time-domain technique can be implemented on a conventional digital tester. Hence, resulting in a reduction in the cost and time of testing analogue and mixed-signal ICs.

4- The transient response data, extracted at the CUT external nodes, contain a wealth of information about the CUT. This data can be processed in a number of ways to estimate a variety of device parameters, such as its impulse response.

5- Being time-domain the technique enables the dynamic testing of the CUT. Hence, it overcomes the limitations associated with the static testing approaches, such as dc fault dictionary and digital modelling.

6- The technique can handle catastrophic, parametric and stuck-at faults.

Three CMOS circuit examples were simulated in Chapter 5, they included a first-order low-pass filter, fourth-order low-pass filter, and mixed-signal circuits. For each circuit, both the transient voltage at the output node/s and the transient supply current (Idd) were measured.

Four methods were devised to analyse the voltage and current transient responses. The objectives were to establish which type of measurement (i.e. voltage or current) is best at detecting a particular fault, which method of analysis achieves the highest fault-coverage and which one is computationally most efficient. The methods of analysis utilized are: samples values, rate of change, auto and cross correlation and response digitization.

The results of processing the data for the three circuit examples indicate that the time-domain technique achieves a high percentage of fault-coverage without the need for partitioning. The four methods of data analysis are comparable in terms of fault-coverage achieved. Of the four analysis methods, the response digitization method is the most efficient in terms of computation, because it eliminates the need for floating point computation once the digitization is performed. The method can also be readily implemented on a digital tester, hence resulting in saving in both the time and cost of testing analogue and mixed-signal ICs.

In general the rate of change method results in a higher number of detections for each fault than the other methods. This indicates that the method is more sensitive to faults than the others. However, closer inspection of the results showed that the extra detections do not lead to higher confidence in the detection. In the case of the first-order low-pass filter circuit, when the rate of change method resulted in the detection of four extra faults, the confidence in the detection of those fault was low.

Auto and cross correlation detection instances and normalised coefficients of variations bar charts indicate that both correlation functions are equivalent in their ability to detect faults. Both correlation functions resulted in a slightly lower percentage of fault-coverage than the samples values and rate of change methods. The reason for this is that correlation is an averaging process, and hence less sensitive than the other two methods.

The results of the application of the response digitization method indicate that this method leads to a reduced fault-coverage compared with the other methods. This is due to a reduction in the data resolution as a result of digitization, which could be easily overcome by simply increasing the number of tests.

The bar charts of the number of detection instances for the three examples show that on average the current measurement (Idd) achieves higher number of detection instances than the voltage measurement. This indicates that Idd is more sensitive to faults than voltage. However, the plots of the coefficients of variations (CVs) indicate that voltage and current measurements are complementary. This means that some faults are best detected by voltage while others are best detected by current.

To investigate the type of faults (i.e. short, open, soft or stuck-at) best detected by voltage or current, the results of the CVs plots for all the examples simulated were mapped to the list of faults introduced. The results of the mapping show that no clear trend can be detected, which means no set of faults can be associated with a particular measurement.

In Chapter 6 the prototype experimental system implemented to capture the transient response of a CUT was described, and the experimental results of testing an analogue and a mixed-signal circuits were presented. The experimental results for both circuits showed that time-domain testing achieved 100% fault-coverage. The transient response voltage data for both circuits were processed with the same methods of analysis applied to the simulation data in Chapter 5. The results of processing the experimental data were in agreement with the simulation results.

The effect of the PRBS test signal bit interval and length of the PRBS on the detectability of a fault were investigated experimentally in Chapter 6. The results of the analysis showed that a fault is more detectable when the bit interval is selected such that the major Fourier components of the test signal fall in the critical region of the CUT response. As for the length of the PRBS, the results indicated that the detectability of a fault is directly proportional to the PRBS length (i.e. the longer the PRBS the higher the number of detection instances). The reason for this is that a long PRBS sequence has characteristics close to those of white noise, and hence injects a wide range of frequencies into the CUT.

The experimental system used is a first cut design with some limitations. Therefore, the use of a dedicated data acquisition system with capability to capture both voltage and current would allow more in depth evaluation of the time-domain testing techniques and associated methods of data analysis.

The simulation and experimental results in chapters 5 and 6 respectively, demonstrate that the time-domain technique is an effective strategy for testing analogue and mixed-signal ICs in a unified fashion. The technique and the response digitization analysis method can be implemented on a digital tester. Therefore, they present an attractive solution to the problem of testing analogue and mixed-signal ICs on a digital tester.

The time-domain technique and data analysis methods results presented in Chapters 5 and 6 are all for go/no-go testing. However, diagnosis to determine which fault is likely to be present can be achieved by building a dictionary of signatures.

The signature of the CUT is then compared with the entries of the dictionary to diagnose a fault. An efficient implementation of such a dictionary would store the digitized signature rather than the original one in the dictionary. Thus it will result in a reduction in the dictionary memory and storage requirements, and an increase in the speed of comparison due to the elimination of floating point calculations.

The subject of testing analogue and mixed-signal ICs has only attracted the attention of researchers fairly recently. A great deal of work still needs to be done to make the testing of such ICs as efficient as that of digital ICs. Suggestions for future research work in this field include the following:

1- Efficient simulation of analogue circuits by simulating the behaviour of an analogue module.

2- Development of a true mixed-signal simulator that is capable of simulating different parts of a circuit at different levels (e.g. component or behaviour).

3- Study of the dominant faults in analogue ICs.

