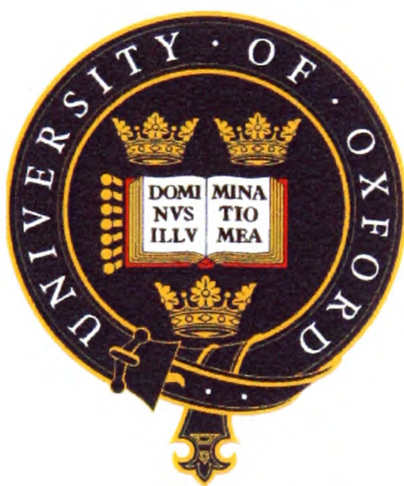
Intra-Gate Fault Diagnosis of CMOS Integrated Circuits

Xinyue Fan

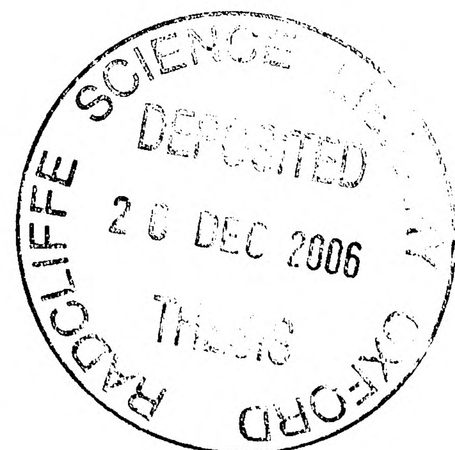
St Cross College



Thesis submitted for the degree of Doctor of Philosophy

at the University of Oxford

Trinity 2006


RADCLIFFE SCIENCE LIBRARY OXFORD
DEPOSITED
2 0 DEC 2006
THESIS

To My Parents and My Grandma

Abstract

Xinyue Fan **Doctor of Philosophy**

St Cross College **Trinity 2006**

Intra-Gate Fault Diagnosis of CMOS Integrated Circuits

Knowing the root cause of why an Integrated Circuit (IC) device fails to function properly is the key to provide the corrective measures to increase the yield and shorten the time to market. In recent years, electrical fault diagnosis method has received growing attention due to the effective and indispensable guiding role it plays in modern fault localization practice when physical measures are more and more confined by the shrinking feature size and condensed internal structure.

While most of the fault diagnosis tools are based on gate level fault models, many faults are actually at the transistor level (the intra-gate fault). This thesis provides an innovative method to diagnose the intra-gate faults. It covers a wide range of different types of intra-gate faults. The method extends the capability of gate level fault diagnosis tools to the intra-gate domain by building connections with these intra-gate faults to particular types of gate level faults. Intra-gate faults are transformed to gate level representations so that they can be diagnosed directly by the widely available and well developed gate level diagnosis tools. Real diagnosis of intra-gate faults from wafer data and physical failure analysis photos are provided as solid proofs of the effectiveness of this method.

Acknowledgements

I would like to give my sincere gratitude to my supervisor, Dr. Will Moore, for guiding me through the DPhil research at Oxford, and for his constant encouragement, friendship, inspiring advice and support over the past three years.

I would also like to express my gratitude to ED&T of Philips Research, University UK Overseas Research Students Award Scheme and Henry Lester Trust for their financial support. I am also very grateful to Philips Research for the access of internal software tools and for the provision of samples and data. Special thanks are given to Dr. Camelia Hora for the very useful discussions and the solid previous work from which this thesis builds up. Much of the work in this thesis would have been impossible without the generous help from Mr. Maurice Lousberg, Dr. Stefan Eichenberger, Mr. Guido Gronthoud, Dr. Mario Konijnenburg, Dr. Ananta Majhi and Dr. Mohamed Azimane. I am very grateful to them for the fascinating discussions in which I learned a lot from their expertise and for the friendship I enjoyed during the time I spent in Philips Research, The Netherlands.

Last but not least, I would like to thank my parents and my grandma in China, and my friends here at Oxford for their love and support.

Table of Contents

List of Figures v

List of Tables ix

Chapter 1. Introduction 1

1.1. Background 1

1.2. Intra-gate fault diagnosis 3

1.3. Outline of the Thesis 4

Chapter 2. IC Testing and Diagnosis in General 5

2.1. Introduction 5

2.2. IC Testing in General 6

2.2.1. CMOS basics 6

2.2.2. CMOS Manufacturing 9

2.2.3. Common Physical Faults in CMOS Circuits 12

2.2.4. Fault Models and Test Pattern Generation 17

2.2.5. Design for Testability 23

2.3. Electrical Fault Diagnosis 28

2.3.1. Introduction 28

2.3.2. Diagnosis algorithms 29

2.3.3. Fault diagnosis with fault models 39

2.4. Conclusion 61

Chapter 3. Intra-gate Open Fault Diagnosis 63

3.1. Introduction 63

3.2. Stuck-open fault model 64

3.3. Stuck-open fault diagnosis 66

3.3.1. Stuck-open fault under stuck-at diagnosis 67

3.3.2. Transforming stuck-open faults to stuck-at faults 71

3.3.3. Stuck-open fault diagnosis flow 81

3.3.4. Implementation and experimental results 83

3.3.5. Consecutive high impedance vectors 98

3.3.6. Short summary 106

3.4. Intra-gate resistive open fault diagnosis 106

3.4.1. Delay fault testing 107

3.4.2. Intra-gate resistive open fault diagnosis 109

3.5. Conclusion 114

Chapter 4. Intra-gate Bridging Fault Diagnosis 116

4.1. Introduction 116

4.2. Method to shortlist possible gates with intra-gate bridging faults 117

4.3. Intra-gate bridging fault transformation 121

4.3.1. The transformation from Principle 1 122

4.3.2. The transformation from Principle 2 126

4.4. Intra-gate diagnosis flow 128

4.5. Experimental results from wafer testing data 129

4.6. Conclusion 134

Chapter 5. Diagnosis Flow for Intra-gate faults 135

Chapter 6. Conclusions and Future Work 139

6.1. Conclusions 139

6.2. Future work 142

References 143

Appendix A. Failure Analysis Techniques 150

Appendix B. Published Papers on Intra-gate Fault Diagnosis 169

List of Figures

FIGURE 2.1 LOGIC SWITCHING MODEL 7

FIGURE 2.2 CMOS INVERTER 8

FIGURE 2.3 CROSS SECTION OF A CMOS CIRCUIT 9

FIGURE 2.4 THE BASIC CMOS PROCESS 10

FIGURE 2.5 A PINHOLE DEFECT IN SILICON DIOXIDE 13

FIGURE 2.6 PICTURE OF OPEN CONTACT 14

FIGURE 2.7 COMPARISON OF NON-PLANARIZED AND PLANARIZED DEVICES 15

FIGURE 2.8 CMP SCRATCH CAUSING OPEN METAL CONNECTION 16

FIGURE 2.9 A PHYSICAL LEVEL EXAMPLE OF STUCK AT FAULT 19

FIGURE 2.10 TEST PATTERNS FOR NOR GATE 21

FIGURE 2.11 TEST PATTERN GENERATION FOR STUCK-AT-0 FAULT 21

FIGURE 2.12 THE MAIN FLOW OF ATPG DELIVERING A FAULT COVERAGE TARGET 23

FIGURE 2.13 A BASIC STRUCTURE OF SCAN BASED TEST 24

FIGURE 2.14 BUILT-IN SELF-TEST CIRCUIT CONFIGURATION 25

FIGURE 2.15 EXAMPLE OF STANDARD LFSR CONFIGURATION 26

FIGURE 2.16 NON-PREDICTION AND MIS-PREDICTION 32

FIGURE 2.17 MATCHING ALGORITHM OF 35

FIGURE 2.18 AN EXAMPLE OF *OBS* AND *SIM* 36

FIGURE 2.19 POSSIBLE RELATIONS OF *OBS* AND *SIM* 37

FIGURE 2.20 LOGIC CONE 39

FIGURE 2.21 FAN-OUT NET WITH FOUR BRANCHES B1, B2, B3, B4 41

FIGURE 2.22 BYZANTINE GENERAL'S PROBLEM 46

FIGURE 2.23 FEEDBACK BRIDGING FAULT CAUSING DYNAMIC BEHAVIOUR ... 47
FIGURE 2.24 EXTRACTION OF REALISTIC BRIDGING FAULTS ... 49
FIGURE 2.25 MODELLING WITH WIRED-AND AND WIRED-OR ... 51
FIGURE 2.26 RESISTIVE OPEN AND HARD OPEN ... 53
FIGURE 2.27 RAFIQ'S FLOATING GATE MODEL ... 54
FIGURE 2.28 A SUB-SECTION OF CIRCUIT CONTAINING A HARD OPEN FAULTS ... 55
FIGURE 2.29 RELATION OF OBS AND SIM OF D STUCK-AT-1 ... 56
FIGURE 2.30 EQUIVALENT STUCK-AT FAULTS OF D STUCK-AT-1 ... 57
FIGURE 2.31 RELATIONSHIP OF OBS AND SIM OF NET D UNDER NET DIAGNOSIS MODEL ... 58
FIGURE 3.1 STUCK-OPEN FAULT AT TRANSISTOR T1 ... 65
FIGURE 3.2 SCHEMATIC OF THE EXAMPLE CIRCUIT ... 72
FIGURE 3.3 REPLACEMENT OF N-TRANSISTOR ... 72
FIGURE 3.4 REPLACEMENT OF PARALLEL N-TRANSISTORS ... 73
FIGURE 3.5 REPLACEMENT OF P-TRANSISTOR ... 73
FIGURE 3.6 REPLACEMENT OF PARALLEL P-TRANSISTORS ... 74
FIGURE 3.7 STUCK-OPEN FAULT AT C_1 ... 74
FIGURE 3.8 STUCK-AT FAULT AT C_1 ... 76
FIGURE 3.9 INVALID SENSITIZATION VECTOR ... 76
FIGURE 3.10 THE FIRST PHASE OF TRANSFORMATION ... 77
FIGURE 3.11 FINAL PICTURE OF TRANSFORMATION ... 79
FIGURE 3.12 MULTIPLE BLOCKS GATE TRANSFORMATION ... 81
FIGURE 3.13 OVERALL FLOW OF STUCK-OPEN FAULT DIAGNOSIS ... 82
FIGURE 3.14 THREE EXPERIMENTAL GATES ... 84
FIGURE 3.15 SCHEMATICS OF STUCK-OPEN GATES ... 91
FIGURE 3.16 MULTIPLE STUCK-OPEN FAULTS ... 94
FIGURE 3.17 LAYOUT ANALYSIS OF AO52 ... 96

FIGURE 3.18 WAFER MAP 97
FIGURE 3.19 MISSING PATTERNING 97
FIGURE 3.20 GATE AO36 WITH STUCK-OPEN FAULT AT TRANSISTOR A_1 98
FIGURE 3.21 SUSPECTED GATE 100
FIGURE 3.22 A SEQUENCE OF TEST VECTORS 101
FIGURE 3.23 AN EXTENDED VERSION OF TRANSFORMATION FOR CONSECUTIVE HIGH-IMPEDANCE ISSUE 102
FIGURE 3.24 FA RESULT OF STUCK-OPEN DEVICE 105
FIGURE 3.25 BASIC STRUCTURE OF DELAY FAULT TESTING 108
FIGURE 3.26 TRANSITIONAL DELAY FAULT CAUSED BY RESISTIVE OPEN AT A_2 109
FIGURE 3.27 TRANSFORMATION OF SLOW-TO-TURN-ON 110
FIGURE 3.28 COMPARISON OF SLOW-TO-TURN-ON AND SLOW TO TURN-OFF 111
FIGURE 3.29 INTEGRATED TRANSFORMATION FOR INTRA-GATE TRANSITIONAL DELAY FAULT 112
FIGURE 4.1 SAMPLE GATE 118
FIGURE 4.2 GATE TO BE TRANSFORMED 122
FIGURE 4.3 REPLACEMENT OF N-TRANSISTOR 123
FIGURE 4.4 REPLACEMENT OF PARALLEL N-TRANSISTORS 123
FIGURE 4.5 FIRST ROUND AND SECOND ROUND JUSTIFICATIONS 124
FIGURE 4.6 FINAL JUSTIFICATIONS 125
FIGURE 4.7 REPLACEMENT OF P-TRANSISTOR 125
FIGURE 4.8 THE TRANSFORMATION OF P TRANSISTORS 126
FIGURE 4.9 FINAL TRANSFORMATION 127
FIGURE 4.10 INTRA-GATE DIAGNOSIS FLOW 129
FIGURE 4.11 SEVEN INTRA-GATE BRIDGING FAULTS 131
FIGURE 4.12 FAILURE ANALYSIS OF DIE #1, DIE #2, DIE #3 133

FIGURE 5.1 DIAGNOSTIC MAP OF INTRA-GATE FAULTS..........................136

List of Tables

TABLE 2.1 EXAMPLE TEST SCENARIO 30

TABLE 2.2 MATCHING AND PREDICTION OF FIGURE 2.19 38

TABLE 2.3 INDICATION OF POSSIBLE FAULT TYPE 43

TABLE 2.4 *OBS* OF THE EXAMPLE IN FIGURE 2.28 56

TABLE 2.5 *SIM* OF D STUCK-AT-1 56

TABLE 2.6 *SIM* OF D UNDER NET DIAGNOSIS MODEL 58

TABLE 2.7 MATCHING AND PREDICTION UNDER NET DIAGNOSIS MODEL 59

TABLE 3.1 THE BEHAVIOUR OF DIFFERENT FAULTS IN FIGURE 3.1 67

TABLE 3.2 INITIALIZATION AND SENSITIZATION VECTORS OF EXPERIMENTAL GATES 84

TABLE 3.3 DIAGNOSIS RESULTS 86

TABLE 3.4 SEVEN FULLY DIAGNOSED SINGLE STUCK-OPEN FAULTS . 89

TABLE 3.5 STUCK-OPEN FAULTS AFFECTED BY TIMING SKEW 93

TABLE 3.6 MULTIPLE STUCK-OPEN FAULTS IN ONE GATE 95

TABLE 3.7 RESULT FROM THE FIRST RUN STUCK-AT DIAGNOSIS 99

TABLE 3.8 DIAGNOSIS RESULT AFTER NORMAL STUCK-OPEN TRANSFORMATION 100

TABLE 3.9 DIAGNOSIS RESULTS AFTER THE EXTENDED TRANSFORMATION 104

TABLE 3.10 SIMULATION BASED EXPERIMENT 113

TABLE 4.1 VECTORS OF SAMPLE GATE 117

TABLE 4.2 SUCCESSFULLY DIAGNOSED DIES 130

Chapter 1. Introduction

1.1. Background

The technology of modern Integrated Circuit (IC) has developed to a state where a single modern IC device can have more than one billion transistors, miles of narrow metal interconnections and billions of metal vias or contacts [Segura04]. Therefore ensuring the quality of Integrated Circuit (IC) products has become a more challenging task than ever. Failures can have a severe business impact. The largest factor impacting product life-cycle profits is the time to market and this is typically more important than developments budgets or production costs [Turino97], it is therefore a priority for IC companies to understand the causes of failure and to take corrective actions as quickly as possible.

Failure Analysis (FA) refers to the activities undertaken to find the root cause of failure in a given IC device. The size and complexity of modern IC design make it imperative to localize the defects prior to any destructive analysis. To achieve this, it is necessary to combine a series of both physical and electrical steps aiming at finding out the exact location and failure mechanism of the real defects. Physical measures either make use

of passive observation of a physical phenomenon that occurs along with the defects and its effects on the chip such as the concentrated heat area, or take a more active style such as using an election beam to provide interactive stimulus to the device and observe the changes. However, the effectiveness of physical measures are hindered by the fact that not all defects have the physical phenomenon like light emitting or localized heating, and as wiring density and layers increase rapidly, accessing the internal areas of a device without destructive measures becomes extremely difficult .

Therefore failure analysis has turned increasingly to electrical measures, normally referred as Fault Diagnosis, to provide accurate diagnosis of the defect. Basic fault diagnosis algorithms consist of using a fault model to predict the possible behaviour of the faulty circuit and identify the one that most closely matches with the observed faulty behaviour. A lot of progress has been made on fault diagnosis practice in the last two decades. Fault models have been extended from the stuck-at model to bridging models, open models, delay models and others. Matching algorithms have developed from exact matching to more sophisticated versions [Waicukauski89] [Kunda93] [Lavo96] [Li01] [Venkataraman01] [Hora02]. Software diagnosis tools are now well developed to cover a variety of defect types and successful diagnosis results have also been reported frequently [*e.g.* Venkataraman01] [Hora02].

1.2. Intra-gate fault diagnosis

Despite this success in the fault diagnosis development, one area is almost always neglected — the intra-gate fault diagnosis. Almost all the diagnosis tools developed so far [Venkataraman01] [Hora02] are based on the gate level circuit netlist. Although proven to be very effective and comprehensive in the gate level fault diagnosis, they do not have the capability to handle the intra-gate level defects. One of the main reasons for the absence of intra-gate level analysis is that the logic simulation cannot accurately predict the behaviour within a gate and transistor-level electrical simulation is extremely time-consuming when conducted on a large scale.

This thesis focuses on the intra-gate fault diagnosis. It provides an innovative method based on the idea of transforming the intra-gate level faults to a gate level representation, so that the well developed gate level diagnosis tools like POIROT [Venkataraman01] and FALOC [Hora02] can be implemented to diagnose the intra-gate faults directly. This method exploits the efficiency of gate level logic simulation and avoids the unnecessary time that would otherwise be spent on electrical simulation. By having this method, gate level fault diagnosis tools can extend their capability to various types of intra-gate defects: Methods have been developed to use gate level stuck-at diagnosis tools to handle the intra-gate stuck-open faults, intra-gate resistive open faults and intra-gate bridging faults. To our best knowledge, this thesis is the only work systematically addressing the intra-gate fault diagnosis issue and it is the first time that intra-gate faults have been diagnosed and then confirmed by real

wafer data failure analysis pictures.

1.3. Outline of the Thesis

The first part of Chapter 2 gives a brief review of the general knowledge and most important aspects on IC testing. The second part of Chapter 2 focuses on the gate level electrical fault diagnosis issue which is the foundation of this thesis development. Chapter 3 explains the novel method developed for intra-gate open fault diagnosis, including stuck-open faults, and intra-gate resistive open faults. Chapter 4 is the complementary part of the method that addresses the issue of intra-gate bridging fault diagnosis. Chapter 5 shows how the various parts of the method come together with a practical diagnosis flow and Chapter 6 draws the conclusion and looks at the possible future work.

Chapter 2. IC Testing and Diagnosis in General

2.1. Introduction

It is clearly the responsibility of the Integrated Circuits (IC) manufacturers to ensure that every IC product is fully functional before it can be shipped to their clients. Unlike many other systems that operate at normal life size, the issue of testing for IC system suffers from the fact that the visual inspection is of very little use due to the complexity and size of its features [Hurst98a] [Wilkins86]. As the feature size of the devices continue to shrink to the deep-submicron domain (the leading players in this industry are now moving toward to the 65 nanometer and 45 nanometer technology), the quality of deep-submicron ICs places very high demands on the factory buildings, the production environment and the chemicals [Veendrick98]. The slightest variations in temperatures, humidity, the UV light in the photolithographic process, etc, can have serious impact on the final IC products and cause yield loss to the manufacturers. Unfortunately, comprehensive testing of every IC product is an ideal but basically impractical target. Thus, the ability to test end products as comprehensively as possible within a given budget or time constraint is a universal problem [Hurst98a]. It is also very important to understand the root cause of a faulty device, which incurs the need to

perform diagnosis work. Remedy actions can be taken if detailed knowledge of the location and mechanism of the fault become available.

This chapter covers some of the main topics related to IC testing. Firstly, the CMOS circuit is introduced along with details of its manufacturing process and the relevant technologies. Secondly, based on the knowledge of CMOS manufacturing, some of the most common physical faults in CMOS circuits caused by imperfections in the manufacturing process are explained. Next, different types of fault models are presented as mathematical means to handle these real defects in higher abstract levels. As a core issue in determining the efficiency and quality of a test, digital test pattern generation problem is addressed in the next section, followed by introductions of some of the Design for Test (DFT) measures such as Scan Chain and BIST developed to alleviate the difficulties of testing large scale designs. Section 2.3 talks about techniques used in electrical fault diagnosis and the diagnosis issues for various types of faults.

2.2. IC Testing in General

2.2.1. CMOS basics

All digital circuits require three terminal switching devices (Figure 2.1) as the basic components to realize controlled status, on and off, to represent the digital 1 and 0 [Plummer00].

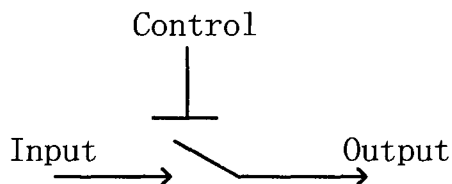


Figure 2.1 Logic switching model

Today, more than 90% of the ICs manufactured use MOSFET (Metal-Oxide Semiconductor Field-Effect Transistor) as the basic switching elements [Plummer00]. Among them, the CMOS design plays a dominant role in providing its unique power saving advantages by adopting a complementary design [Veendrick98] [Plummer00].

The acronym CMOS stands for Complementary Metal Oxide Semiconductor. It is complementary in that every n type MOSFET (nMOS) used in CMOS technology has its counterpart p type MOSFET (pMOS). The most basic element in digital CMOS circuits is an inverter which consists of one nMOS transistor and one pMOS transistor. Each type of MOSFETs has three nodes denoted as source, drain and gate, shown in Figure 2.2. In a CMOS inverter, the gates of the two MOSFETS are tied together as the input net while the two drains are put together as the output nets. The source of pMOS connects to the power supply Vdd and the source of nMOS connects to the ground. See Figure 2.2.

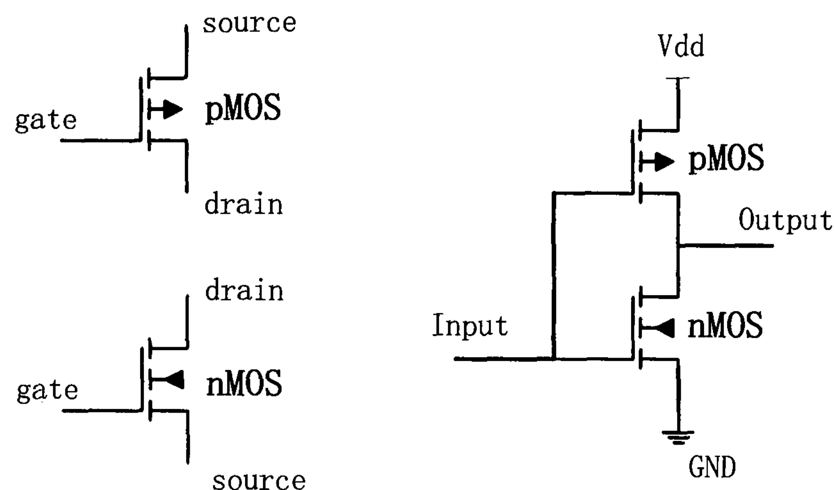


Figure 2.2 CMOS inverter

When the input signal is set to high voltage to represent digital 1, i.e. $V_{Input} = V_{dd}$, the gate-to-source voltage of the nMOS equals to V_{dd}, which turns it on. At the same time, the gate-to-source voltage of the pMOS equals to zero, which turns the pMOS transistor off. The voltage of the output net is thus pulled down to zero by a current flowing through the nMOS transistor. The reversed pMOS-on and nMOS-off situation happens when the input signal is set to zero, and connects V_{dd} to the output. Therefore by allowing only one transistor to be conducting at a time, the simple CMOS circuits realizes the logic function of an inverter. Because of this unique feature of CMOS that there are only half of the transistors conducting at any time, there is near to zero static current or static power dissipation. Only when the inverter is actually switching does a transient current appears [Plummer00] [Taur98]. All CMOS circuits have the same property. With the millions of transistors in today's digital chips, power consumption and cooling requirements become critical issues in obtaining high level of integration. This major advantage over the earlier technologies has helped CMOS to become the most popular technique in today's IC design [Veendrick98] [Plummer00].

2.2.2. CMOS Manufacturing

From the physical point of view, a cross section picture of a final CMOS integrated circuit typically looks like Figure 2.3. As shown in this illustration, a MOSFET consists of three electrodes – the source, gate and drain. In general, a polysilicon film in which phosphorus atoms are doped is utilized as the gate electrode. Source and drain regions are formed to be connected with the inversion layer under the MOS gate. They are normally formed by ion implantation. For the CMOS device, both nMOS and pMOS sit on the same substrate. To construct this simple CMOS section that includes only one p transistor and one n transistor, a variety of different materials have to be used to form a number of components and layers according to a pre-designed pattern. The picture in Figure 2.4 shows the basic process of CMOS manufacturing. The basic CMOS process begins with the choosing of a wafer substrate, in this example, the p-type substrate is chosen.

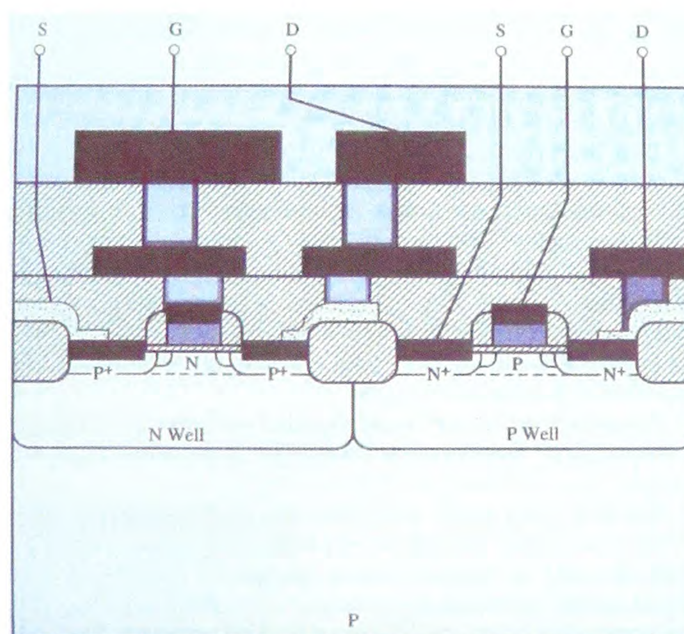


Figure 2.3 Cross section of a CMOS circuit [Plummer00]

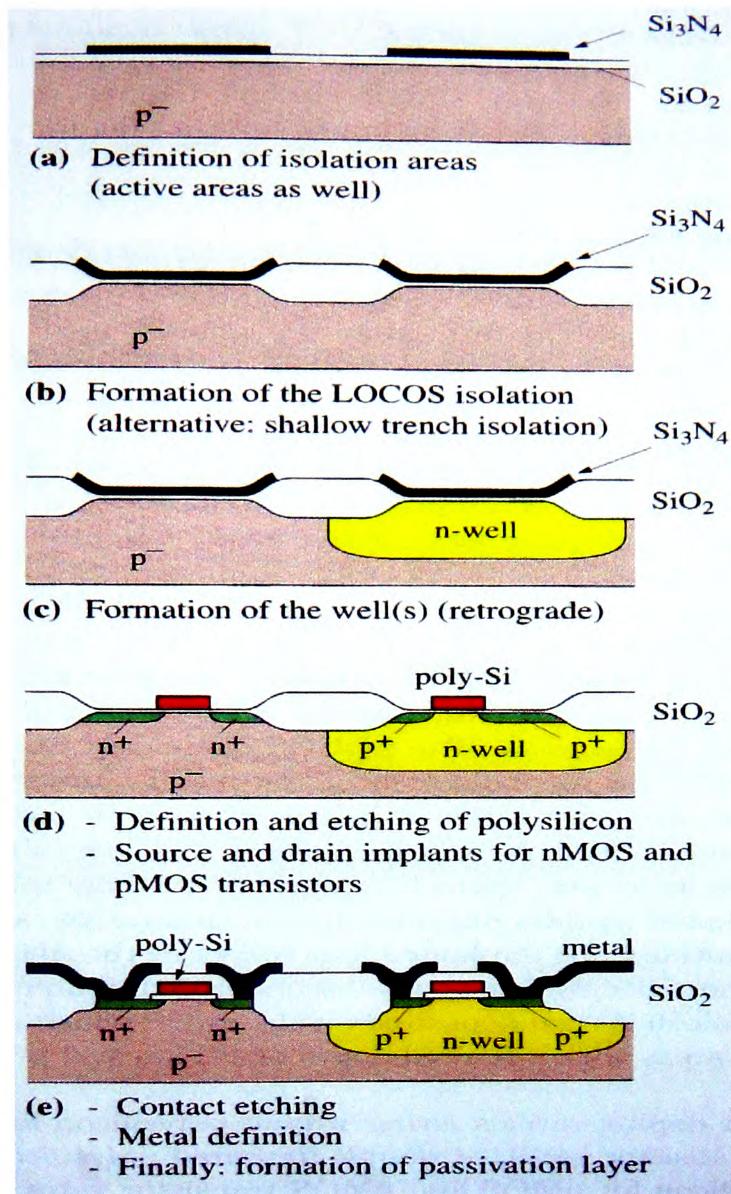


Figure 2.4 The basic CMOS process [Veendrick98]

The wafer is exposed in a high temperature furnace to a gaseous source of a p-type dopant such as boron. The boron diffuses into the crystal by diffusion, creating the p-type substrate [Veendrick98] [Plummer00] [Uyemura02]. A layer of silicon nitride (Si_3N_4) is then deposited on the substrate, plus a photoresist layer. This photoresist layer is later to be subjected to a lithographic process and a pattern is created in the photoresist layer to decide the active area where the circuit elements will be created. The area covered by the photoresist mask will remain during the subsequent etching steps but the rest of Si_3N_4 layer will be etched away (shown in Figure 2.4.a). The wafer is then exposed to oxygen at a high temperature to create a oxide area called

LOCOS (Local Oxidation of Silicon). This area sits on the substrate to separate different active areas like the one in the middle of the p-transistor and n-transistor in Figure 2.4.b. Another photoresist layer is created and subjected to a lithographic process to expose the areas that need to be implanted with phosphorous to form the n-well in Figure 2.4.c. The implanted ions diffuse through the wafer surface and get deeply into the substrate during a high temperature step. The creation of the polysilicon is very much similar to the previous steps. A layer of polysilicon is first deposited on the wafer. A lithographic step follows to create the mask that would retain the required pattern of polysilicon layer during the etching step. Implantation processes for the source and drain area are conducted in the same way that another two masks are used to define the source and drain area of each transistor. The picture in Figure 2.4.d shows the status after the implantation. Up until this stage, both n-transistor and p-transistor have been completely formed, but to connect these transistors according to the designs means further metal layers have to be put on top of the transistors. This step begins with depositing a SiO_2 layer on the wafer to isolate the first metal layer and the transistor surface. The only connections are made through the contact holes created by the same lithographic and etching steps, Figure 2.4.e.

A final CMOS product normally uses 20 to 30 masks. As explained earlier, the wafer undergoes repetitive process including Lithography, Etching, Thermal Oxidation, Deposition, Planarisation.

2.2.3. Common Physical Faults in CMOS Circuits

An integrated circuit can fall victim to a large variety of failure mechanisms. There are all kinds of factors during the manufacturing process that could result in a failure device and cause yield loss. Temperature, humidity, vibration, light, dust particle, the purity of the chemicals all have significant impact on the quality of the final devices [Veendrick98]. It is difficult to mention all the possible causes and types of CMOS defects. The following three sections give a brief review of some of the most commonly occuring defects.

2.2.3.1.Lithographic defects

Lithographic defects are caused by the imperfections in the mask so that the layout of this particular mask does not reflect the original specifications. Any variations of the mask are faithfully reproduced onto the silicon structure during wafer processing. Fluctuations in temperature may cause the projected image of the mask on the wafer to exceed the required tolerances. The slightest vibrations during the lithographic step can cause inaccurate pattern images on the wafer and result in open or short circuits. Particles that contaminate the wafer may damage the layer or disturb a lithographic process. Such errors will result in either extra or missing materials on the substrate, depending on whether the previous deposit layer is positive or negative photo-resist [Veendrick98] [Mourad00]. An extra metal or polysilicon may cause shorts between the transistors that are not meant to be connected. A missing metal, on the contrary, can cause open breaks on the wanted connections.

2.2.3.2. Oxide defects

The yield loss in gate oxide is largely caused by the formation of pinhole defects due to insufficient oxygen at the interface of silicon and silicon dioxide, nitride cracking during field oxidation, and crystal defects [Mourad00]. Other contaminations in the interface between the silicon and silicon dioxide would also lead to pinhole defects as shown in Figure 2.5.

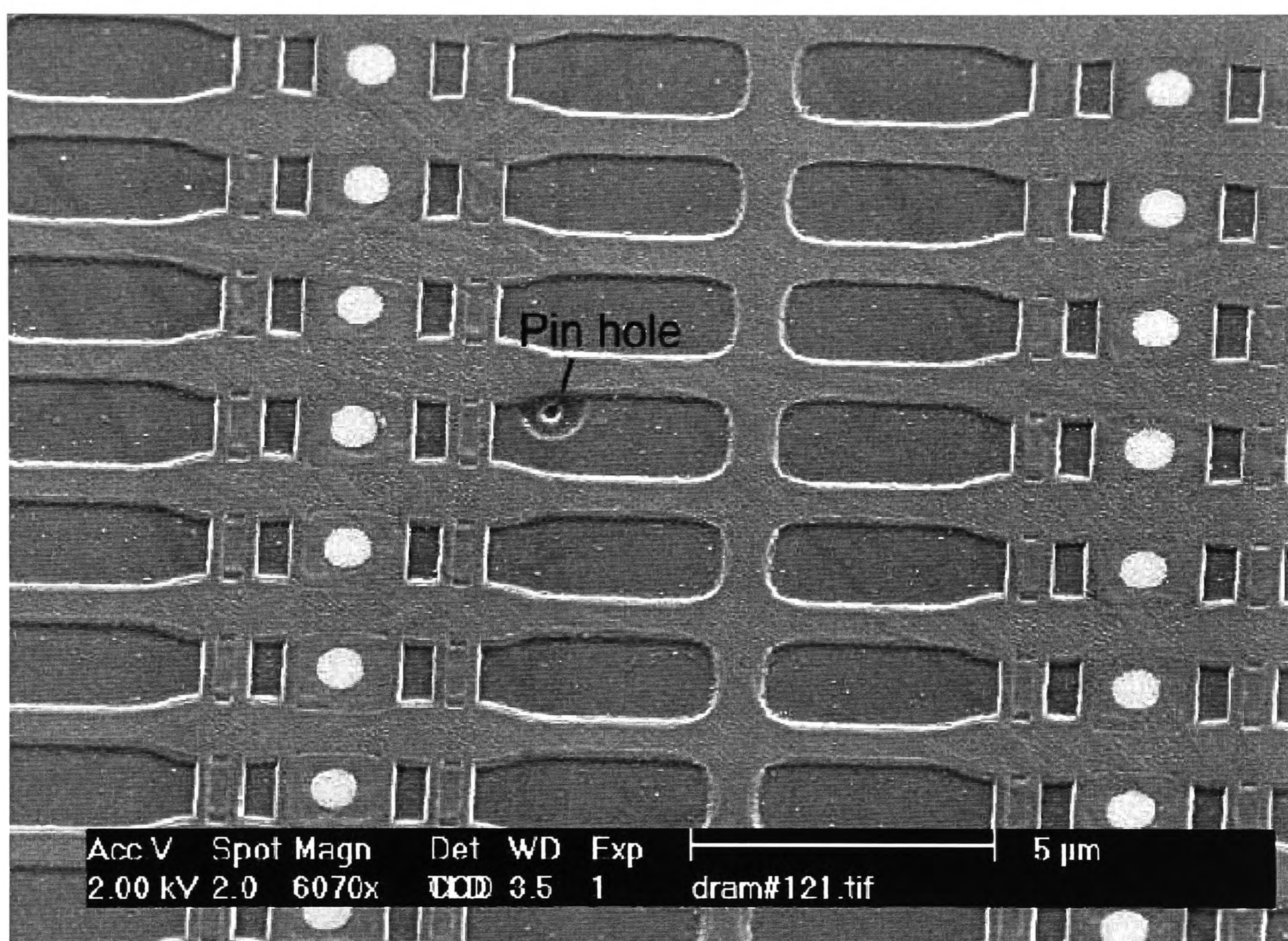


Figure 2.5 A pinhole defect in silicon dioxide [Philips FA Lab]

There are also operational causes that would lead to dioxide layer breakdown. For example, when the dioxide layer thickness decreases but the power supply voltage remains the same, the electric field across the oxide is increased and causes local breakdown. During the plasma and ion implantation process, devices may be exposed to charges collected by aluminum or polysilicon conducting lines connected to the gate

electrodes of MOSFETs. Hot electrons may also cause trapping of charges in the gate oxide layer. As a result, the rising stress currents may eventually cause oxide breakdown and affect the functionality of the device [Mourad00] [Hora02].

Not only can the oxide defects break down the oxide layer to short the insulated components, but they may also block the layers that are meant to be connected. Figure 2.6 shows a contact hole meant to connect the metal layer on top of the oxide and the transistor channel underneath. Not all the oxide materials are removed during the etching step. A thin layer of oxide remains at the bottom of the contact hole, preventing the subsequent step of building a metal connection through the contact hole.