4- Derivation of an adequate and efficient fault model for analogue ICs, preferably similar in simplicity to the stuck-at model in digital ICs.

5- Design of a fast current sensor that can be implemented on chip to monitor variations in the voltage supply current.

6- Development of DFT techniques for analogue circuit, that would be compatible with digital DFT structures and occupy an acceptable amount silicon area.

7- Development of the IEEE 1149.4 mixed-signal standard test bus. This bus should be compatible with the ANSI/IEEE 1149.1 standard for digital circuits.

LIST OF PUBLICATIONS

The research work in this thesis resulted in the following publications:

1- M.A. Al-Qutayri and P.R. Shepherd, "Testing Approaches for Mixed-Mode (Analogue/Digital) Integrated Circuits", Proceedings of the Silicon Design Conference, pp. 37-45, London, November 1989.

2- M.A. Al-Qutayri and P.R. Shepherd, "On the Testing of Mixed-Mode Integrated Circuits", Journal of Semicustom ICs, Vol. 7, No. 4, pp. 32-39, June 1990.

3- M.A. Al-Qutayri, P.S. Evans and P.R. Shepherd, "Testing Mixed Analogue/Digital Circuitry Using Transient Response Techniques", 7th European Design for Testability Workshop, Segovia, Spain, June 1990.

4- P.S. Evans, M.A. Al-Qutayri and P.R. Shepherd, "On the Development of Transient Testing Techniques for Mixed-Mode ICs", Journal of Semicustom ICs, Vol. 8, No. 2, pp. 34-38, September 1990.

5- P.R. Shepherd, M.A. Al-Qutayri and P.S. Evans, "Testing Mixed-Signal Integrated Circuits", IEE Colloquium on Design and Test of Mixed Analogue/Digital ICs, Savoy Place, London, 15 November 1990.

6- P.S. Evans, M.A. Al-Qutayri and P.R. Shepherd, "A Novel Technique for Testing Mixed-Signal ICs", Proceedings of 2nd European Test Conference, Munich, Germany, pp. 310-306, April 1991.

7- P.R. Shepherd and M.A. Al-Qutayri, "A Time-Domain Strategy for Testing Mixed-Signal ICs", 6th UK Design Automation Workshop, Hilton Hotel, Bath, May 1991.

8- M.A. Al-Qutayri and P.R. Shepherd, "PRBS Testing of Analogue Circuits", IEE Colloquium on Testing Mixed-Signal Circuits, Savoy PLace, London, May 1992.

9- M.A. Al-Qutayri and P.R. Shepherd, "Go/No-Go Testing of Analogue Macros", IEE Proceedings on Circuits, Devices and Systems -Part G, Vol. 139, No. 4, pp. 534-540, August 1992.