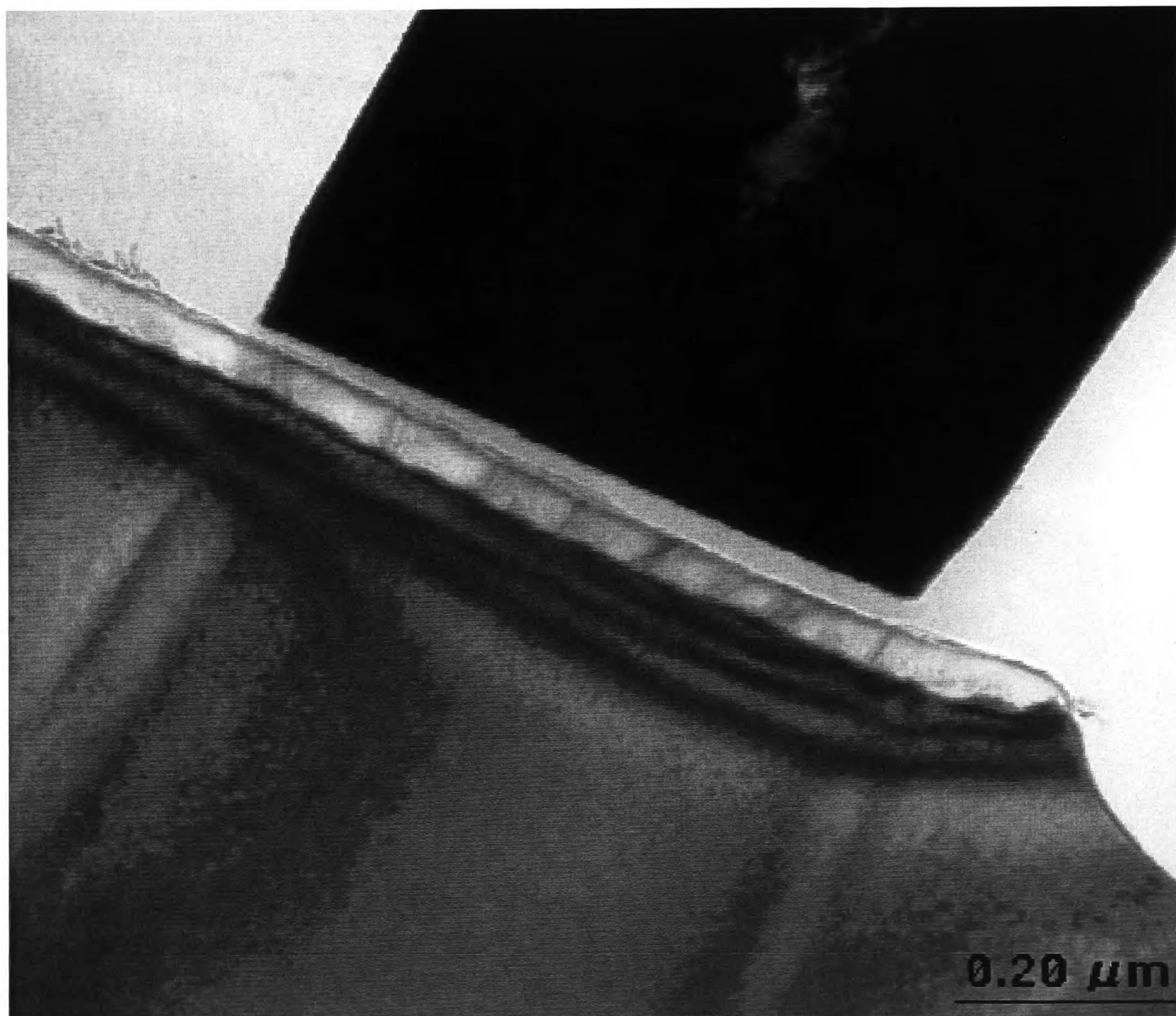


Figure 2.6 Picture of open contact [Philips FA Lab]

2.2.3.3. Metal Failure Defects

Metal failures are caused by various reasons. First of all, as addressed in section 2.2.3, dust particles on the mask during the photolithographic process may cause unwanted metal or unwanted etching of metal. Secondly, metal failures can be introduced by the CMP (Chemical Mechanical Polishing) planarisation process. The need of CMP planarisation arises from an increasingly uneven surface of the devices as a result of the increasing number of processing steps. Therefore, all submicron processes use several planarasation steps to flatten the surface before the next step is performed. Effective planarized surface has as enormous amount of benefits like minimization of prior level defects, elimination of contact interruption and limitation in the stacking height of metallization layers [Zantye04]. Figure 2.7 shows a comparison of non-planarized device and planarized device.

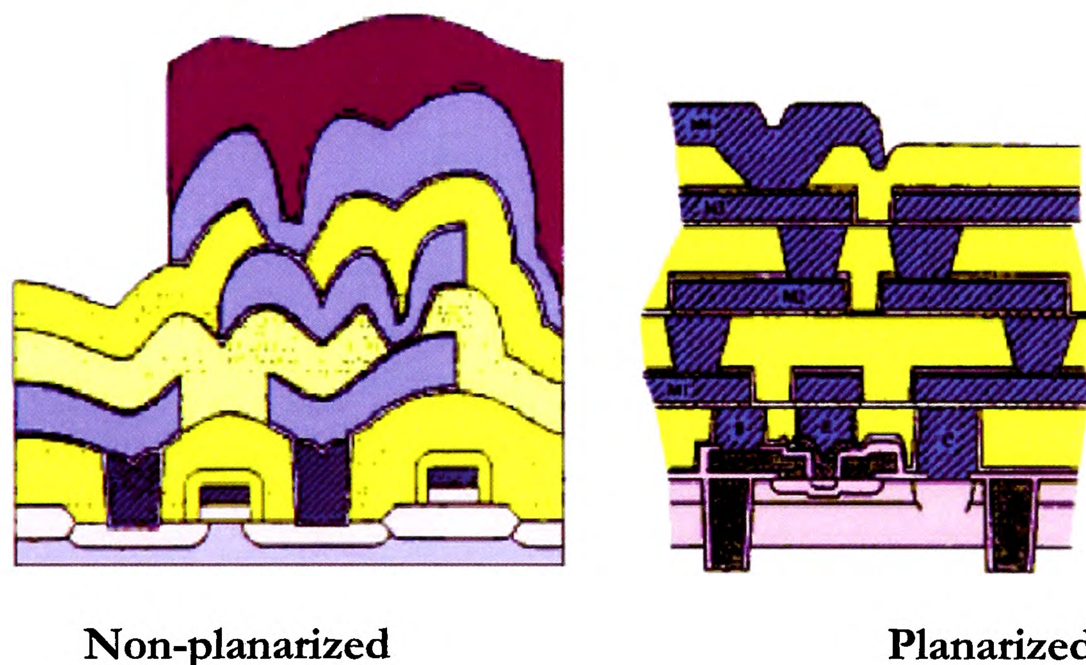


Figure 2.7 Comparison of non-planarized and planarized devices [Zantye04]

However, because of the chemical reactions and the presence of abrasive particles at the

interface can introduce surface defects and contaminations. Particle defects generate from adhesion of the various particles generated during the process of polishing. Chemical contaminations can cause corrosion of exposed metal portions on the wafer surface. Scratches, voids, grooves and pits are also some of the typical surface defects [Zantye04]. One example is given in Figure 2.8 that scratches are caused by the planarisation process on the oxide layer. The residues of etching substance are left in the deep scratch hole after the cleaning step. The metal layer beneath are eroded as a result, forming an open defect.

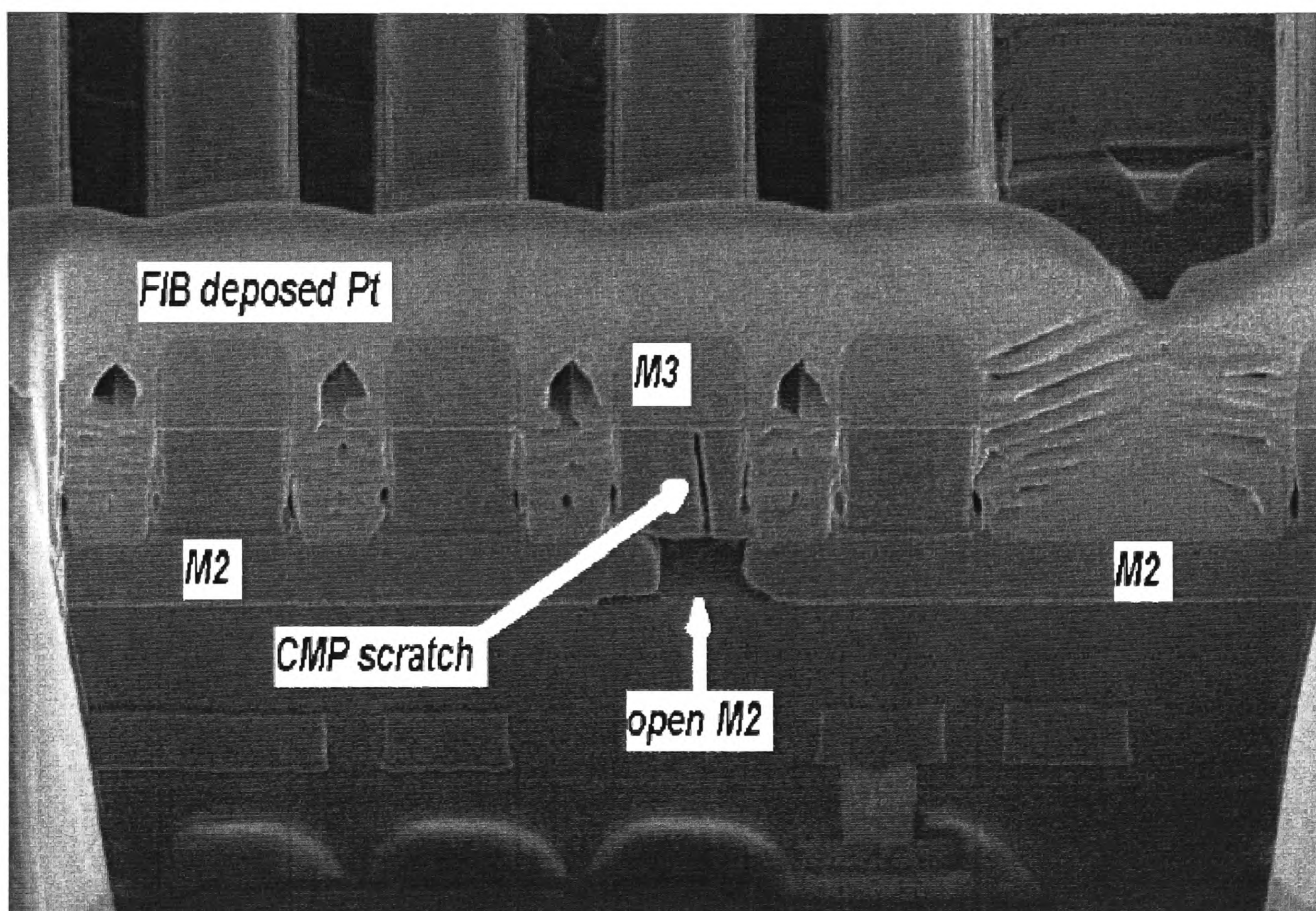


Figure 2.8 CMP scratch causing open metal connection [Philips FA Lab]

2.2.4. Fault Models and Test Pattern Generation

2.2.4.1.Fault Models

An electrical test consists of applying electrical signals at inputs and observing electrical responses at the outputs of devices and is the main framework of IC testing. It is first important to distinguish a physical defect such as those described in the previous section and a fault, which is the electrical effect of the defect. A particular fault could result from different kinds of defects. The fault models serve as the bridge between defects and faults, so engineers can concentrate on the electrical attributes of a fault without necessarily relating the faults to particular defects. There are basically two ways to construct a test [Hurst98a]:

(1) A series of functional tests and check for the correct 0 or 1 output responses

(2) To consider the possible faults that may occur in the circuit, and apply a series of tests that are particularly designed to check whether each of these faults is present or not.

The first one is normally known as functional testing which does not require the design information but only the correct functional behaviours of devices. There is nothing fundamentally difficult in the functional testing of digital system other than the volume of test that needs to be performed in order to achieve confidence in system [Hurst98a].

One simple example will give us an intuitive feeling of what could be the cost of an

exhaustive test. Imagine an arithmetic 16 bits accumulator, with 16 primary inputs and 16 internal latches to record two 16-bit numbers to be added, plus three control inputs and 16-bit output lines. The number of all possible input states of this simple device is $2^{16} \times 2^{3} \times 2^{16}$. Assuming the tester operates at the speed of 10MHz, i.e. each test cycle takes roughly 0.1 ms, it will take 57 minutes to complete the whole test. One might be able to afford this amount of time on a single device, but the problem becomes exceedingly hard as circuit size escalates and the number of tests increases exponentially. To test thousands of real products in this manner would simply paralyze the production line by having a huge pile-up of devices waiting to pass the tester.

The second approach relies on fault modelling. The potential advantages of using fault modelling for test purposes over functional testing is that the number of test patterns (combinations of logic inputs values) can be significantly reduced. This is aided by the fact that a test for one potential fault will often also test for other faults. [Wilkins86] [Hurst98a].

Common fault models include Stuck-at fault, Bridging fault, Stuck-open fault, Stuck-on fault, Transition delay fault, Path delay fault, *etc.* More details of these fault models will be introduced in Section 2.3. This section only gives a brief look at the stuck-at fault model as being the most typical and widely used one. A line is said to be stuck-at-0 (SA0) or stuck-at-1 (SA1) if the line remains fixed at a low or high voltage level respectively regardless of the input status. The stuck-at model is popular due to its simplicity, and because it has proved to be effective both in providing test coverage

[McCluskey03] and diagnosing a range of faulty behaviours. As an abstract representation of a class of defects, the stuck-at fault is commonly used to represent the defect of a circuit node shorted to either power or ground. Figure 2.9 illustrates how a real physical defect is represented by the stuck-at-0 fault model.

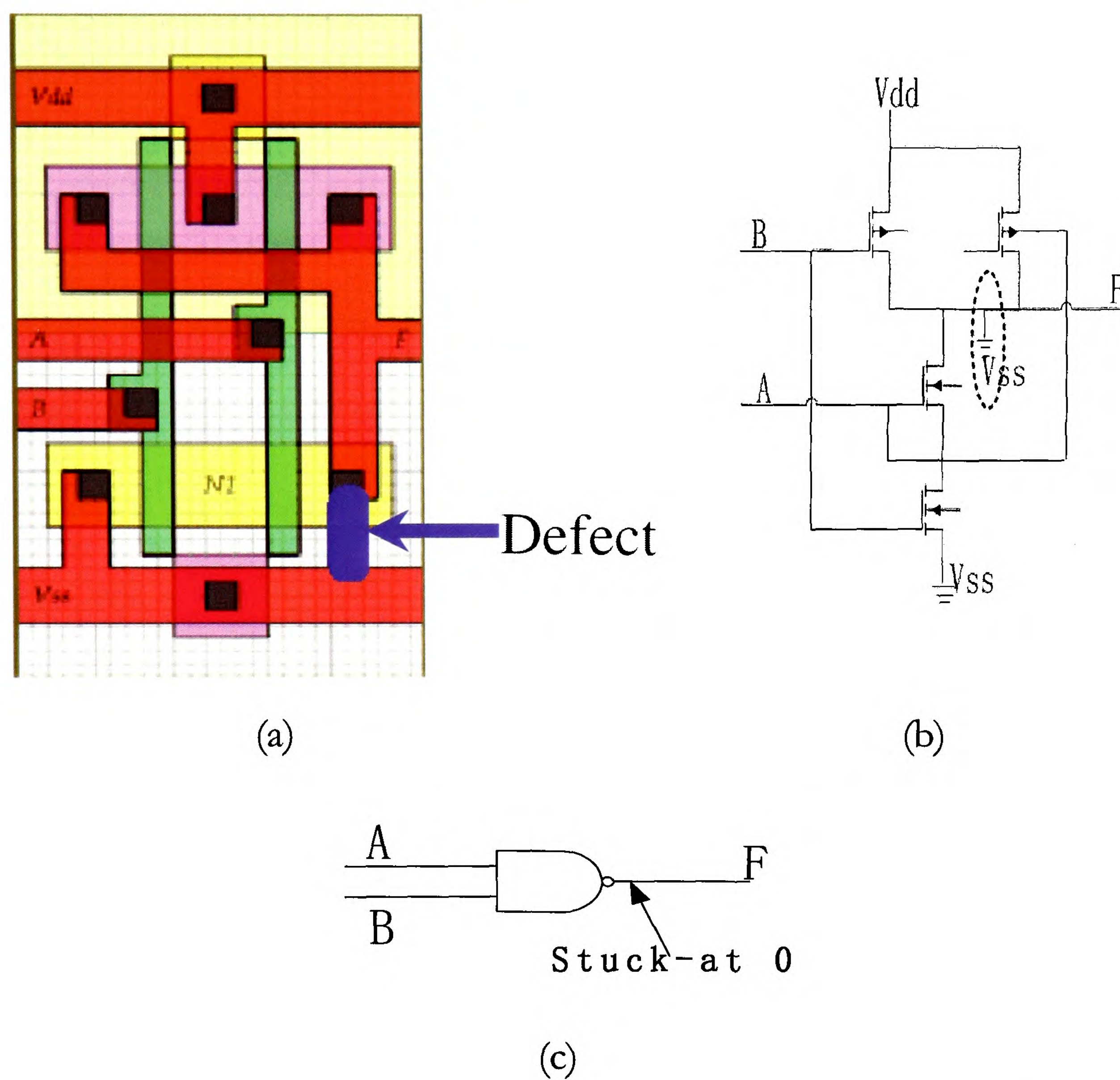


Figure 2.9 A physical level example of stuck at fault

Figure 2.9.a is the layout design of a simple NAND gate, the spot on the right bottom corner is a defect caused by extra metal material, it conducts the line F with the ground line (Vss), holding the voltage of line F consistently below the logic threshold. This defect can be symbolized in the transistor level schematics, as illustrated in Figure 2.9.b.

The defect is further abstracted to the gate level as a stuck-at-0 fault on the output of NAND gate, Figure 2.9.c. A stuck-at fault does not necessarily imply that the line is shorted to ground or power line. It could be a model for many other opens and shorts defects. For example, an open connection at the gate of a nMOS transistor in the presence of hot-holes in the gate oxide, or other positive charge that could turn the transistor permanently on, can be modelled as a stuck-at-1 fault [Timoc83].

2.2.4.2. Test Pattern Generation

This thesis will only address the problem of IC testing in combinational logic. A combinational logic circuit is the one has no dynamic elements like flip-flops and latches. While nearly all large-scale modern circuits are sequential, most of them are tested in such a way that their operation modes are transformed from sequential to combinational [Wilkins86] [Abramovici90] [Mourad00] [Jha03]. This is because test generation and fault diagnosis problems are much more complicated when sequential circuits are considered. Normally, to sensitize and propagate a single fault in sequential circuits requires several test patterns, but in the case of combinational circuits, one or two test patterns are enough. This thesis will come back to the issue of how to transform a sequential circuit to combinational circuit in the next section.