APPENDICES

APPENDIX - 1

```
PROGRAM opamptesting ( opampvoldata , opampstore , OUTPUT ) ;

{ This program implements the DC FAULT DICTIONARY STRATEGY to tests the
  CMOS Operational Amplifier Circuit. The program for the CMOS Comparator is
  very similar to this one. }
{ It manipulates the VOLTAGES calculated at all circuit nodes. }
{ The number of faults simulated is 70.                        }

CONST

    min  =  1 ;
    max  =  70 ;          (* The number of faults in the dictionary *)
    testpoints  =  14 ;  (* The number of selected test points *)
    testone  =  1 ;
    testtwo  =  2 ;
    tolerance  =  300.0E-3 ;

TYPE

    tests  =  testone .. testtwo ;
    faults  =  min .. max ;
    numofnodes  =  min .. testpoints ;
    numberofsets  =  0 .. max ;
    voltagevalue  =  REAL ;
    test  = RECORD
            faultfreevoltage  :  voltagevalue ;
            faultyvoltage  :  ARRAY [ faults ] OF voltagevalue
            END ;
    nodeparameters  =  RECORD
                   nodenumber  :  numofnodes ;
                   firsttest  :  test ;
                   secondtest  :  test ;
                   END ;
    excitation  =  ARRAY [ numofnodes ] OF test ;
    detectfault  =  A  RRAY [ faults ] OF BOOLEAN ;
    setrecord  =  RECORD
              low  :  voltagevalue ;
              nominal  :  voltagevalue ;
               high  :  voltagevalue ;
               setno  :  numberofsets ;
               END ;
    storesets  =  ARRAY [ numofnodes , faults ] OF setrecord ;
    faultstatus  =  RECORD
                   nodeno  :  numofnodes ;
```

```
                    faultno  :  faults ;
                    setstatus  :  setrecord ;
                    fpset :  faults ;
                    testno  :  tests ;
                    END ;
    faultpointer  =  faultstate ;
    faultstate  =  RECORD
                    faultst  :  faultstatus ;
                    next  :  faultpointer
                    END ;

VAR

    opampstore  :  TEXT ;
    opampvoldata  :  FILE OF nodeparameters ;
    nodetest  :  nodeparameters ;
    stimulione , stimulitwo  :  excitation ;
    detectone , detecttwo  :  detectfault ;
    setone , settwo  :  storesets ;
    detectable  :  BOOLEAN ;
    nodenumb  :  numofnodes ;
    nofaults  :  faults ;
    listone ,listtwo ,complistone ,complisttwo ,combinedlist ,finallist : faultpointer ;

(* ---------------------------------------------------------------- *)

PROCEDURE initializestnodes ;

BEGIN

    REWRITE ( opampstore )

END ;  (* initializestnodes *)

(* ---------------------------------------------------------------- *)

PROCEDURE echonodes ;

VAR

    count  :  0 .. testpoints ;

BEGIN

    RESET ( opampvoldata ) ;
    count  :=  0 ;
    WHILE NOT EOF ( opampvoldata ) DO
        BEGIN
        READ ( opampvoldata , nodetest ) ;
        WITH nodetest DO
          BEGIN
            count := count + 1 ;
            stimulione [ count ]  :=  firsttest ;
            stimulitwo [ count ]  :=  secondtest
          END
      END ;
    WRITELN ('The data in FILE opampvoldata.dat have been echoed')
END ;  (* echonodes *)

(* ------------------------------------------------------------ *)

PROCEDURE  checkdatafile ;

VAR

    noddy  :  numofnodes ;
    fft  :  faults ;

BEGIN

    FOR  noddy  :=  min TO testpoints DO

      BEGIN
        WRITELN ;
        WRITELN ('For node NO', noddy ) ;
        WRITELN ( opampstore , 'For node No.   ', noddy ) ;
        WRITELN ('Correct voltage = ',stimulione [noddy].faultfreevoltage,
        ' ',stimulitwo [noddy].faultfreevoltage ) ;
        WRITELN (opampstore ,'Correct voltage is = ' , stimulione [noddy],
        faultfreevoltage, '  ', stimulitwo [noddy].faultfreevoltage ) ;

        FOR  fft  :=  min TO max DO

          BEGIN

            WRITELN ( ' FOR fault No.    '. fft) ;
            WRITELN ( opampstore, 'For fault No. ' , fft ) ;
            WRITELN ('faulty voltage = ',stimulione [noddy].faultyvoltage[fft] ,'
            ',stimulitwo [noddy].faultyvoltage [fft] );
            WRITELN  (  opampstore,  'faulty  volts  =    ',  stimulione
            [noddy].faultyvoltage [fft], ' ',stimulitwo [noddy].fault)
          END
      END
END ;  (* checkdatafile *)

(* ------------------------------------------------------------ *)
```

```
PROCEDURE  detector  ( VAR stimulous  :  excitation ;
                 VAR faultdetector  :  detectfault ) ;

VAR

    deviation , sumsqudeviation  :  voltagevalue ;
    count  :  0 .. testpoints ;

BEGIN

    WRITELN  (’Fault Number ’,’     ’,’Detection Status ’) ;
    WRITELN (opampstore ,’Fault Number’,’     ’,’Detection Status’);
    count  :=  TRUE ;
    FOR nodenumb  :=  min TO testpoints DO
        BEGIN
            count  :=  count + 1 ;
            deviation    :=    stimulous  [  nodenumb  ].faultfreevoltage  -  stimulous
            [nodenumb].faultyvoltage
            sumsqudeviation  :=  sumsqudeviation + SQR (deviation)
        END ;
        IF sumsqudeviation > ( SQR ( tolerance ) * count ) THEN
           Faultdetector [ nofaults ]  := detectable
        ELSE
           Faultdetector [ nofaults ]  := NOT ( detectable ) ;
           WRITELN (opampstore,’     ’,nofaults,faultdetector [ nofaults ]) ;
           WRITELN (nofaults,’     ’,faultdetector [ nofaults ] )
    END ;
END ; (* detecotr *)

(*--------------------------------------------------------------*)

PROCEDURE  isolator ( VAR  stimulant : excitation ) ;

TYPE

    isolate  :  isolatestatus ;
    lowerbound , upperbound , faultyvalue  :  voltagevalue ;
    noddy  :  numofnodes ;
    fft  :  faults ;

BEGIN