There are several methods available for deriving tests for combinational circuits like D-Algorithm [Roth66], PODEM [Goel81] and FAN [Fujiwara83]. It is not possible to explain much detail of these test pattern generation algorithms in this thesis. The

following example briefly shows the basic idea of path sensitizing in generating test patterns for a particular fault.

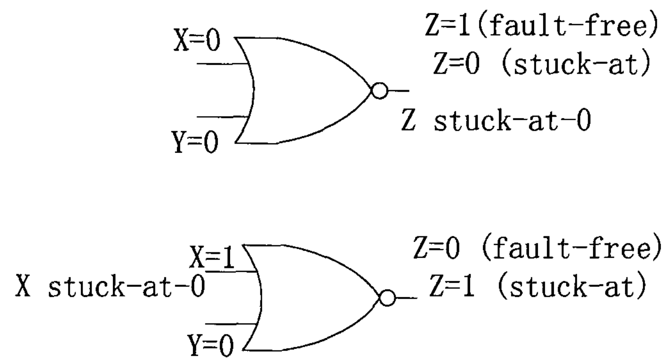


Figure 2.10 Test patterns for NOR gate

Figure 2.10 shows two scenarios of stuck-at-0 fault at a simple NOR gate. In the first scenario a stuck-at-0 fault occurs at gate output Z. In order to see the faulty effect of the stuck-at-0 fault, inputs must be set as XY=00. This is the only input combination that would set output Z to a different value of the one in the fault-free case (the fault is sensitized), so that when Z stuck-at-0 is present a faulty value is observed at the output Z. Likewise for the next scenario where input net X is stuck-at-0, the only combination that would allow us to see a different output value from the fault free one is XY=10, in which case the fault is propagated through this gate.

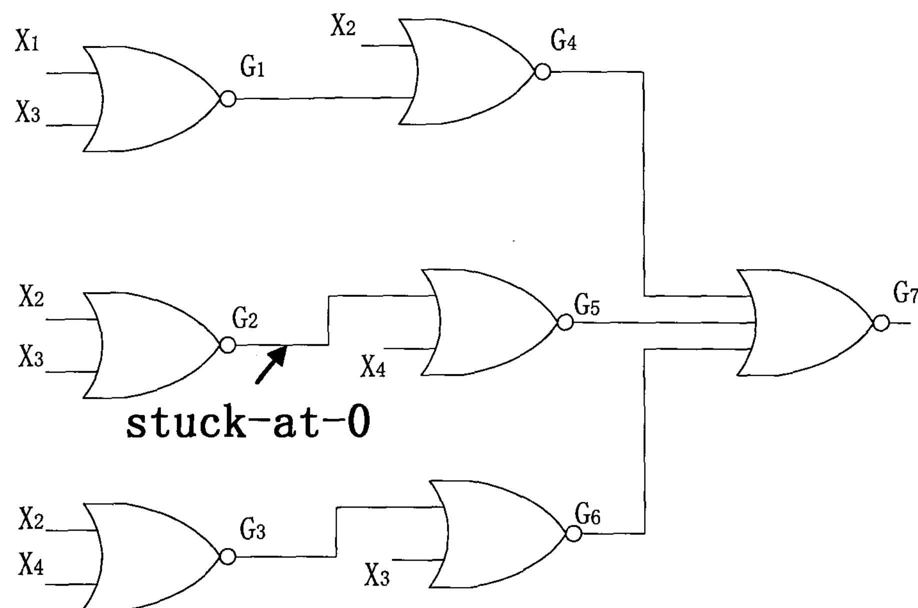


Figure 2.11 Test pattern generation for stuck-at-0 fault

Figure 2.11 shows a more complicated circuit, where net G_2 is stuck-at-0. To derive a test pattern for this fault, it has to be propagated to the output through the only path G_2 - G_5 - G_7. To sensitize the fault, as said early on, X_2X_3=00. To make sure the fault is then propagated to G_5, X_4 must also be set to 0. To further propagate it through G_7, the other two inputs of G_7 also have to be set to 0, i.e. G_4G_6=00. Because X_2, X_4 have already been set to 0, G_3=1 and consequently G_6=0. To make sure G_4=0, under the condition that X_2=0, G_1 must be 1 and consequently X_1=0. Therefore the only one possible test pattern generated for fault G_2 stuck-at-0 is $X_1X_2X_3X_4$=0000.

The real test pattern generation problem is much more complicated than what is illustrated in the above example. Issues like fanout nets and reconvergence can lead to redundancy in the circuit thus some faults being undetectable [Wilkins86] [Hurst98a]. What matters in real practice is the fault coverage which indicates how many of the faults in the required fault list are effectively tested by a given test pattern set. Figure 2.12 illustrates a basic flow of ATPG (Automatic Test Pattern Generation) to reach a fault coverage target. Each of the first few test patterns is likely to cover a number of faults, so the fault coverage will increase rapidly during the initial test pattern generation stage. ATPG algorithms provide a mechanism to generate a test pattern for a particular fault and fault simulation algorithms allow us to determine if any additional faults are covered by this test pattern. However the processing time to find the tests for the final remaining faults could be very long, particularly when feedback loops or redundancy structures exist. Therefore, a balance should be kept between the

fault coverage quality and the time ATPG spends. A small compromise on fault coverage may reduce the test pattern generation time significantly.

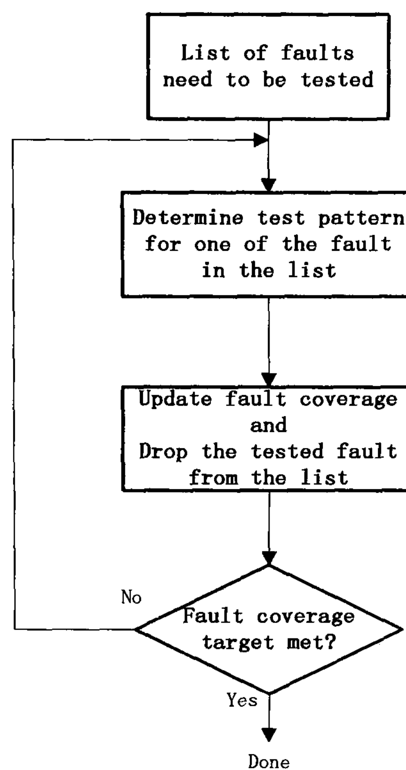


Figure 2.12 The main flow of ATPG delivering a fault coverage target

2.2.5. Design for Testability

2.2.5.1.Scan Design

The transformation from sequential circuits to combinational circuits is usually accomplished by implementing a *scan-based test* [Wilkins86] [Abramovici90], in which all state-holding flip-flops in the circuit are modified so that they can be controlled and observed by shifting data through one or more scan chains. Figure 2.13 illustrates a basic example of scan-based test.

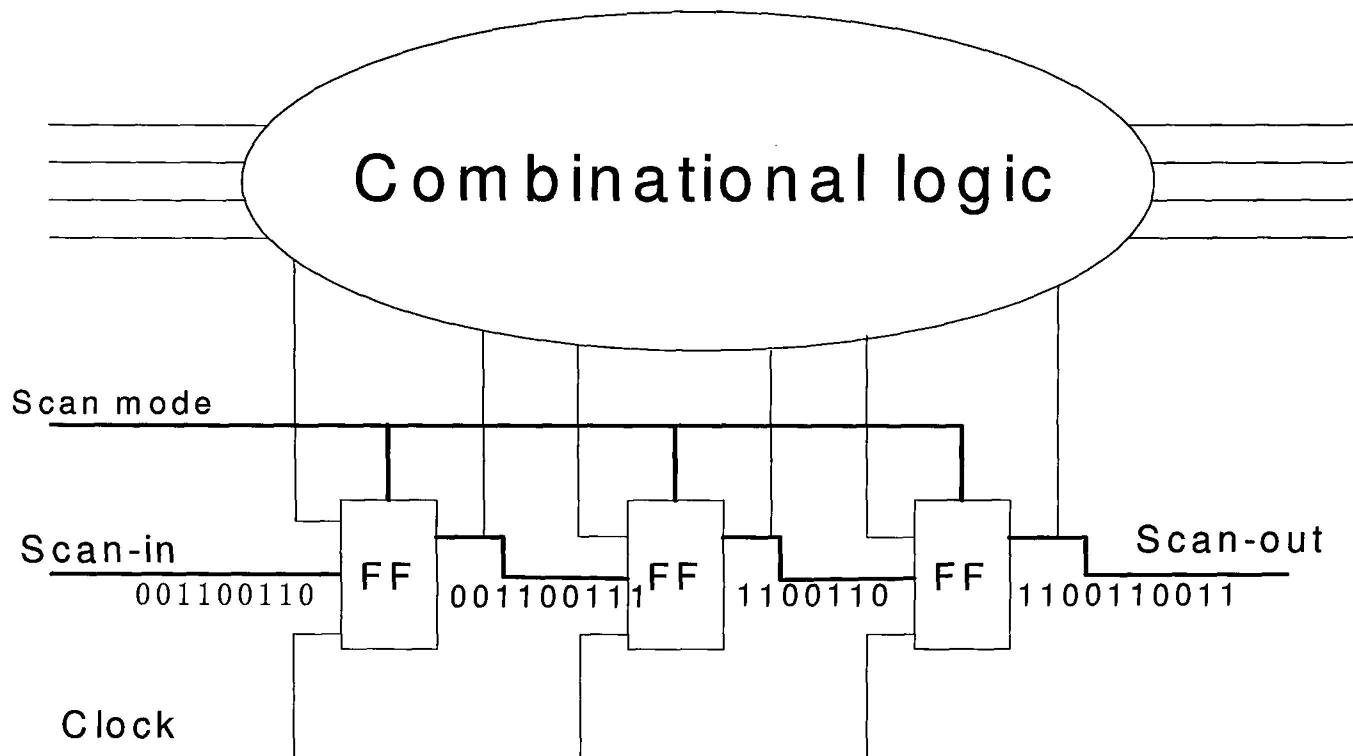


Figure 2.13 A Basic structure of scan based test

The “Test mode” signal is used to control the circuit to work either in the normal mode or in the test mode. While working in the normal mode, the scan elements work exactly the same as ordinary flip-flops. Once in the test mode, the scan chain first shifts in the test pattern by one bit per clock tick, when the whole pattern is installed, the combinational circuits can be tested and then scan chain can be used to shift out the response of the combinational circuits in the same way.

In the past few years, structured scan-based methods, most often full-scan, are being increasingly used to generate test patterns that are capable of achieving high coverage [Abramovici90] [Lee92], *i.e.* to make sure as many defects are tested as possible. A circuit is referred to as a full-scan design when all the memory elements in a sequential circuit are stitched together to form scan chains [Abramovici90] [Fujiwara90]. The advantage lies in the fact that scan-based tests provide us with higher controllability and

observability of the combinational circuits. The controllability problem is whether it is possible to find an input pattern that assigns a certain logic value on a certain net [Fujiwara90], in other words, to control the internal states of a combinational circuit. The observability problem is to decide whether there is an input pattern that propagates the logical value on a specified signal line to a primary output of the circuit [Fujiwara90], in other words, to observe the internal states of a combinational circuit. Designers often make use of multiple parallel scan chains to reduce the test application time. Scan based test has been widely adopted by the industry, it has changed the test of circuits from the functional view to the structural view.

2.2.5.2.BIST

BIST (Build-In Self-Test) refers to techniques and circuit configurations that allow ICs to test themselves. A basic BIST design is shown in Figure 2.14.

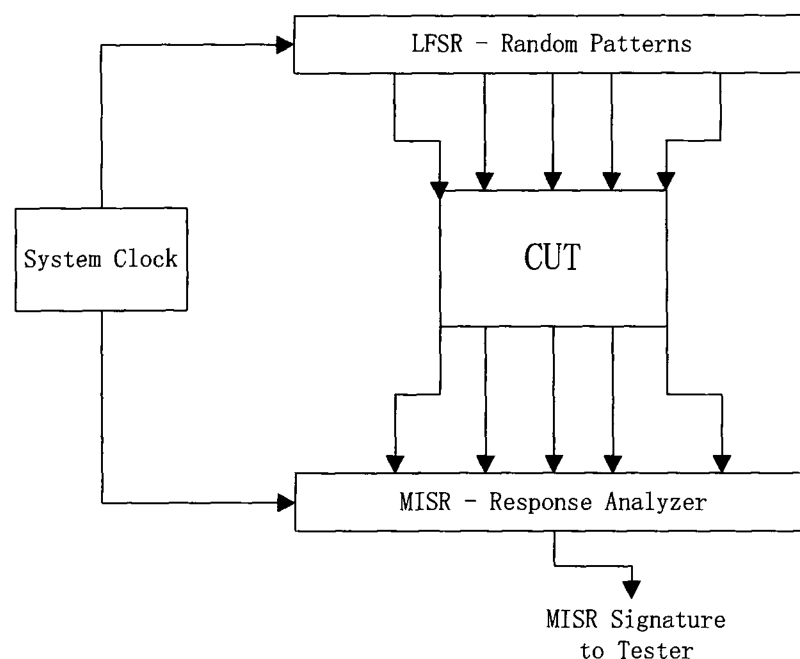
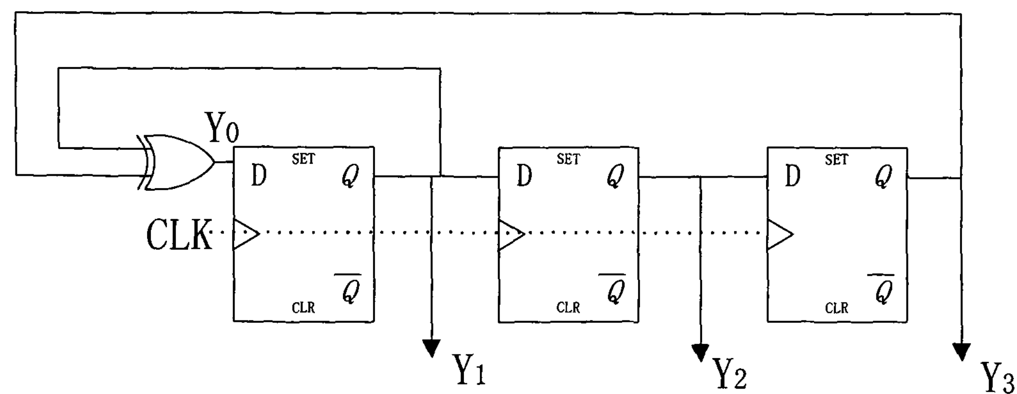


Figure 2.14 Built-In Self-Test circuit configuration

A built-in test pattern generator called a pseudorandom number generator is constructed by a linear feedback shift register – LFSR, which drives the circuit under test (CUT)

with pseudorandom combinations of the patterns [Segura04] [Mourad00]. The output of the CUT feeds into the response analyzer MISR (Multiple Input Signature Register) and the results are compared with the pre-computed signature to determine if there is a fault or not.

The mathematical foundation of the LFSR lies in the relationship between the LFSR sequence and feedback polynomial [Jha03]. Figure 2.15 shows an example of a standard LFSR configuration and the pattern sequence it generates by feeding back the values of Y_1 and Y_3 into an XOR gate connecting to the input of the LFSR chain.



CLK	Y0	Y1	Y2	Y3
0	1	0	0	1
1	1	1	0	0
2	1	1	1	0
3	0	1	1	1
4	1	0	1	1
5	0	1	0	1
6	0	0	1	0
7	1	0	0	1

Figure 2.15 Example of standard LFSR configuration

The LFSR is arbitrarily initialized to $Y_1Y_2Y_3 = 001$ at CLK = 0, $Y_0 = 1$ is a result of

Y_1 XORed with Y_3. Therefore in the next CLK cycle, every bit of the output value is passed on to the next D flip-flop, $Y_1Y_2Y_3$ thus become 100, and $Y_0 = 1$ consequently. The same procedure is repeated seven times until the LFSR returns to the initial status $Y_1Y_2Y_3 = 001$ at CLK = 7. Note that this standard LFSR configuration generates all the possible combination for a three input circuit except $Y_1Y_2Y_3=000$, since an all zero status will result in a static loop. This is an example of maximal length sequence for a LFSR chain, which has the maximal length sequence of $2^N - 1$, N being the number of the flip-flops in the LFSR (N=3 for this example). The way LFSR feeds back to the input determines if the characteristic polynomial associated with the LFSR chain is primitive and consequently if it can generate a maximal length sequence [Mourad00] [Jha03].

It is not generally a good idea to compare the CUT response of every pattern generated in BIST with the fault-free response, as this would require a large on-chip memory to store the complete fault-free responses. To avoid this extra memory cost, different compaction techniques, like parity testing, one counting testing, transition testing and MISR, are used to compact the responses of n patterns into one and then compare it with the pre-computed compacted fault-free response [Mourad00]. Among them the most popular one is the MISR which again is based on LFSR, modified to allow multiple inputs to be taken and compacted into a final signature.

BIST has several advantages over testing using automatic test equipment. First, BIST

provides an alternative solution when the cost of test pattern generation and the volume of data keep increasing with the circuit size. Secondly, because BIST is incorporated in the chip the testing can be executed at high speed. It is also very useful for cutting edge chip testing. It is very expensive to supply ATE that can keep pace with latest chip speed requirement but BIST can always be performed at the normal chip function speed.

However, detecting defects in IC devices is not the end of the story, people are also often interested in why these IC devices fails, which incurs the need for electrically locating and analyzing these defects, *i.e.* fault diagnosis. The next section will focus on this topic.

2.3. IC Diagnosis in general

2.3.1. Introduction

Historically, IC failure analysis practice has been a purely physical process (Appendix A). Even some of present-day FA teams still use only physical methods to investigate chip failures. The problem of pure physical failure analysis lies in the fact that every net and gate in the circuit is a possible suspect. The growing size and the number of layers of the modern IC devices means that it is often impossible for FA engineers to locate failures solely using physical measures. It is the responsibility of the other

part of failure analysis – electrical fault diagnosis, to do the guidance work. The purpose of electrical fault diagnosis is to logically analyze the information from test data and produce a list of likely fault candidates. This list has to be as short and accurate as possible because it could take only one misguidance for the subsequent FA process to destroy the real defect along with the chip.

2.3.2. Diagnosis algorithms

2.3.2.1.Basic diagnosis structure and terminologies

As explained in section 2.2.5, by implementing a scan-based test, all dynamic elements (flip-flops), can be converted to data input elements. Therefore the circuits connected to these flip-flops can be treated and tested as purely combinational circuits. The input data applied on the combinational circuit through scan chains during each scan operation is called *test pattern* or *test vector*. The operation of applying one test pattern and collecting corresponding output values is called a *test*. Automatic Test Equipment (ATE) is usually referred to as a *tester*, and is able to apply tests on either wafers or packaged dies. The tester records the actual responses measured through the scan chains and records any differences between the observed responses and expected fault-free responses. Unlike the usual manufacturing test setting where the tester stops at the first failure, the electrical fault diagnosis setting requires a full test data log of all mismatched responses.