    FOR noddy  :=  min TO testpoints DO
      BEGIN
        WRITELN (’For node No.    ’,noddy ) ;
        WRITELN (opampstore,’For node No.    ’, noddy ) ;
        lowerbound  :=  stimulant[noddy].faultfreevoltage - tolerance ;
        upperbound  :=  stimulant[noddy].faultfreevoltage + tolerance ;
        FOR  fft  :=  min TO max DO
          BEGIN
            faultyvalue  :=  stimulant[noddy].faultyvoltage[fft] ;
            IF (faultyvalue >= lowerbound) AND (faultyvalue<=upperbound) THEN
                isolate  :=  notisolatable
            ELSE
                isolate  :=  isolatable ;
            WRITELN ('Fault No. = ',fft,' is ',isolate ) ;
            WRITELN (opampstore,'Fault No. = ',fft,' is ',isolate)
          END
        END
END ;  (* isolation *)

(*----------------------------------------------------------------*)

PROCEDURE  ambiguitysets ( VAR measurements :  excitation ;
                    VAR stimulisets  :  storesets ) ;

CONST

    setzero = 0 ;

TYPE

    ambigset = ARRAY [ min .. max ] OF setrecord ;

VAR

    sets  :  ambigset ;
    noddy  :  numbofnodes ;
    terminate  :  BOOLEAN ;
    setinf , setrec  :  setrecord ;
    termpvalue , lowtemp , hightemp  :  voltagevalue ;
    count , setcount, maxset , tempsetno , fft  :  numberofsets  ;

BEGIN

    FOR noddy  :=  min TO testpoints DO
      BEGIN
        tempvalue := measurements [ noddy ].faultfreevoltage ;
        lowtemp := tempvalue - tolerance ;
        hightemp := tempvalue + tolerance ;

        setcount := 0 ;

        FOR  fft := min TO max DO
          BEGIN
            WITH  setinf  DO
```

```
               BEGIN
                 nominal := measurements[noddy].faultyvoltage[fft] ;
                 low := nominal - tolerance ;
                 high := nominal + tolerance ;
                 IF (nominal >= lowtemp) AND (nominal <= hightemp) THEN
                   setno := setzero
                 ELSE
                   BEGIN
                     setcount := setcount + 1 ;
                     setno := setcount ;
                   END ;
                 WRITELN (’For fault No. ’,fft,’ ’,’nominal value = ’,nominal);
                 WRITELN (opampstore,’For fault No. ’,fft,’ ’,’nominal value=’,
                 nominal) ;
               END ;
                sets [fft] := setinf
          END ;

FOR fft := max DOWNTO min DO
    BEGIN
      setinf := sets [fft] ;
      IF setinf.setno <> setzero THEN
      BEGIN
        count := setzero ;
        terminate := FALSE ;
        WHILE NOT ( terminate ) DO
          BEGIN
             count := count + 1 ;
             setrec := sets [ count ] ;
             IF ( setrec.setno <> setzero ) THEN
             IF ( setinf.nominal >= setrec.low ) AND
               (setinf.nominal <= setrec.high ) THEN
             BEGIN
                 setinf := setrec ;
                 sets [fft ] := setinf ;
                 terminate := TRUE ;
             END (*IF*)
          END (*WHILE*)
      END (*IF*)
    END ; (*FOR DOWNTO*)

maxset := setzero ;
FOR fft := min TO max DO
  BEGIN
    setinf := sets [ fft ] ;
    IF ( setinf.setno <> setzero ) THEN
    BEGIN
      IF ( setinf.setno > maxset ) AND
```

```
        ( setinf.setno = ( maxset + 1 ) ) THEN
        maxset := setinf.setno
      ELSE
        IF ( setinf.setno > ( maxset + 1 ) ) THEN
          BEGIN
            maxset := maxset + 1 ;
            tempsetno := setinf.setno ;
            count := fft ;
          WHILE ( count <= max ) DO
            BEGIN
              setinf := sets [ count ] ;
              IF ( setinf.setno = tempsetno ) THEN
              BEGIN
                setinf.setno := maxset ;
                sets [ counts ] := setinf ;
              END ; (*IF*)
              count := count + 1 ;
             END ; (*WHILE*)
          END ;
    END ;
  END ; (*FOR*)

FOR fft := min TO max DO
   BEGIN
    stimulisets [ noddy , fft ] := sets [ fft ]  ;
    WRITELN ('Fault No. ',fft,' ','Set No.',stimulisets [noddy,fft].setno) ;
    WRITELN (opampstore,'Fault No. ',' ',fft,' ','Set No. ',' ',stimulisets
    [noddy,fft].setno) ;
   END
  END (* FOR noddy *)
END ; (* ambiguitysets *)

(*------------------------------------------------------------------*)

PROCEDURE constructlist (testset:tests ; seter:storesets ; VAR list : Faultpointer) ;

CONST
    setzero = 0 ;
VAR
    noddy : numofnodes ;
    listpointer : faultpointer ;
    tempset , shareset : setrecord ;
    count , counter , faultcounter , fft : INTEGER ;

BEGIN

  NEW ( list ) ;
  list ^.next := NIL ;
  listpointer := list ;
  faultcounter := 1 ;
  WHILE faultcounter <= max DO
    BEGIN
    FOR noddy := min TO testpoints DO
      BEGIN
        FOR fft := min TO max DO
          BEGIN
            tempset := seter [ noddy , fft ] ;
            count := 0 ;
            IF tempset.setno <> setzero THEN
             BEGIN
              shareset := seter [ noddy , counter ] ;
              IF shareset.setno = tempset.setno THEN
                count := count + 1 ;
              END ; (* FOR counter *)
              IF count = faultcounter THEN
              BEGIN
                WITH listpointer ^.faultst DO
                 BEGIN
                  nodeno := noddy ;
                  faultno := fft ;
                  setstatus := tempset ;
                  fpset := count ;
                   testno := testset ;
                 END ; (*WITH*)
              NEW ( listpointer ^.next ) ;
              listpointer := listpointer ^.next ;
              listpointer ^.next := NIL ;
            END ; (* IF count *)
          END ; (* IF tempset *)
        END ; (* FOR fft *)
      END ; (* FOR noddy *)
    faultcounter := faultcounter + 1 ;
   END ; (* WHILE faultcounter *)
END ; (* constructlist *)

(*-------------------------------------------------------------------*)

PROCEDURE compresslist (listno : faultpointer ; VAR complist : faultpointer) ;

VAR
    setslist , temp , window : faultpointer ;
    unique : BOOLEAN ;

BEGIN

NEW (setslist ) ;
```

```
setslist ^.faultst := listno^.faultst ;
setslist ^.next := NIL ;
complist := setslist ;
temp := listno ;
WHILE ( temp <> NIL ) DO
  BEGIN
    window := setslist ;
    unique := TRUE ;
    WHILE ((window <> NIL) AND (unique = TRUE)) DO
      BEGIN
        IF (temp^.faultst.faultno = window^.faultst.faultno) THEN
          BEGIN
            unique := FALSE ;
            IF (temp^.faultst.fpset) < (window^.faultst.fpset) THEN
             window^.faultst := temp^.faultst ;
          END
        ELSE
          unique := TRUE ;
          window := window ^.next ;
        END ; (* WHILE *)
      IF unique = TRUE THEN
        BEGIN
          NEW ( complist ^.next ) ;
          complist := complist ^.next ;
          complist^.faultst := temp^.faultst ;
          complist^.next := NIL ;
        END ; (* IF *)
      temp := temp^.next ;
    END ; (* WHILE *)
  complist := setslist ;
END ; (* compresslist *)

(*----------------------------------------------------------------*)

PROCEDURE joinsetlist ( listerone , listertwo : faultpointer ;
                VAR joinlist : faultpointer ) ;

VAR

    listheadone , listheadtwo , track : faultpointer ;

BEGIN

    listheadone := listerone ;
    listheadtwo := listertwo ;
    NEW ( joinlist ) ;
    joinlist ^.next := NIL ;
    track := joinlist ;
    WHILE listheadone ^.next <> NIL DO
      BEGIN
        track ^.faultst := listheadone ^.faultst ;
        NEW ( track ^.next ) ;
        track := track ^.next ;
        track ^.next := NIL ;
        listheadone := listheadone ^.next ;
      END ; (* WHILE lisheadone ^.next *)

WHILE listheadtwo ^.next <> NIL DO
  BEGIN
    track ^.faultst := listheadtwo ^.faultst ;
    NEW ( track ^.next ) ;
    track ^.next := NIL ;
    listheadtwo := listheadtwo ^.next ;
  END ;
END ; (* joinsetlist *)

(*-----------------------------------------------------------------*)

PROCEDURE printsetlist ( VAR lister : faultpointer ) ;

CONST

    fixed = 100 ;

VAR

    percentdet : REAL ;
    count , single : INTEGER ;
    linkpointer : faultpointer ;

BEGIN

    count := 0 ;
    single := 0 ;
    WRITELN ;
    WRITELN ('Node No.',' ','Fault No.',' ','Set No.',' ','Low V.',' ',' ',
    ' High V.',' ','fpset',' ','Test No.') ;
    WRITELN (opampstore,'Node No.',' ','Fault No.',' ','Set No.',' ','Low V.',
    ' ',' High V.',' ','fpset',' ','Test No.') ;
    WRITELN (opampstore) ;
    linkpointer := lister ;
    WHILE linkpointer ^.next <> NIL DO
      BEGIN
        WITH linkpointer ^.faultst DO
          BEGIN
            count := count + 1 ;
            IF fpset = 1 THEN
                single := single + 1 ;
            WRITELN  (nodeno:4,'      ',faultno:4,'       ',setstatus.setno:2,'          ',
            setstatus.low :10,'    ',setstatus.high:10 ) ;
            linkpointer := linkpointer ^.next ;
          END ;
      END ; (* WHILE linkpointer *)
WRITELN ;
percentdet := ( count / max ) * fixed ;
WRITELN ( opampstore ) ;
WRITELN ('The Number of Faults Detected is = ' , count:1) ;
WRITELN ( opampstore, 'The Number of Faults Detected is = ' , count:1 ) ;
WRITELN  ( opampstore ) ;
WRITELN ('The Number of Sets Having One Fault = ', single:1) ;
WRITELN (opampstore,'The Number of Sets Having One Fault = ',single:1)  ;
WRITELN ( opampstore,'Percentage of Faults Detected = ',percentdet:3:1,'%') ;
WRITELN  ( opampstore ) ;
WRITELN  ( opampstore ) ;
WRITELN  ( opampstore,'fpset = faults per set' ) ;
WRITELN  ( opampstore,'Low V. % High V. are the bounds of the set' ) ;

END ; (* printsetlist *)

(*----------------------------------------------------------------*)

(* main program -- opamptesting -- *)

BEGIN

  initializestnodes ;
  WRITELN ('Testing of the OP-AMP Circuit') ;
  WRITELN (opampstore,'Testing of the OP-AMP Circuit') ;
  WRITELN (opampstore) ;
  WRITELN (opampstore, '** The Voltage Run **') ;
  WRITELN (opampstore) ;
  WRITELN (opampstore, 'THE TOLERANCE = ',tolerance:%:£ , '   volt') ;
  WRITELN (opampstore) ;
  WRITELN (opampstore,'The Number of Faults Introduced = ',max:1) ;
  WRITELN (opampstore) ;
  WRITELN (opampstore) ;
  echonodes ;
  WRITELN ;
  WRITELN ;
  WRITELN ('The following correspondes to the FIRST stimulant') ;
  WRITELN (opampstore,'The following corresponds to the FIRST stimulant') ;
  WRITELN ;
  WRITELN ('The following corresponds to the SECOND stimulant') ;
  WRITELN (opampstore,'The following corresponds to the SECOND stimulant') ;
  WRITELN (opampstore,'Isolation Status of First Stimulant') ;
  WRITELN (opampstore) ;
  WRITELN (opampstore,'Isolation Status of Second Stimulant') ;
  WRITELN (opampstore) ;
  ambiguitysets ( stimulione , setone ) ;
  ambiguitysets ( stimulitwo , settwo ) ;
  constructlist ( testone , setone , listone ) ;
  printsetlist ( listone ) ;
  constructlist ( testtwo , settwo , listtwo ) ;
  printsetlist ( listtwo ) ;
  compresslist ( listone , complistone ) ;
  printsetlist ( complistone ) ;
  compresslist ( listtwo , complisttwo ) ;
  printsetlist ( complisttwo ) ;
  joinsetlists ( complistone , complisttwo , combinedlist ) ;
  printsetlist ( combinedlist ) ;
  compresslist ( combinedlist , finallist ) ;
  printsetlist ( finallist ) ;

END .  (* opamptesting *)
```