The response of the faulty device to test sets is referred to as the *observed behaviour* or *observed fault signature*. In this thesis, it is defined as a set called *Obs*. The logic simulation is used to provide information about what failures there would be in the presence of a particular fault. This information is referred to as the *simulated fault signature* and is defined in this thesis as *Sim*. The fault signatures of both *Obs* and *Sim* are stored in the format of a number of failing bits-{Test vector : Failed output pin number}. For example, Table 2.1 shows a test scenario in which nine tests are conducted and records of output pins from 1 to 7 are given. Each Fail in the table indicates the output pin which fails to give the correct response while each Pass indicates otherwise. The set *Obs* for this test scenario will be recorded as {#1:4, #1:5, #2:2, #2:4, #3:1, #3:4, #3:6, #4:5, #5:2, #5:7, #8:1, #8:2, #9:2}.

Output Pins / Test Vectors	1	2	3	4	5	6	7
#1	Pass	Pass	Pass	Fail	Fail	Pass	Pass
#2	Pass	Fail	Pass	Fail	Pass	Pass	Pass
#3	Fail	Pass	Pass	Fail	Pass	Fail	Pass
#4	Pass	Pass	Pass	Pass	Fail	Pass	Pass
#5	Pass	Fail	Pass	Pass	Pass	Pass	Fail
#6	Pass	Pass	Pass	Pass	Pass	Pass	Pass
#7	Pass	Pass	Pass	Pass	Pass	Pass	Pass
#8	Fail	Fail	Pass	Pass	Pass	Pass	Pass
#9	Pass	Fail	Pass	Pass	Pass	Pass	Pass

Table 2.1 Example test scenario

Likewise, the set *Sim* for storing the simulated fault signature is written in the same format. The diagnosis process basically proposes possible faults, generates the

corresponding set *Sim* and tries to identify the closeness between *Sim* and *Obs*. Ideally the diagnosis engineer wants to see there is a *Sim* which matches with *Obs* exactly because the closer *Sim* is related to *Obs*, the more likely that the assumed fault is the real fault that has occurred in the circuit. However, for real test results and current fault models, having identical *Sim* and *Obs* is often not achievable. Therefore, different algorithms have been developed to match their relationship. The next section will give a brief review of the Matching algorithms developed and also the Matching and Prediction system [Hora02] that this thesis uses.

2.3.2.2. Diagnosis algorithms

As explained in section 2.2.4, the stuck-at fault model is the simplest and earliest model which enjoys popularity and effectiveness in most aspects of IC testing such as pattern generation, fault simulation, *etc.* It is very natural that early diagnosis algorithms adopted the simple stuck-at fault model, expecting that the simulated fault signature from one of the assumed stuck-at fault will match exactly with the observed fault signature. However it soon became obvious to the testing community that most failures do not exactly follow this expectation. Although the stuck-at model is efficient and effective in many aspects of test, often the fault mechanism does not accurately follow the stuck-at fault model. Therefore diagnosis algorithms were developed by increasing the sophistication of the matching to account for these unmodelled behaviour as described below. The DORA system developed by Allen [Allen92] uses a

nearness algorithm called a fuzzy match. The idea of the fuzzy match is to give some tolerance over the simulated fault signature, allowing for some of the observed results not to match exactly with the simulated fault signature. A similar idea from [Ratford86] further employed the concept of prediction penalties. Each simulated fault signature, from each assumed fault, is matched with the observed faulty behaviour. This algorithm scores each candidate by penalizing each failing bit that has been predicted by simulation but not found in the observed fault signature, as well as for each failing bit in the observed fault signature but not predicted by the simulations. The first part of this penalty is called Mis-prediction and the second part of this penalty is called Non-prediction.

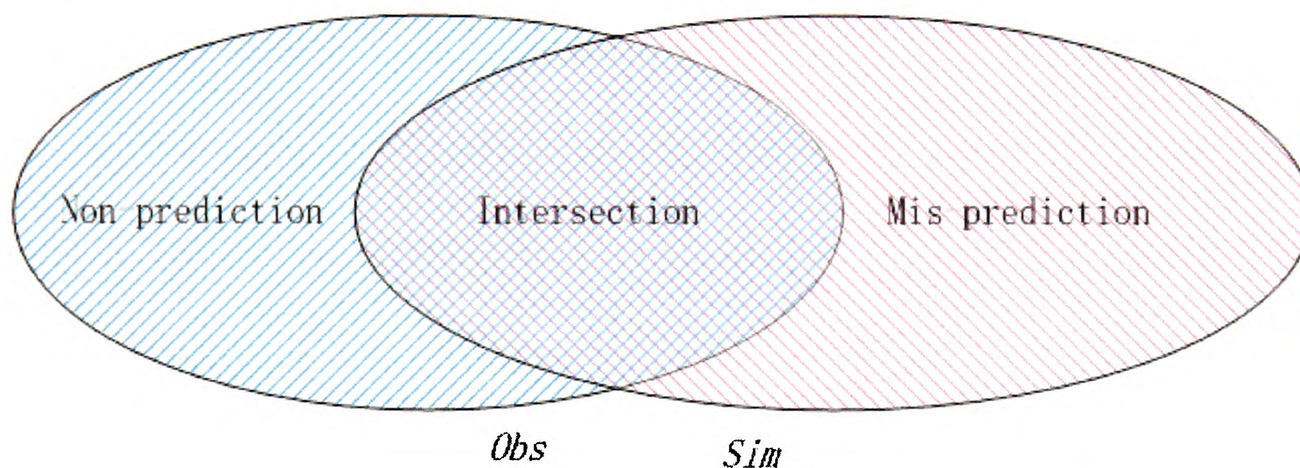


Figure 2.16 Non-prediction and Mis-prediction

In Figure 2.16, the set of observed fault signatures are represented by the ellipse on the left and the set of simulated fault signatures by the ellipse on the right. The Mis-prediction part is the behaviour that happens in the simulation of a certain fault candidate, but is not observed on the tester. Likewise, the Non-prediction is what has been seen on the tester but not predicted by the simulation.

An algorithm introduced by [Kunda93] ranks the candidate by the size of the intersection between the *Sim* and *Obs*. By doing this, it implicitly focuses on the Non-prediction penalty while ignores the Mis-prediction penalty. This is based on the intuitive assumption that most defects involve a single fault site and behave intermittently like a stuck-at fault. Therefore the simulation of this particular stuck-at fault usually leads to an over-prediction, *i.e.* the stuck-at fault model is not perfect but any unmodelled behaviour will only cause fewer failing bits in the simulated fault signature. This turns out to be a poor assumption to make and therefore a more sophisticated algorithm was developed by De [De95] in which the author provides the flexibility to adjust weightings to mis-prediction and non-prediction. Each fault is scored with a *merit* value. For each fault *f*, the number of failing bits of simulated stuck-at fault signature that have been observed on the tester is counted as Detected(*f*), whereas the number of failing bits of simulated stuck-at fault signature that are not observed on the tester is counted as NoDetected(*f*). The merit value is then given by:

$$Merit(f) = C_1(NFO - Detected(f)) + C_2 NoDetected(f)$$

NFO is the number of failing bits in the observed fault signature. By choosing the parameters C_1 and C_2, users can specify the importance of matching non-prediction part and mis-prediction part. The authors claim the method can be used to explicitly target defects that behave similar to but not exactly like the stuck-at model, such as some opens and multiple independent stuck-at faults.

When addressing bridging fault diagnosis, Lavo [Lavo96] proposed a new matching

algorithm specially for bridging fault diagnosis. Fault signatures of bridging faults are composed from the stuck-at fault signatures of the two nodes unexpectedly connected (the electrical model of bridging defects). Instead of combining all the stuck-at signatures together, the authors used selective measures to prune out those failing bits of the stuck-at simulated fault signature that put the same value on both of the bridged nodes. This makes sense because the nodes with the same value do not compete with each other and thus will behave in a fault-free way. Some of the failing bits have more importance than the others thus are defined as *required vectors*. They are the failing bits that detect stuck-at-0 for one bridged node and stuck-at-1 for the other. The faulty effect of these failing bits would have been guaranteed if the assumed bridging fault is true. Naturally, it becomes more damaging to the assumed bridging fault when not observing the *required vectors* than not observing the others. Figure 2.17 illustrates the relationship between the simulated fault signature and observed fault signature. The part of the signature that is contained in both signature sets is defined as intersection. The upper part of Figure 2.17 shows mis-prediction and non-prediction and is equivalent to Figure 2.16. The lower part of the Figure 2.17 shows four different assumed bridging faults' test results. The stripe named *R* indicates one *required vector*.

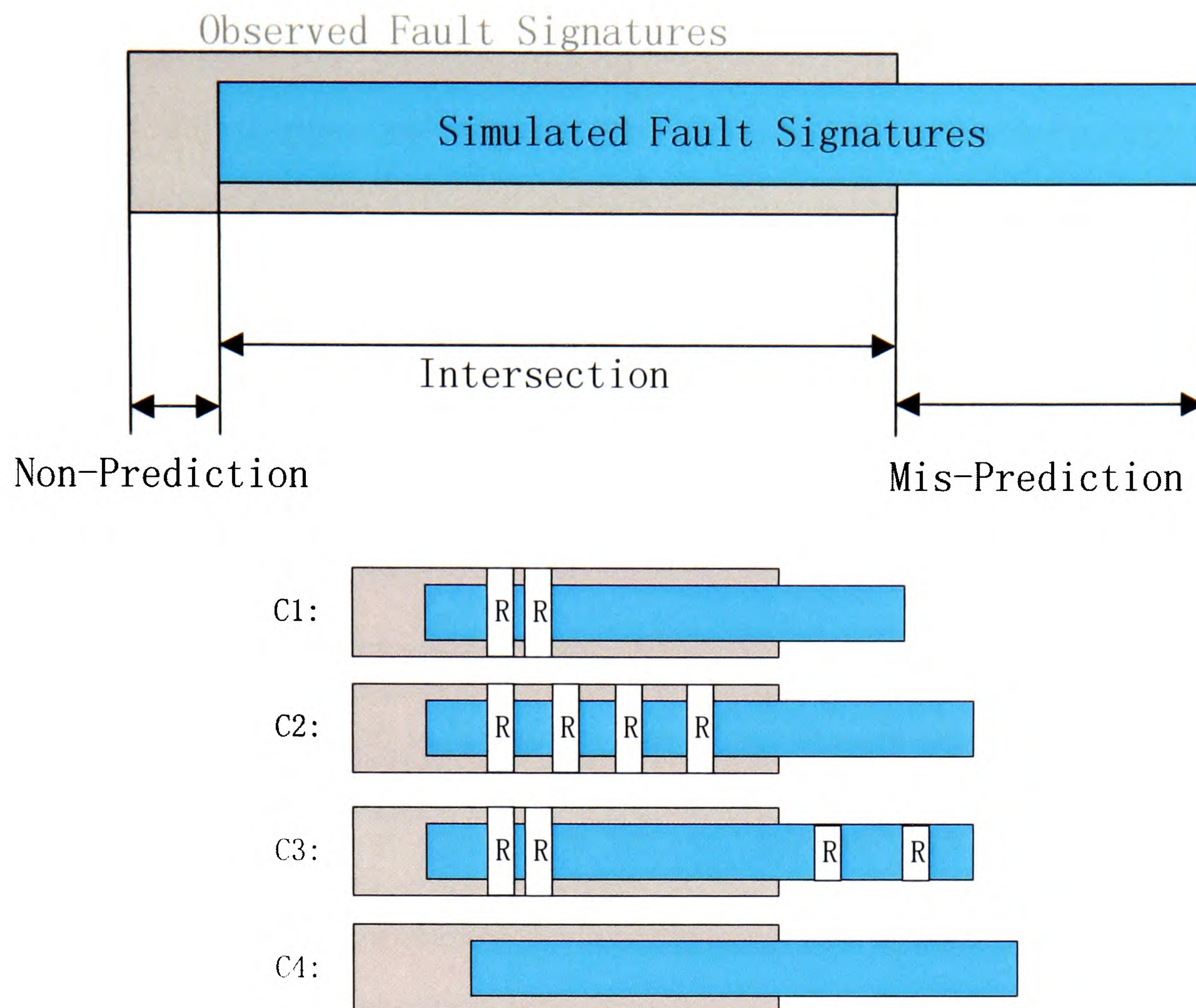


Figure 2.17 Matching algorithm of [Lavo96]

Lavo ranks the intersection part as the top priority. The candidate with a larger intersection part is the favourite. Clearly candidates C1, C2 and C3 win over C4 because of their size of intersection part. If there is a tie in the intersection size, as between C1, C2 and C3, the candidate with higher percentage of *required vectors* appearing in the intersection part wins. C1 and C2 beat C3 in this regard. Finally, if there is still a tie, the mis-prediction part is used to break the tie. Candidate C1 ranks at the top because of its smaller mis-prediction part in comparison with C2.

Hora's Matching and Prediction system:

The most recently developed matching algorithm was published by [Hora02], in which a Matching and Prediction system is proposed. One of the major advantages of this new

system is that with different scenarios of diagnosis results, one can identify a category of possible defect types and thus be able to take further action to make a more convincing judgment. Hora02 follows up the diagnosis with a number of FA results which provide convincing evidence of the accuracy of the approach. This Matching and Prediction system will be used as the backbone diagnosis algorithm of this thesis. Therefore a detailed explanation follows.

First, let us assume the following data. A set of test vectors are applied and the failed output pins (failing bits) are recorded as the set of *Obs* in Figure 2.18(a). As explained above, the first digit corresponds to the vector number, the second digit means the failed output pin number. For instance, "#1:3" means pin 3 failed when vector #1 is applied. Suppose one net has a stuck-at fault and it is simulated to record the failing bits in the set of *Sim* in Figure 2.18(a). As "#1:3" and "#2:5" are the common failing bits of *Obs* and *Sim*, so the figure looks like Figure 2.18(b), which illustrates the intersection part of the two sets.

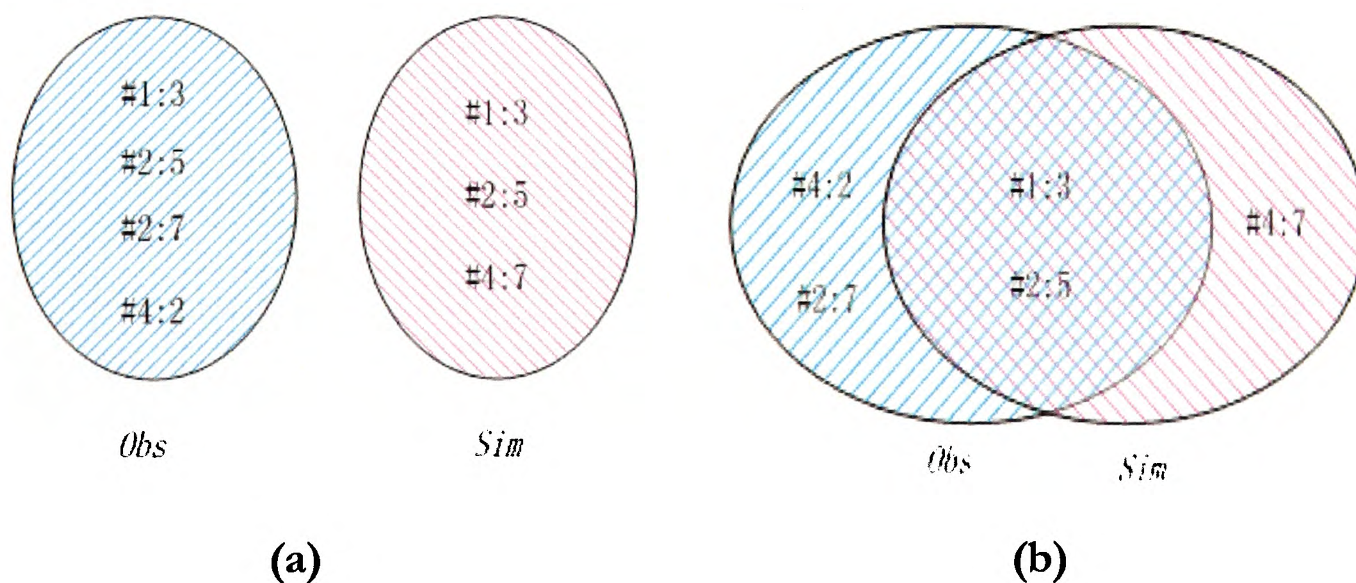


Figure 2.18 An example of *Obs* and *Sim*

Hora [Hora02] has summarized this idea by defining two measurements, Matching and Prediction, to order the likelihood of the suspected nets. The term "Matching" is used to quantify the extent to which the observed failing results actually match with the simulation of a particular fault.

$$Matching(M) = \frac{Obs \cap Sim}{Obs} \times 100\%$$

A second term "Prediction" is used to quantify the extent to which simulations of a particular fault predict only those failing observed results.

$$Prediction\ (P) = \frac{Obs \cap Sim}{Sim} \times 100\%$$

For the example in Figure 2.18, Matching = $\frac{2}{4} \times 100\% = 50\%$ and Prediction = $\frac{2}{3} \times 100\% = 67\%$. All the possible relations of the set *Obs* and *Sim* are concluded as seen in Figure 2.19.

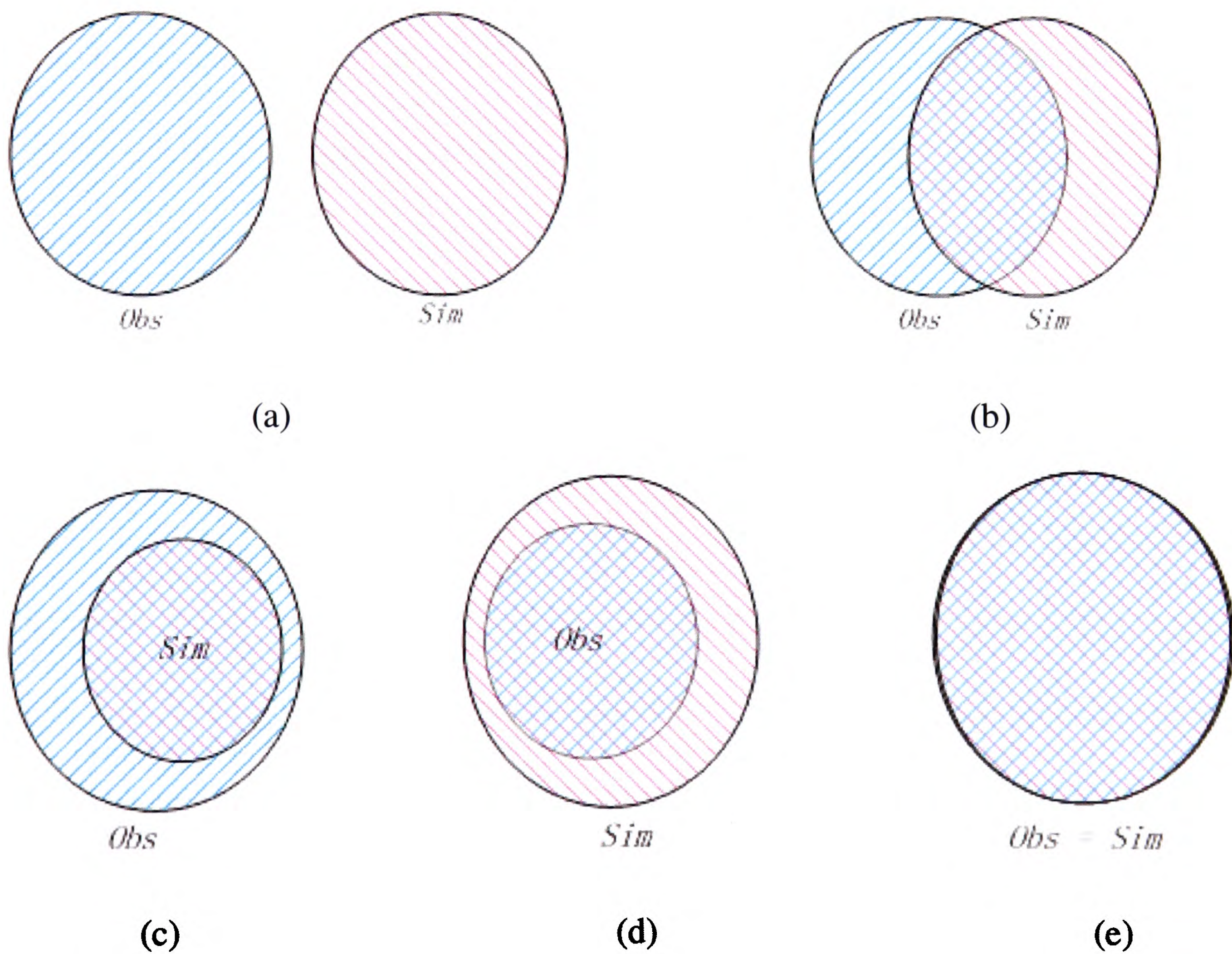


Figure 2.19 Possible relations of *Obs* and *Sim*

	Matching	Prediction
4.4.a	0%	0%
4.4.b	<100%	<100%
4.4.c	<100%	100%
4.4.d	100%	<100%
4.4.e	100%	100%

Table 2.2 Matching and Prediction of Figure 2.19

Table 2.2 shows the corresponding Matching and Prediction for each situation in Figure 2.19. It is quite clear that the situation of a perfect diagnosis is 4.4.e, where the observed output exactly matches the simulated output, indicating the assumptions made before the simulation, the fault location, the fault model, *etc.*, are verified. However in real life, very often 4.4.e is not achievable. (This may also arise when the faults are within a gate, which will be further discussed in the remaining chapters of the thesis.) Given an observed fault signature, the algorithm has to try to find a candidate fault that can explain all the failures. Therefore the Matching result is considered to bear more significance than the Prediction. If two candidates have the same Matching results, the Prediction is referred to break the tie. This is actually in line with the previous work by Lavo [Lavo96], which weighs the intersection between *Obs* and *Sim* as the first priority and use Mis-prediction to break the tie should the other factors are the same. Hora's algorithm doesn't employ the idea of *required vectors* as it is used as a comprehensive matching algorithm for various types of faults rather than the bridging faults focused by [Lavo96]. The way Matching and Prediction are presented facilitates further analysis by indicating different fault scenarios that cannot be diagnosed by the simple stuck-at fault model, as the following chapters will show.

2.3.3. Fault diagnosis with fault models

2.3.3.1.Preliminary shortlist

As discussed in the previous section, the electrical fault diagnosis mainly works under the framework of the cause-effect method, by comparing the observed fault signature and the simulated fault signature. However it is impossible to consider all the nets in modern complex IC circuits as suspects and do simulation for each net. Therefore preliminary work has to be done to shortlist the suspects to a manageable size for the simulation concern. The back-coning step, actually borrowed from the idea of effect-cause diagnosis method, is performed as the first step of the whole diagnosis process. For each failed output, the back-coning step identifies a group of nets that have direct propagation path to this particular output, also called the logic cone of an output. Figure 2.20 shows an example of a logic cone for a flip-flop output.

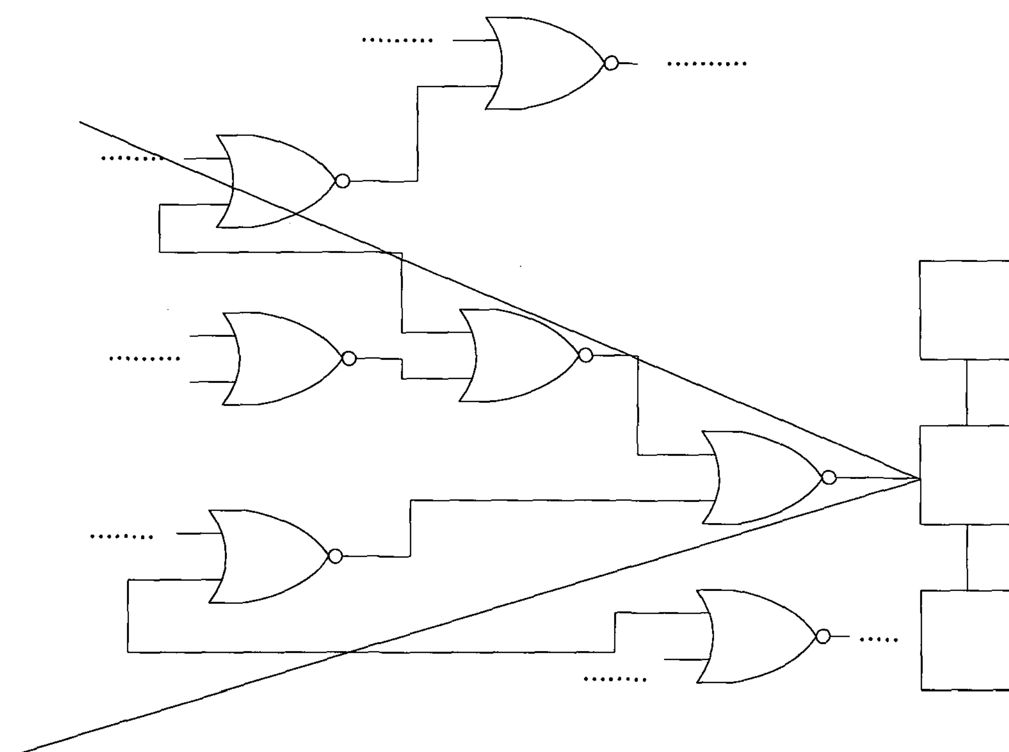


Figure 2.20 Logic cone

Clearly, if one can be sure that there is only one fault in the circuit, the back-coning searching space can be further reduced by intersecting all the logic cones as a mean to find the nets that have direct path through to all the failed output pins. This assumption can significantly reduce the searching space and consequently save simulation time and signature storing space, but could be potentially misleading when there are actually multiple faults in the circuits. Therefore a conservative approach is to take the union of all the logic cones as the searching space.

When the list of back-coned nets is obtained, stuck-at fault simulation is performed on each net on the list. During the stuck-at simulation, a variable called *suspect* is assigned to each candidate on the list, recording the number of times that an output pin is failing on both the tester and in the simulation, in other words, the number of failing bits that appear in both *Obs* and *Sim*. The stuck-at faults with *suspect* = 0 after the simulation are dropped from the list and their stuck-at simulation information is discarded (they fall in the category of Figure 2.19.a). The stuck-at nets with non-zero *suspect* are considered as the *suspect faulty nets*. The stuck-at simulated fault signatures are stored in a format like the one introduced in Section 2.3.2. Also, it has to be mentioned that when it comes to a fan-out net, Figure 2.21, the fault signatures of each fan-out branches are recorded separately even though electrically they are connected. Defects like open connections on one of the fan-out branches could have effects that manifest themselves only on the particular branch while the other branches are still properly driven by the previous gate.

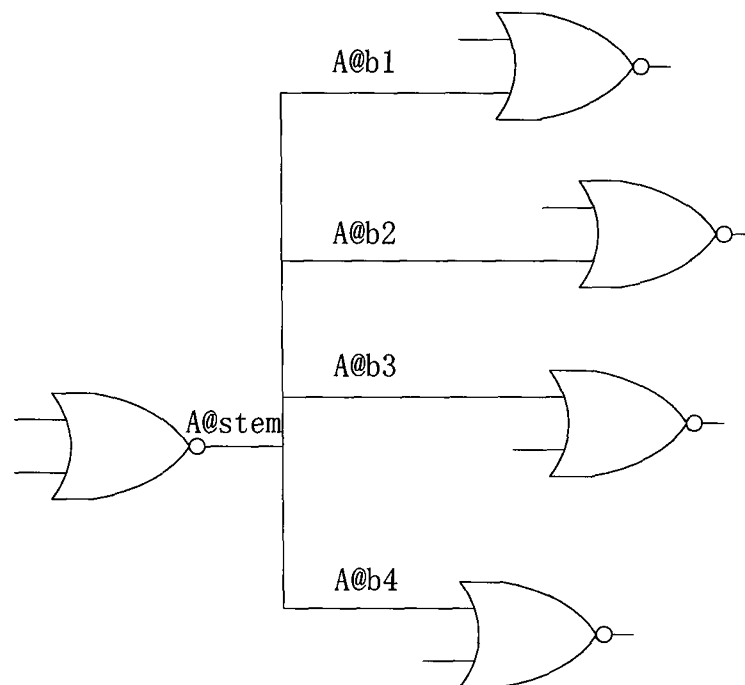


Figure 2.21 Fan-out net with four branches b1, b2, b3, b4.

Therefore it is realistic to assume a stuck-at fault on each fan-out branch and record its stuck-at simulation signatures respectively. Following is one example of how simulated fault signatures of the fan-out net in Figure 2.21 are recorded:

Net A@stem SA0 {#1:4, #1:5, #2:3, #3:5, 5:5, #6:2, #6:5, #6:8}

SA1{#1:6, #2:5, #2:8, #4:3}

Net A@b1 SA0 {#1:4, #3:5} SA1 {#1:6, #2:8}

Net A@b2 SA0 {#5:5, #6:2, #6:5} SA1 {#2:5}

Net A@b3 SA0 {#2:3} SA1{#4:3}

Net A@b4 SA0 {#1:5, #6:5, #6:8} SA1 {null}

2.3.3.2. Stuck-at fault diagnosis

Based on the most common stuck-at fault model, stuck-at fault diagnosis shortlists and ranks the candidates according to the similarities of their simulated fault signatures and

the observed fault signature. The Matching and Prediction rule applies. For each *suspect faulty net* the *Obs* is compared with both the SA0 *Sim* and SA1 *Sim* (or whichever is available) associated with the net as was described in the previous section. Stuck-at fault diagnosis looks for Matching = 100% and Prediction = 100%, *i.e.,* all the observed failing bits are explained by the assumed stuck-at fault with no predicted failing bits unobserved. The stuck-at fault has to be the optimal explanation for what goes wrong on the tester.

A cruel fact is that the variety of defect types that actually occur means that there is no model that can universally describe the behaviour of modern IC defects. Open cracks in the gate connections and inside the gates, unwanted connections between two nets caused by extra metal materials, *etc.*, can all cause fault mechanisms beyond the capability of the stuck-at fault model. (The next three sections will give more details of how to model these special defects.) As far as the diagnosis is concerned, they would give out non-100% results under the current Matching and Prediction system used here. Nevertheless, they provide a useful hint at the possible defect type [Hora02]. Given the different combination of Matching and Prediction values, some of the possible explanations can be found as shown in Table 2.3.

Matching	Prediction	*Obs* vs. *Sim*	Fault type
100%	100%	Figure 2.19.e	Stuck-at fault
100%	<100%	Figure 2.19.d	Bridging fault, or stuck-at faults on a few of the net's fan-out branches, or intra-gate open faults
<100%	100%	Figure 2.19.c	Multiple faults from which at least one is a stuck-at fault
<100%	<100%	Figure 2.19.b	Bridging fault or multiple faults.

Table 2.3 Indication of possible fault type

When the Matching and Prediction both give 100%, the *Obs* fits *Sim* exactly as in Figure 2.19.e. As discussed earlier, it is a successful stuck-at fault diagnosis. When Matching is 100% but Prediction is less than 100%, possible explanations include bridging fault, when the net is unexpectedly connected with another net and was affected from time to time when certain conditions allow, but not constantly like a stuck-at fault. In this case, only a part of the stuck-at simulated fault signatures actually shows in the *Obs*, as in Figure 2.19.d. Stuck-at faults at a few of the net's branches have the same effect of producing only part of the fault signatures of the stem net, thus having the same diagnosis results. Intra-gate open crack defects can cause the same situation and this will be fully addressed in Chapter 3.

Multiple faults can cause situations such as Prediction of 100% and Matching lower than 100%. Because of the existence of multiple faults, the *Obs* ought to be the union of

the *Sims* of these multiple faults. Clearly *Obs* contains *Sim* therefore leading to a situation like Figure 2.19.c.

When both Matching and Prediction fall below 100%, actually not much useful information can be inferred because too many complex defects scenarios could lead diagnosis results into this category. Possible guesses could be bridging faults with more complicated behaviour and multiple faults with some fault effects cancelled with each other.

Although the stuck-at based diagnosis is effective in diagnosing stuck-at like faults and is informative for other defect types, it has been universally accepted that a model correctly representing the effect of the real physical defect is extremely important to accurate diagnosis. Aitken [Aitken95] has concluded that a better matched fault model makes more sense than a sophisticated diagnosis algorithm. A number of papers [Greenstein92] [Millman90] [Lavo98] use specific bridging models to cope with the electrical short defects and gain a better diagnosis resolution than the general stuck-at model. All the factors converge to demonstrate the importance of an accurate fault model representation of the real physical defects. There are two main types of static fault models other than stuck-at fault model that are commonly used now: bridging model and open model and these will be explained below.

2.3.3.3. Bridging fault diagnosis

The bridging fault model is used to represent an electrical short between signal lines. It was first introduced by [Mei74]. When two nets are bridged and their driving values are in conflict, one of the bridged net holds a strong position and the other holds a weak position whose value is overruled by the strong net. Effects of this overruling have been assumed equal to an AND gate or an OR gate, known as the wired-AND and wired-OR bridging fault model. Another straightforward bridging fault model is the line dominant model, in which one of the shorted lines always drives the value of the other. These bridging fault models ignore bridge resistance and transistor parametric factors, and instead focus on the logical behaviour of the fault, thus they are easy to implement in simulation, test pattern generation and diagnosis. They have been widely practiced and proven to be effective in fault diagnosis [Hora02]. There are arguments that these models are idealized and simplified logic assumptions and may not be able to accurately reflect the real behaviour of some of the bridges in modern CMOS technologies [Timoc83] [Hawkins94]. A more sophisticated explanation of bridging defects is to calculate the voltage value of the bridged nodes, instead of assigning simple logical value of 1 or 0 to the nodes. The calculation of more accurate value of the bridged nodes depends on a number of factors, the relative driving strength of the pull-up and pull-down networks that are driving these two nodes, the value of the input patterns on the pull-up and pull-down networks and the resistance of the defect [Acken91].

It is worth mentioning that some special bridging situations may cause difficulties in modelling and diagnosing faults. The first complication is known as the Byzantine General's problem [Chess95], where the threshold voltages on which the following gates interprets the logical value differently. This has quite an influence on the model's accuracy.

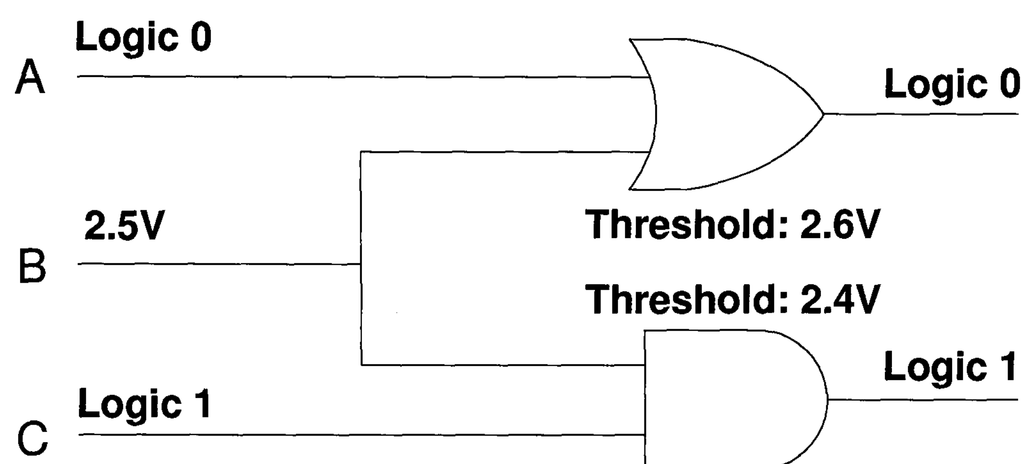
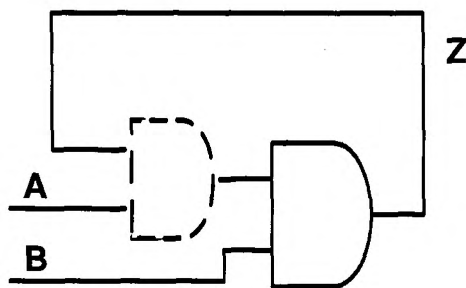


Figure 2.22 Byzantine General's Problem

Figure 2.22 illustrates the case where the voltage of the shorted node lies between the two threshold voltages that interpret it and is interpreted as different logic values. In [Acken91] a voting model is used and SPICE simulations performed to generate tables with the relative strength of driving networks consisting of serial and parallel connected transistors in the pull-up and pull-down networks. The gates driving the bridged nodes are SPICE simulated to determine the bridge voltage, which is then compared against the SPICE-computed logic thresholds of downstream gates to model the effect of the Byzantine General problem. Similar approaches are described in [Greenstein92], where SPICE-like simulation is conducted in the region within a certain distance around the fault location, combined with a digital simulation for the rest of the circuit.

Another problem in bridging fault modelling is that bridging defects can form a feedback connection and potentially transform a combinational circuit to a sequential circuit [Li02]. Consider the circuit in Figure 2.23 as an example. Node Z is bridged with node A and they take on the wired-AND bridging model. The truth table for this situation is listed alongside. Once the input pattern is A=1 and B=1, node Z keeps the previous value while the faulty free response will be 1. This shows that accurately modelling this type of defects requires not only the information mentioned above, but also the information of previous and even earlier test.



A	B	Z_{good}	Z_{fault}
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	Hold

Figure 2.23 Feedback bridging fault causing dynamic behaviour

The feedback bridging faults can lead to an even more daunting problem of oscillation if there are an odd number of inversions in the loop under a certain propagation condition [Konuk97].