APPENDIX - 2

```
PROGRAM  digcomparator (digcompdata,digcompstore,INPUT,OUTPUT) ;

{ This program applies the FUNCTIONAL K_MAP DIGITAL MODELLING Testing
  Strategy to the CMOS Comparator Circuit. }

CONST

    min = 1 ;
    max = 53 ;
    testsnumb = 4 ;

TYPE

    voltvalue = REAL ;
    faults = min .. max ;
    test = min .. testsnumb ;
    testrecord = RECORD
              testno : tests ;
              nominal : voltvalue ;
              faulty : ARRAY [faults] OF voltvalue ;
              END ;
    Faultrec = RECORD
            tnumb : tests ;
            factor : voltvalue ;
            END ;

VAR

    digcompstore : TEXT ;
    digcompdata : FILE OF testrecord ;
    digitaltests : ARRAY [tests] OF testrecord ;
    finalstatus : ARRAY [faults] OF faultrec ;
    teststate : ARRAY [tests,faults] OF voltvalue ;

PROCEDURE  initialize ;

BEGIN
    REWRITE ( digcompstore ) ;
END ;

PROCEDURE  echodata ;

VAR
    count : 0 .. testnumb ;
```

```
BEGIN
    count := 0 ;
    RESET ( digcompdata ) ;
    WHILE NOT EOF ( digcompdata ) DO
      BEGIN
        count := count + 1 ;
        READ ( digcompdata , digitaltests [count] ) ;
      END ; (* WHILE *)
    WRITELN ('* The Data Stored in digcompdata Have Been Echoed *') ;
END ; (* echodata *)

PROCEDURE checkdatafile ;

VAR

    fft : faults ;
    numtest : tests ;
    tester : testrecord ;

BEGIN

    FOR numtest := min TO testsnumb DO
      BEGIN
        tester := digitaltests [ numtest ] ;
        WITH tester DO
          BEGIN
            WRITELN (digcompstore,'Test No. =  ',testno) ;
            WRITELN (digcompstore,'Nominal Volt =  ',nominal) ;
            FOR fft := min TO max DO
              BEGIN
                WRITELN (digcompstore,'Fault[',fft:2,'] = ',faulty[fft]);
              END ; (* FOR fft *)
          END ; (* WITH tester *)
      END ; (* For numtest *)
      WRITELN ('** File digcompdata Now Stored in digcompstore **') ;
END ; (* checkdatafile *)

PROCEDURE  manipulate ;

VAR

    numtest : tests ;
    testrec : testrecord ;
    fft, count : faults ;
    low, high, temp, diff : voltvalue ;

BEGIN
```

```
    FOR numtest := min TO testsnumb DO
      BEGIN
      WRITELN (digcompstore,'TEST NO = ',numtest:1) ;
      count := 0 ;
      testrec := digitaltests [numtest] ;
      WITH testrec DO
        BEGIN
          WRITELN ('This is Test No. =   ',testno:1) ;
          WRITELN ('The Nominal Voltage =  ',nominal) ;
          WRITELN ('Please Enter LOW & HIGH Bounds Imposed on Nominal') ;
          READLN ( low ) ;
          READLN ( high ) ;
          FOR fft := min TO max DO
            BEGIN
              temp := faulty [fft] ;
              diff := ABS (temp - nominal) ;
              IF (temp < low) OR (temp > high) THEN
                BEGIN
                  teststate [numtest,fft] := diff ;
                  count := count + 1 ;
                  WRITELN (digcompstore,fft:2,'     ',diff) ;
                      END
                ELSE
                  teststate [numtest,fft] := 0 ;
            END ; (* FOR fft *)
            WRITELN ('Test No. =  ',testno:1,' Faults Detected =  ',count:1) ;
        END ; (* WITH testrec *)
      END ; (* FOR numtest *)
END ; (* manipulate *)

PROCEDURE  detectedfaults ;

VAR

    fft,count : faults ;
    voltfact : voltvalue ;
    numtest, index : tests ;

BEGIN

    FOR fft := min TO max DO
      BEGIN
        voltfact := -1.0 ;
        FOR numtest := min TO testsnumb DO
          BEGIN
            IF (teststate [numtest,fft] > voltfact) THEN
              BEGIN
                voltfact := teststate [numtest,fft] ;
        index := numtest ;
       END ;
     END ; (* FOR numtest *)
    finalstatus [fft].tnumb := index ;
    finalstatus [fft].factor := voltfact ;
   END ; (* FOR fft *)

  WRITELN (digcompstore,'FAULT NO.','     ','TEST NO.','     ','FACTOR') ;
  WRITELN (digcompstore) ;

  count := 0 ;
  FOR fft := min TO max DO
   BEGIN
    IF (finalstatus[fft].factor <> 0 ) THEN
     WITH finalstatus [fft] DO
      BEGIN
       count := count + 1 ;
       WRITELN  (digcompstore,fft:2.'     ',tnumb:1,'     ',factor) ;
       WRITELN (digcompstore ) ;
      END ; (* WITH *)
   END ; (* FOR fft *)
  WRITELN ('TOTAL FAULTS DETECTED =  ',count:1) ;
  WRITELN (digcompstore,'TOTAL FAULTS DETECTED =  ',count:1) ;
  WRITELN (digcompstore) ;
  WRITELN (digcompstore,'THE UNDETECTED FAULTS are :  ') ;
  FOR fft := min TO max DO
   BEGIN
    IF (finalstatus[fft].factor = 0 ) THEN
     WITH  finalstatus [fft] DO
      BEGIN
       WRITE (digcompstore,fft:2,'  ') ;
      END ; (* WITH *)
   END ; (* FOR fft *)
END ; (* detectedfaults *)

BEGIN

initialize ;
echodata ;
WRITELN (digcompstore,'DIGITAL TESTING OF THE FAST COMPARATOR');
WRITELN (digcompstore) ;
manipulate ;
detectedfaults ;

END .  (* digcomparator *)
```