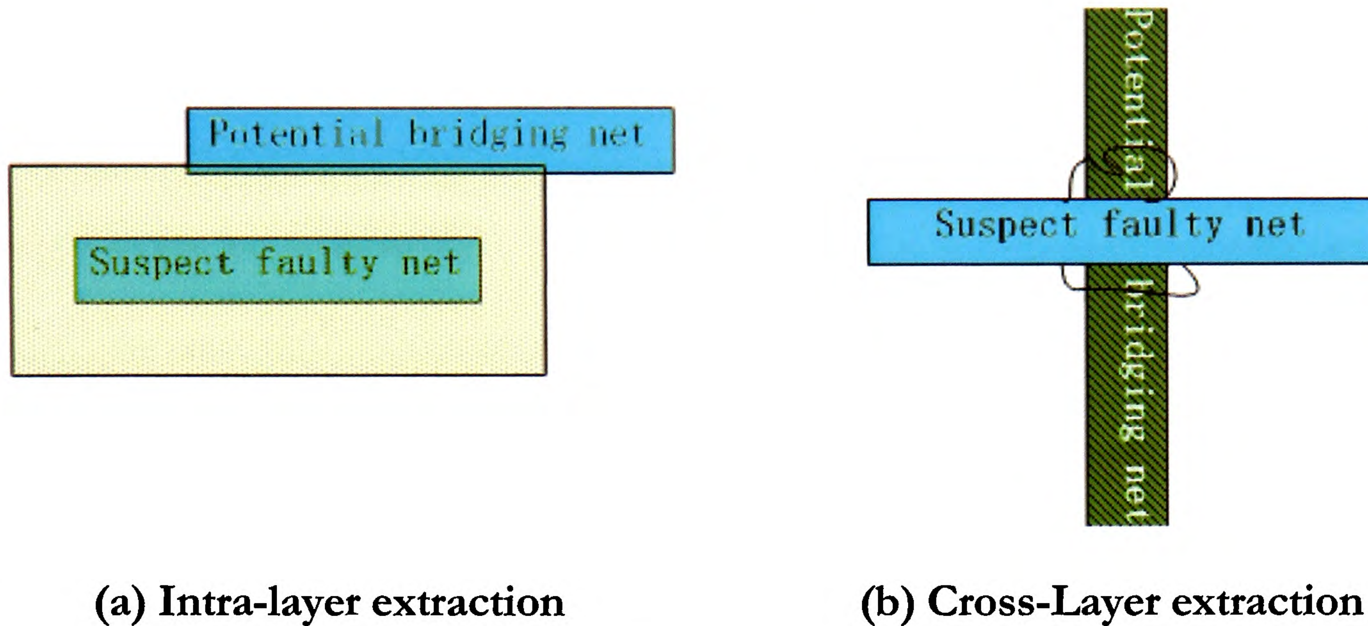
Bridging fault diagnosis algorithms

In the previous section, purely logic models like Wired-AND and Wire-OR and Line dominant model were introduced in juxtaposition with more sophisticated models which take into account of the specific bridged gates. Though the latter provide better

description of the effect of the bridging, they all rely heavily on the electrical level simulation, which is time consuming, and the correctness of this kind of simulation relies on the transistor parameters that are susceptible to process variations and hard to predict accurately [Chess94]. Besides this, the resistance of the bridge defects makes a significant difference and that the resistance is unpredictable [Sar-Desai99]. Therefore logic models have been more widely accepted and used as a cost-effective way to tackle the bridging fault diagnosis issue.

In a searching space of n nets, there are $\binom{n}{2}$ possible pairs of bridged nets, which is too many to handle even if the searching space can be reduced by back-coning steps as explained in Section 2.3.3.1. Also it is risky to assume that both bridged nets are among the back-coned *suspect faulty nets*. One can literally only be sure that the *victim net* is among the suspect faulty nets, while there are cases where the other bridged net always plays like *aggressor net* and never becomes a victim net, or when it gets affected, the effects never propagates to the outputs. If this worse case happens, the searching space cannot be simply limited to those n suspect faulty nets. To get a set of realistic and manageable bridging fault candidates, layout information has to be used to minimize the number of possible pairings of bridged nets [Ferguson88] [Jee93] [Stroud00]. Bridging faults happen a lot in the same layer and they are normally caused by extra conducting materials between two regions that are physically close to each other [Ferguson88]. Given the knowledge of searching space of at least one of the bridged net – the suspect faulty nets, inspections are performed on each suspect faulty net to check

those nets that are sufficiently close for a realistic bridging fault. Circular or rectangular regions centred at the suspect faulty net are assumed and the threshold distance within which a realistic bridging fault can occur is assigned, as shown in Figure 2.24.a.



(a) Intra-layer extraction **(b) Cross-Layer extraction**

Figure 2.24 Extraction of realistic bridging faults

Cross layer bridging can happen when the insulating oxide layer in between the two layers breaks down. This happens most often in the thin gate-oxide. A defect in the insulating layer could connect the suspect faulty net with potential bridging net in the underneath or the upper layer, as shown in Figure 2.24.b.

After extracting realistic bridging faults from the layout information, simulated fault signatures have to be built for all those bridging faults that have been extracted. The faulty signature of a specific bridging fault can actually be derived from the stuck-at signatures of the two bridged nets. The reason lies in this important observation: when two nets are bridged and a certain input vector is applied, one of the bridged nets will take on the aggressor position, affecting the other victim net. The effect of this fault thus has to be seen from the propagation through the victim net to the primary output.

This propagation is effectively the same as the propagation of a stuck-at fault on the victim net. When the input vector allows the stuck-at fault to propagate, the bridging faulty effect is seen at the output, otherwise, no faulty effect is observed. This leads to the conclusion that the failing bits of a bridging fault should be the same as the failing bits in one or other of the stuck-at fault signatures of the bridged nets. Therefore in the early works of bridging fault diagnosis [Millman90], a composite bridging fault signature was constructed from both stuck-at-0 and stuck-at-1 signatures on the two bridged nets. Suppose net A bridges with net B, the fault signatures for A**(bridge)**B is {SA0@A} ∪ {SA1@A} ∪ {SA0@B} ∪ {SA1@B}. This inevitably leads to a situation called *over-prediction*, because though every bridging fault signature should also be a stuck-at signature on either the bridged net, this is hardly true the other way around. The best diagnosis result one could achieve in the presence of over-prediction is like Figure 2.19.d.

Developments have been made to reduce the size of the fault signature of A**(bridge)**B by taking into account the value of the bridged nets under each stuck-at failing bit. Those bits which put the two bridged nets to the same values are dropped from the set of bridging fault signature [Lavo98] [Hora02]. Bridging models are used to further reduce the size of the faulty signature. Line dominant models specify one net as always being the aggressor, affecting the other bridged net when their values are at odds. Wired-AND, Wired-OR models describe the effect of bridging faults as a function of a logic gate, as shown in Figure 2.25.

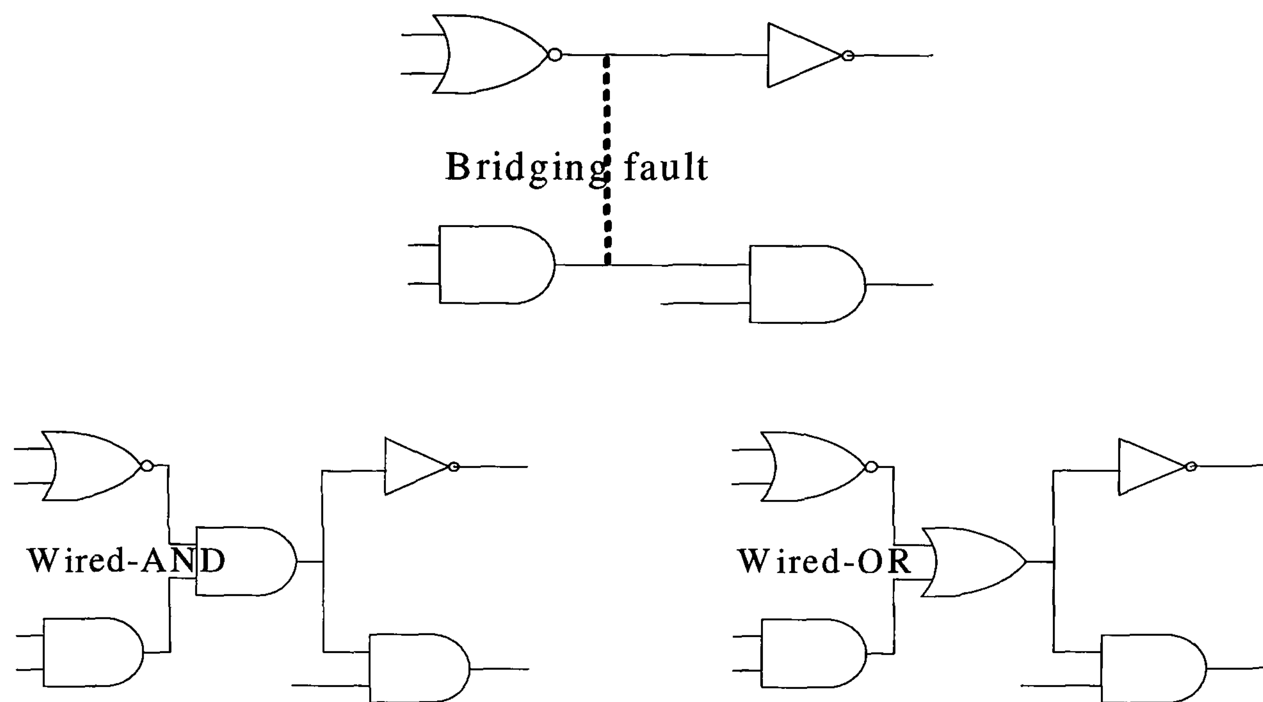


Figure 2.25 Modelling with Wired-AND and Wired-OR

Suppose vector #9 sets net A to 0 and detects SA1@A at output pin 2, thus the simulated fault signature of SA1@A must include the failing bit of {#9:2}. At the same time vector #9 also sets net B to 1 and detects SA0@B at output 5, thus the simulated fault signature of SA0@B must include the failing bit of {#9:5}.

If line dominant model of A-dominates-B is applied, the simulated fault signature of $A_{(bridge)}B$ should include {#9:5} but exclude {#9:2}. B is dominated thus its failing bit is on the bridging fault signature whereas A is always assumed to be the dominant net and never becomes faulty thus its failing bits are excluded. The opposite situation happens when B-dominates-A model is applied.

Under the wired-AND model, the bridged nets would take on a final value of 0 as a result of A = 0 and B = 1, therefore B is the faulty net. B's failing bit – {#9:5} of SA0@B should be included in the bridging fault signature and A's failing bit - {#9:2}

of SA1@A is discarded. In fact, all the failing bits of SA0 of both bridged nets should be included in the bridging fault signature whereas all the failing bits of SA1 of both bridged nets should be excluded under Wired-AND model.

Likewise, the final value of the bridged nets would be 1 under Wired-OR model and A is the faulty net. As a result, {#9:2} is kept and {#9:5} is discarded. As a matter of fact, all the failing bits of SA1 of both bridged nets should be kept in the bridging fault signature while all the SA0 failing bits are discarded.

The bridging fault signatures created by different bridging fault models are matched against the *Obs* under the Matching and Prediction algorithms introduced earlier. The best modelled candidate is selected as the bridging fault diagnosis result. Here, because redundant failing bits have been removed from the *Sim*, a perfect diagnosis of Matching = Prediction = 100% is achievable if the bridging fault behaves as modelled. Successful diagnosis results can be found in Hora's work [Hora02].

2.3.3.4. Open fault diagnosis

2.3.3.4.1. Introduction of open fault models

Open defects are basically categorized as resistive opens and hard opens. A resistive open defect is defined as an imperfect circuit connection that can be modelled as a defective resistor between two circuit nodes that should be connected. A hard open

defect is a complete break between two circuit nodes that should be connected [Li01]. Figure 2.26 illustrates two examples for these two models respectively.

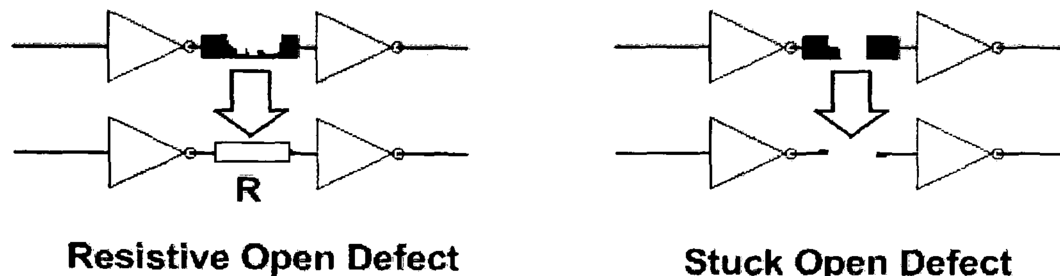


Figure 2.26 Resistive Open and Hard Open

Research shows that resistive open defects will cause extra delay on the open node, i.e. slow-to-rise and slow-to-fall [Li01] [Krishnamachary02]. They are proved to be timing dependent defects, *i.e.*, the test results depend on the testing speed. Therefore the problem of testing and diagnosis of resistive open defects fits in the scope of delay fault testing.

A hard open defect will create a floating gate, whose behaviour is rather complex because the floating gate acquires a voltage that depends on the coupling capacitances of the transistor device and on the surrounding circuitry [Chakravarty99] [Champac94]. Rafiq provides a model summarizing the influence factors of the final voltage value of a floating gate input, as shown in Figure 2.27 [Rafiq98].

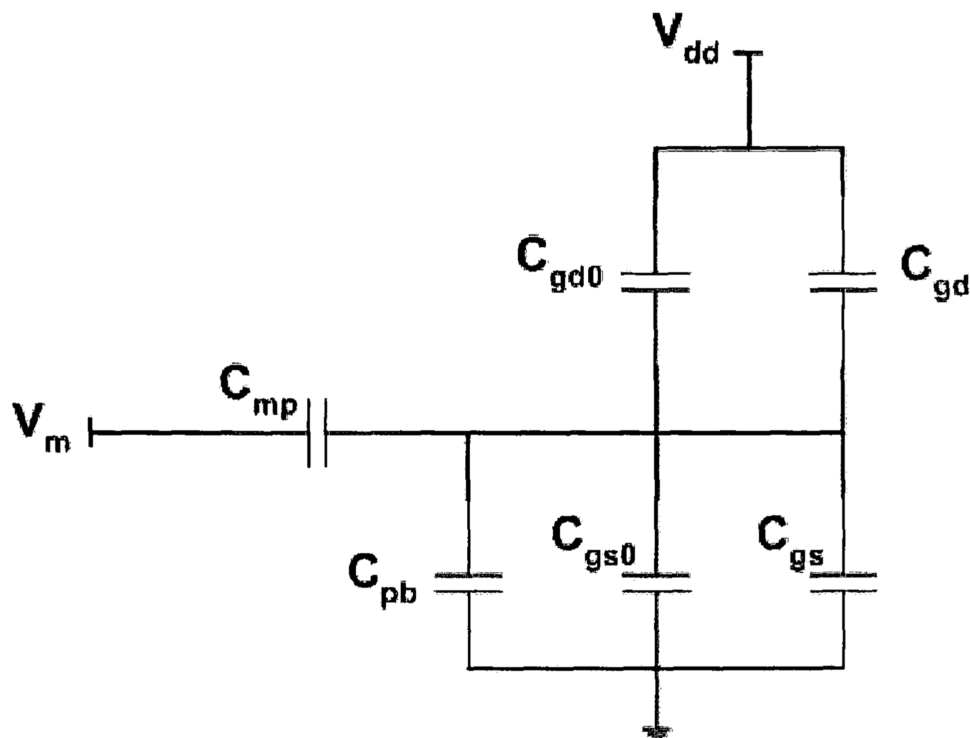


Figure 2.27 Rafiq's Floating Gate Model

Based on Rafiq's model, the final voltage of a floating gate's input is determined by a set of parameters which include the gate to source and gate to drain capacitances Cgs, Cgd, the gate to source and gate to drain overlap capacitance Cgs0, Cgd0, the metal to gate overlap capacitance Cmp, and polysilicon on thick oxide to bulk capacitance Cpb. This method describes how to model the behaviour of a floating gate due to hard open defects and Rafiq has proved full detectability by combining various testing strategies [Rafiq98]. However, from the fault diagnosis point of view, it is not very useful because the parameter Cpb depends on the exact location of the open defect, and the trapped charges Q_0 on the floating node is not obtainable.

Predicting the behaviour of an open defect is very difficult. Therefore diagnosis methods for open faults have to take different approaches from those implemented in stuck-at and bridging fault diagnosis. Compromises in diagnosis results have to be made to incorporate the unpredictability of open faults before a more advanced

modelling method is found.

2.3.3.4.2 Open fault diagnosis

To diagnose open faults, especially when modelling work is not accurate enough, the net model has to be used to sacrifice the Prediction in exchange for guarantee of a high Matching result. Here is how it works: Suppose in Figure 2.28, net D has a hard open fault and behaves in a way that its fault effect is hard to predict. Basically, in the presence of a hard open fault, there are three possible situations that could happen. First, net D is stuck-at 1 when the fault free value of net D is 0. Second, net D is stuck-at 0 when the fault free value of net D is 1. Third, net D keeps the correct fault free value thus no fault is detected. Following is an example showing how net diagnosis model differs from stuck-at fault model in dealing with hard open fault.

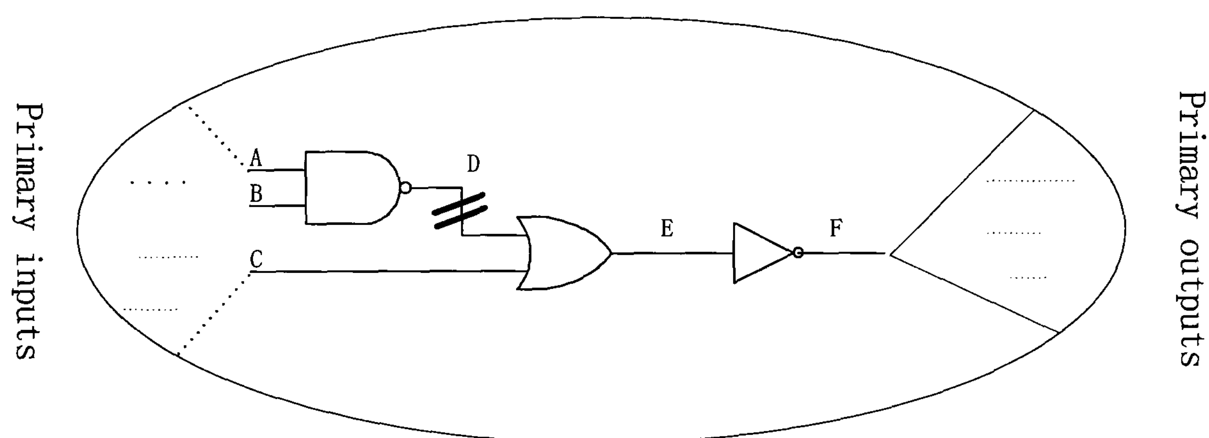


Figure 2.28 A sub-section of circuit containing a hard open faults

Suppose 35 test patterns are applied to the primary inputs. To simplify the problem for the demonstration purpose, all the failing bits are assumed to be observed on output pin number 1; the first 5 vectors happen to set the values of ABC to 001, the following 5 vectors from #6 to #10 happen to set the values of ABC to 101, and the rest are

described in the first column of Table 2.4. The second column of Table 2.4 records the failing bits of the observed fault signatures, *Obs*.