APPENDIX - 3

```
PROGRAM pr_hspice (INPUT , OUTPUT) ;        (***  M. A. AL-QUTAYRI  ***)


(* This program processes an HSPICE Transient Analysis output file. It   *)
(* converts it to a file called NEWFILE containing numerical data only.  *)

(* The numerical data file NEWFILE is then split into a number of files  *)
(* whose format are suitable for MATLAB processing. The number of        *)
(* files depends on the number of data blocks in the HSPICE data file.   *)

(* The user will be requested to enter the name of HSPICE file to be     *)
(* processed and the names of the files that will hold the data blocks.  *)

(* All the files created from spliting NEWFILE will be automatically     *)
(* called by a MATLAB analysis file which the user will be prompted      *)
(* to enter its name. This means that once this program is executed      *)
(* the user can edit the MATLAB analysis file and specify the variuos    *)
(* analysis operation that may be required.                              *)

(* The program then counts the number of circuit nodes that have been    *)
(* probed during HSPICE analysis. This information is appended to the    *)
(* MATLAB file created to help the user in any subsequent analysis.      *)


(* ------------------------------------------------------------------ *)
(* ------------------------------------------------------------------ *)


CONST
  min  = 1  ;
  max  = 80 ;
  sent = 10 ;

TYPE
  psudname = STRING [8];
  filename = STRING [12];
  linetx = PACKED ARRAY [min..max] OF CHAR;
  fmrk   = PACKED ARRAY [min..15] OF CHAR;

VAR
  q : CHAR;
  noline, block, nodes : INTEGER ;
  sp_name, fname, mlf, lastfile : filename;
  spdata, newfile, matfile, dummy : TEXT;
```

```
PROCEDURE procfile;

 CONST
  timx = '         time  ';

 VAR
  marker : fmrk;
  cnt , count : INTEGER;
  val, valid  : BOOLEAN;
  line, blank : linetx;

 BEGIN

  WRITELN;
  WRITELN ('*** Please Enter The Name of HSPICE Data File You Would ***');
  WRITELN ('*** Like to Process. Note That The Name Should not be   ***');
  WRITELN ('*** More Than 12 Characters Including 3 Char Extension. ***');
  WRITELN;

  READLN (sp_name);
  ASSIGN (spdata, sp_name);

  ASSIGN (newfile , 'newfile.dat');

  RESET ( spdata );
  REWRITE ( newfile );

  WRITELN ;
  WRITELN ('** Please Wait -- Processing HSPICE file ',sp_name,' **');
  WRITELN ;

  FOR count := min TO max DO     (* Create a Blank Line *)
    blank[count] := ' ';

  block  := 0;
  noline := 0;

  WHILE ( NOT EOF (spdata) ) DO
   BEGIN

    READ (spdata , q);
    cnt   := 0;
    valid := TRUE;
    val   := FALSE;

    WHILE (ORD(q) <> 10) DO
```

```
BEGIN
 cnt := cnt + 1;
 IF (cnt <= max) THEN
   BEGIN

     IF (cnt <= 16) THEN
       marker[cnt] := q;

     line[cnt] := q;
     val := (line[cnt] = ':') OR (line[cnt] = '*');
     IF (line[cnt] IN ['a'..'d','f'..'z']) OR (val = TRUE) THEN
       valid := FALSE
   END
 ELSE
   valid := FALSE ;

 READ (spdata , q);
END; (* WHILE ORD(q) <> 10 *)

IF (marker = timx) THEN
 BEGIN
   block := block + 1;

   REPEAT              (* Skip Next Line *)
   READ (spdata,q);
   UNTIL ( ORD(q) = 10 );

   WRITELN ('No. of Data Block/s Detected so Far is : ',block:1);
 END;

IF (valid = TRUE)  THEN
 BEGIN

   IF (cnt < max) THEN
     FOR count := (cnt+1) TO max DO
        line[count] := ' ';          (* Pad string with blanks *)
   IF (line <> blank) THEN
     BEGIN
       IF ( cnt = max ) THEN
         line[cnt] := ';'
       ELSE
         line[cnt+1] := ';' ;

       IF (block = 1) THEN
         noline := noline + 1;   (* count no. of lines in block *)

       WRITELN (newfile , line);
```

```
            END; (* IF line <> blank*)
          END;

        READ (spdata);

      END; (* WHILE NOT EOF(spdata) *)

    WRITE (newfile,CHAR(26));

    CLOSE (spdata) ;
    CLOSE(newfile);

    WRITELN ;
    WRITELN ('*** File NEWFILE Has been Created ***');
    WRITELN ;
    WRITELN ('Total No. of Data Blocks in ',sp_name,' is : ',block:1);
    WRITELN ('The No. of Lines in Each Data Block of ',sp_name,' is : ',noline:1);
    WRITELN ;
    WRITELN ('** The Data in ',sp_name,' Will Now be Split in ',block:1,' File/s
**');
    WRITELN;

  END; (* procfile *)


PROCEDURE op_mat_file;

  VAR
    pseudo : psudname;

  BEGIN

    WRITELN;
    WRITELN ('** Please Enter The Name of MATLAB File That Will **');
    WRITELN ('** be Used to Perform The Analysis on HSPICE Data **');
    WRITELN ('** Only The First 8 Characters Will be Accepted.  **');
    WRITELN ('** A ".m" Extension Will be Automatically Added.  **');
    WRITELN;

    READLN ( pseudo );
    mlf := pseudo + '.m';
    ASSIGN (matfile , mlf);
    REWRITE (matfile);
    WRITELN (matfile, '% This File Contains Files Created After');
    WRITELN (matfile, '% Processing HSPICE File --> ',sp_name);
    WRITELN (matfile);
    WRITELN (matfile);
```

```
 END; (* op_mat_file *)


PROCEDURE splitfile ;

 VAR
  cnt, blkno : INTEGER;
  pseudo : STRING [8];

 BEGIN

  cnt := 0;
  blkno := 1;

  RESET (newfile);

  WHILE (NOT EOF (newfile)) DO
   BEGIN
    cnt := cnt + 1;

    IF (blkno = 1) AND (cnt = 1) THEN
     BEGIN

      WRITELN ;
      WRITELN ('** Please Enter The Name of File No. : ',blkno:1,' **');
      WRITELN ('** Only The First 8 Characters Will be Accepted **');
      WRITELN ('** A ".m" Extension Will be Automatically Added **');
      WRITELN ;

      READLN ( pseudo );
      fname := pseudo + '.m';
      ASSIGN (dummy , fname);
      REWRITE (dummy);
      WRITELN (dummy,'data',blkno:1,'= [ ');

      WRITELN (matfile , pseudo);

     END; (* IF blkno=1 AND cnt=1 *)

    WHILE (NOT EOLN (newfile)) DO
      BEGIN
       READ (newfile , q);
       WRITE (dummy , q);
      END; (* WHILE NOT EOLN newfile *)
      WRITELN (dummy);
    IF ((cnt MOD noline) = 0) THEN
      BEGIN
       blkno := blkno + 1;
```

```
        cnt := 0;
      (* Close the previous file *)
        WRITELN (dummy,' ];');
        WRITELN (dummy, CHR(26));
        CLOSE (dummy);

      (* Initialise New File *)

        IF (blkno <= block) THEN
         BEGIN

          WRITELN ;
          WRITELN ('** Please Enter The Name of File No. : ',blkno:1,' **');
          WRITELN ('** Only The First 8 Characters Will be Accepted **');
          WRITELN ('** A ".m" Extension Will be Automatically Added **');
          WRITELN ;

          READLN (pseudo);
          fname := pseudo + '.m';
          ASSIGN (dummy , fname);
          REWRITE (dummy);
          WRITELN (dummy,'data',blkno:1,' =[ ');

          WRITELN (matfile , pseudo);

         END;

      END; (* IF *)

    READLN (newfile);

  END; (* WHILE *)

    lastfile := fname;

  close (newfile);

END; (* splitfile *)


PROCEDURE no_nodes ;

 VAR
  q : CHAR;
  valid : BOOLEAN;
  cnt, j, k : INTEGER;
  entry : ARRAY [min..max] OF CHAR;
 BEGIN

   ASSIGN (dummy , lastfile);
   RESET (dummy);
   valid := FALSE;

   WHILE (NOT EOF (dummy)) AND (NOT valid) DO
     BEGIN
       k := 0;
       cnt := 0;

       WHILE NOT EOLN (dummy) DO
         BEGIN
           READ (dummy , q);
           k := k + 1;
           entry[k] := q;
         END; (* WHILE *)

       FOR j := 1 TO (k-1) DO
         IF (entry[j]=' ') AND (entry[j+1] IN ['0'..'9','.','e','-','+']) THEN
           cnt := cnt + 1;

       IF (cnt > 0) AND (cnt <= 5) THEN
         valid := TRUE;

       READLN (dummy);
    END; (* WHILE *)
    CLOSE (dummy);

    nodes := 4 * (block - 1) + (cnt - 1);
    WRITELN ('** Number of Nodes Probed is : ',nodes:1 ,' **');
    WRITELN ('** First Column of Every Data Matrix is TIME **');

    WRITELN (matfile);
    WRITELN (matfile,'% The Number of Nodes Probed is : ',nodes:1);
    WRITELN (matfile);
    WRITELN (matfile,'% Remember that the First Column of Every Data ');
    WRITELN (matfile,'% Matrix in the above M Files is the TIME Entry');
    WRITELN (matfile);

 END; (* no_node *)


PROCEDURE cl_mat_file;

 BEGIN

   WRITELN (matfile , CHR(26));
```

```
    CLOSE (matfile);
    WRITELN ;
    WRITELN ('** The MATLAB File Created is Called -- ',mlf,' -- **');
    WRITELN ;

  END; (* cl_mat_file *)

(* ------------------------------------------------------------------- *)

BEGIN  (* pr_hspice *)

  procfile;

  op_mat_file;

  splitfile;

  no_nodes;

  cl_mat_file;

END. (* pr_hspice *)
```