ABC (#number of vectors)	Fails
001 (#1 - #5)	Nil
101 (#6 - #10)	Nil
110 (#11 - #19)	{#11:1 - #16:1, #18:1, #19:1 }
010 (#20 - #24)	{#20:1 - #24:1}
100 (#25 - #32)	Nil
111 (#33 - #34)	Nil
000 (#35)	{#35:1}

Table 2.4 *Obs* of the example in Figure 2.28

Suppose stuck-at model is used to diagnose this sub-section of circuit containing net D. Table 2.5 records the simulated fault signature of D stuck-at-1,

ABC (#number of vectors)	Fails
001 (#1-#5)	0
101 (#6-#10)	0
110 (#11-#19)	{#11:1 - #19:1}
010 (#20-#24)	0
100 (#25-#32)	0
111 (#33-#34)	0
000 (#35)	0

Table 2.5 *Sim* of D stuck-at-1

The relationship of *Obs* and *Sim* for D stuck-at-1 is drawn as below.

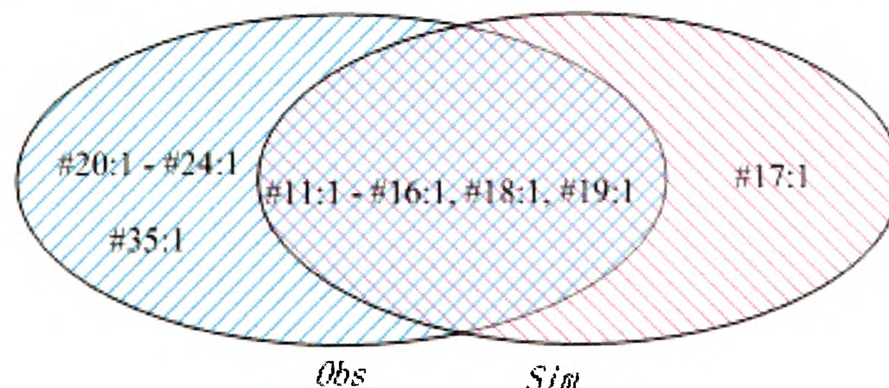


Figure 2.29 Relation of Obs and sim of D stuck-at-1

Thereby Matching and Prediction for D stuck-at-1 are $Matching = \frac{8}{14} \times 100\% = 57\%$, $Prediction = \frac{8}{9} \times 100\% = 88\%$. This low Matching result is hardly convincing enough to identify net D as the faulty net, not to mention the possibility that there may be another net identified with higher Matching results under a totally irrelevant model, such as bridging fault model.

The other problem associated with stuck-at fault diagnosis is the existence of equivalent faults. Equivalent faults are those faults for which no test vector can possibly differentiate, *i.e.*, they give the same response to all the input vectors. Assume in the above case stuck-at model is still able to pick out net D as the top candidate, it inevitably picks out all the equivalent stuck-at faults along with the right one, as shown in Figure 2.30.

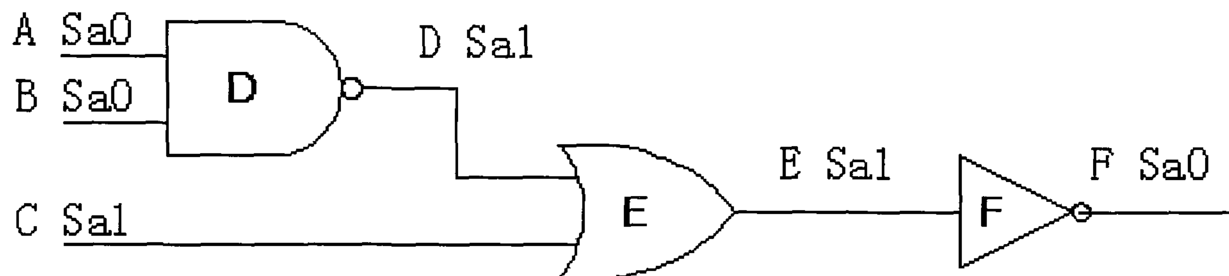


Figure 2.30 Equivalent stuck-at faults of D stuck-at-1

All the stuck-at faults noted in Figure 2.30 will have the same Matching=57% and Prediction=88%, and will thus compromise the resolution of the diagnosis.

The idea of a net diagnosis model is based on focusing on getting a high Matching result for a specific net without much concern about including too many failing bits in *Sim* and having a low Prediction results. It forms the signature of a specific net by

combining both the stuck-at-1 signature and the stuck-at-0 signature of that specific net. In other words, there are not only the stuck-at-1 failing bits in set *Sim,* but also the stuck-at-0 failing bits. The *Sim* of D under net diagnosis model is in Table 2.6.

ABC (#number of vectors)	Fails
001 (#1-#5)	**0**
101 (#6-#10)	**0**
110 (#11-#19)	**{#11:1 - #19:1}**
010 (#20-#24)	**{#20:1 - #24:1}**
100 (#25-#32)	**{#25:1 - #32:1}**
111 (#33-#34)	**0**
000 (#35)	**{#35:1}**

Table 2.6 *Sim* of D under net diagnosis model

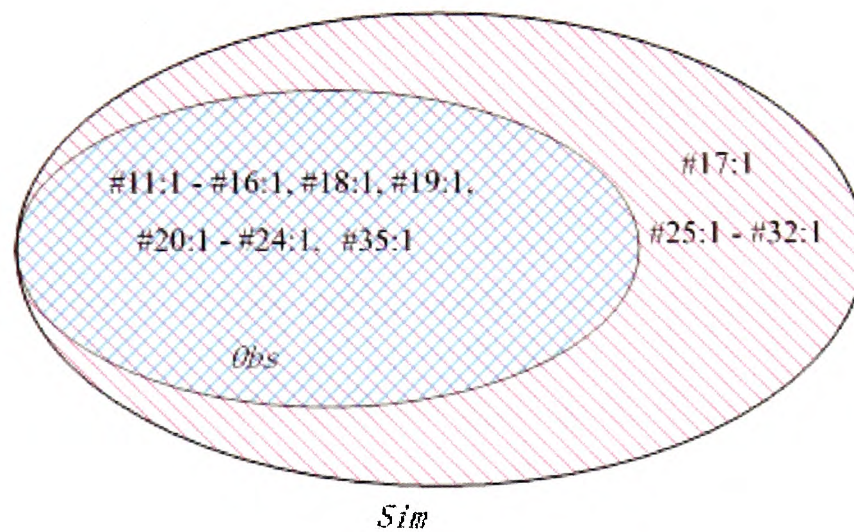


Figure 2.31 Relationship of Obs and Sim of net D under net diagnosis model

Figure 2.31 is the new relationship between *Obs* and *Sim* of net D under net diagnosis model. Table 2.7 shows the diagnosis results from net A to net F under the net diagnosis model. Not only does net D achieve a Matching = 100%, but also this model differentiates all the nets from each other. The reason why the net diagnosis model can distinguish between A-F is that it not only takes into account of the propagated patterns,

but also uses information from those not propagated. The high Matching can be explained by the fact that the net diagnosis model predicts all the possible faulty behaviour on each node, whilst the drawback is clearly a low Prediction.

Net	Matching	Prediction
A	93%	93%
B	57%	47%
C	57%	73%
D	100%	61%
E	100%	40%
F	100%	40%

Table 2.7 Matching and Prediction under net diagnosis model

Bearing in the mind that the net diagnosis model always over-predicts faulty behaviour, the result of Prediction is certainly valued less than the result of Matching in deciding the most likely failure, which is consistent with the rules set for Matching and Prediction algorithms previously. Following this rule, net D has the Matching of 100% and has a higher Prediction than those also having the Matchings of 100%. Thus it is picked out as the most likely faulty net.

Inter-gate level hard open faults can be effectively diagnosed by the net diagnosis model. Successful diagnosis results are given in [Hora02] and [Venkataraman01]. The other common type of hard open faults, stuck-open fault, happens at intra-gate level. It assumes that hard open faults cause a permanent "off" status on one of the transistors at a particular gate. The net diagnosis model is not the best model for the stuck-open faults. The diagnosis issue of stuck-open fault will be the main topic of the next chapter. On the other hand, resistive open faults often result in extra delay on the faulty net,

therefore a transitional delay fault diagnosis can be applied. Details of transitional delay fault diagnosis will be explained in Chapter 3 along with the new method for intra-gate resistive open fault diagnosis.

Li's work on stuck-open fault diagnosis:

One previous work on open fault diagnosis worthy to be mentioned is from Li [Li02]. Li's work targets at the intra-gate stuck-open fault diagnosis which will be main topics of the next chapter. Li's work first use stuck-at fault diagnosis to locate a few suspected gates with intra-gate stuck-open faults, then Li started to build up stuck-open fault signatures for each individual open transistor of these suspected gates. These fault signatures are stored in excitation tables. Using these excitation tables, realistic failing bits are selected from the stuck-at signatures by comparing with the input signals of these particular gates. Those vectors that can not excite a stuck-open fault are filtered out. Since stuck-open fault behaves in a sequential way (it is dependent on a sequence of input vectors), the excitation table contains the description of a sequence of input vectors on the gates. The same Matching and Prediction rules were applied in Li's work by matching these stuck-open fault signatures with the observed failures. The paper claimed several successful diagnosis results of stuck-open faults, however, the excitation tables have to be built manually, therefore this method is not suitable to conduct in the real industry settings.

2.4. Conclusion

This chapter has introduced the basic knowledge of CMOS circuit and its manufacturing process. Before becoming a final CMOS product, the silicon wafer undergoes several cycles of processing including Lithography, Etching, Thermal Oxidation, Deposition and Planarisation, during which normally 20 to 30 masks are used to create the specific layout patterns. The complexity of the process means the IC products are very susceptible to variations which may create defects. Lithographic defects, oxide defects and metal defects are amongst the most common types. Therefore advanced methods have to be developed to test for these defects and eliminate them in the future. Although a functional test is straightforward and intuitive, it suffers from the fact that performing a thorough functional test for sensible sized IC devices is impractical. The fault model based test is an effective alternative which incorporates the issues like fault modelling and test pattern generation. Additional hardware is included in the design to support this method. Design for Testability techniques such as Scan Chain and BIST have been developed to significantly simplify the test pattern generation process and to enable us to maintain high fault coverage.

The diagnosis algorithm of Matching and Prediction is demonstrated in Section 2.3.2.2. It is capable of evaluating the diagnosis quality of various types of faults such as stuck-at fault, bridging fault, open faults. The higher these two measurements are, the better the diagnosis quality is. For the bridging faults, suitable logic bridging fault

models like Wired-AND, Wired-OR and line dominant model play important roles in eliminating the redundant bridging fault signatures, creating a *Sim* of the bridging fault as close to *Obs* as possible. For the open faults, whose behaviour is hard to predict, the net diagnosis model combines both the stuck-at-0 and stuck-at-1 signatures to guarantee a high Matching for the faulty net at the expense of its Prediction. It also differentiates those logically equivalent nets under the common stuck-at fault diagnosis.

All the diagnosis issues mentioned in this chapter are concerned about the inter-gate level fault diagnosis, however, IC defects very often occur inside gates. An effective diagnosis of these defects requires further development in the algorithms dealing with transistor (intra-gate) level representations. The next two chapters will introduce new methods capable of handling intra-gate level fault diagnosis issues.

Chapter 3. Intra-gate Open Fault Diagnosis

3.1. Introduction

The previous chapter introduces the available measures up to now to diagnose the fault by electrical analysis. All of the methods are developed on the inter-gate level and targeting on the inter-gate level faults. Although there have been long time discussions over the intra-gate faults and intra-gate level (transistor level) pattern generations [Madhukar85] [Ferrer90], it is surprising to find that no systematical work on intra-gate level fault diagnosis has ever been introduced so far. Meanwhile, the frequent use of complex gates in modern CMOS design results in high occurrence of intra-gate faults which also have been seen in Philips Research's practice.

This chapter and Chapter 4, for the first time, systematically address the problem of intra-gate fault diagnosis. This chapter talks about intra-gate stuck-open fault diagnosis and intra-gate resistive open fault diagnosis. The next Chapter will address the problem of intra-gate bridging fault diagnosis. In this chapter, a novel transformation method is developed to transform the transistor level circuit description to a gate level description where stuck-open faults are represented by stuck-at faults, so that the stuck-open faults can be diagnosed directly by any of the

stuck-at fault diagnosis tools. The method builds up the same relationship between the intra-gate resistive open faults with the transitional delay faults, therefore the intra-gate resistive open faults can be diagnosed as transitional delay faults at the gate level. By doing this, the intra-gate open fault diagnosis problem can be solved in not only a very efficient and effective way but also with the maximum utilization of the current diagnosis platform with little extra cost.

Section 3.2 introduces the stuck-open fault model and Section 3.3 explains a transformation method developed for stuck-open fault diagnosis, together with some of the special situations of intra-gate hard open faults and their solutions. Section 3.4 introduces a similar approach to diagnose the intra-gate resistive open faults. Section 3.5 is the conclusion.

3.2. Stuck-open fault model

A stuck-open fault refers to a transistor which becomes permanently non-conducting as a result, for example, of an internal failure or a connection break [Elziq81] [Lee89] [Millman89] [Menon93] [Li01]. Figure 3.1 shows a NAND gate consisting of two p-transistors (T1 and T2) and two n-transistors (T3 and T4). Suppose transistor T1 in Figure 3.1 is cut off permanently due to an open defect and a set of exhaustive vectors are applied on A and B in the following sequence: (A, B) = {(0, 0), (0, 1), (1, 1), (1, 0)}. The expected fault free output should be Z = {1, 1, 0, 1}. Considers what happens

when the sequence of vectors is applied. When the first vector (0, 0) applies, the status of the four transistors is T1-off (should be 'on' if not stuck-open), T2-on, T3-off and T4-off. The output Z is driven to 1 by T2. When the second vector (0, 1) applies, the status of the four transistors is T1-off (should be 'on' if not stuck-open), T2-off, T3-off and T4-on, the output Z is then completely detached from both the power supply and the ground as a result of the stuck-open fault, however, as it should keep the previous value of Z of the earlier vector, it stays at 1, which is same as the fault free value. The next two vectors also see no faulty effect on output Z.

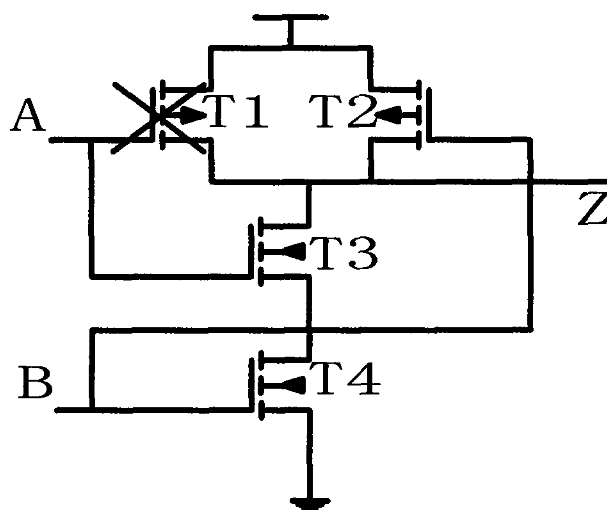


Figure 3.1 Stuck-open fault at transistor T1

This example shows that even an exhaustive set of vectors can fail to detect a stuck-open fault. This does not mean that stuck-open faults are not detectable, but it means the presence of a stuck-open fault has caused the gate to behave in a sequential way. Suppose the sequence of the exhaustive vector set is changed to {(1, 1), (0, 1), (0, 0), (1, 0)} with expected fault free response as {0, 1, 1, 1}. Applying the same analysis, under presence of the stuck-open fault, the output Z should respond as {0, 0, 1, 1} which is different from the fault free response. This shows that the sequence of the

vectors matters in deciding how the stuck-open fault behaves.

In order to test stuck-open faults, two vectors are needed. The first one is called the *initialization vector* and second one is the *sensitization vector*. Because the stuck-open transistor retains its previous logic value, the initialization vector is used to initialize the output value to the one that is complement of the expected logic value when the sensitization value is applied. The sensitization vector has to make sure the stuck-open transistor is turned on and controls the only conducting path to the output from either the power supply or the ground. Therefore when the stuck-open fault cuts off this path, the output will be a floating net. For example, the initialization vector of the above example in Figure 3.1 should be (1, 1) which sets Z to 0, so that when the sensitization vector (0, 1) follows, Z becomes a floating net and keeps the previous value 0 set by the initialization vector, which is different from the fault free value 1.

Much work has been done in respect to stuck-open test pattern generation. Some authors have investigated the stuck-open fault coverage when a set of stuck-at test vectors is applied [Woodhall87] [Millman89]. Most of the work has focused on the detection of stuck-open faults but with little attention to diagnosis.

3.3. Stuck-open fault diagnosis

Until now, the only systematic published work of stuck-open diagnosis is from [Li02].

As discussed in the previous chapter, this method is an independent development which requires excitation table building and the rewriting of a matching procedure that is already contained in most of the conventional diagnosis tools. In Section 3.3, a new automatic and effective stuck-open fault diagnosis method is introduced and successful diagnosis results are presented using wafer test data and an internal diagnosis tool from Philips Research.

Vector#	Previous Z	A	B	Fault free Z	Observ ed Z	Z-SA0	A- SA1
1	0	0	0	1	1	0	1
2	0	0	1	1	0	0	0
3	0	1	0	1	1	0	1
4	0	1	1	0	0	0	0
5	1	0	0	1	1	0	1
6	1	0	1	1	1	0	0
7	1	1	0	1	1	0	1
8	1	1	1	0	0	0	0

Table 3.1 The behaviour of different faults in Figure 3.1

3.3.1. Stuck-open fault under stuck-at diagnosis

To detect a stuck-open fault, a sequence of two vectors needs to be applied. Again, consider the NAND gate with a stuck-open fault in p-transistor T1 in Figure 3.1. Table 3.1 shows all possible vectors that could apply, including the sequential effects - the previous value of Z in column 2. Column 5 is the expected fault free output (Z), column 6 is what are observed at the output (*Obs*) by performing electrical simulation when T1 is stuck open, and column 7 and column 8 are the fault signatures of Z

stuck-at 0 (SA0) and A stuck-at 1 (SA1), respectively (To show how the fault is diagnosed under the stuck-at fault model.).

Suppose a stuck-open occurs in T1 in Figure 3.1, from Table 3.1, *Obs* is {#2:Z} (the only observed failing bit of Z). To show how the stuck-open fault manifest itself under the stuck-at fault diagnosis, output Z stuck-at-0 fault is simulated, the *Sim* is {#1:Z, #2:Z, #3:Z, #5:Z, #6:Z, #7:Z}. As the direct input of transistor T1, A stuck-at-1 is also simulated, the *Sim* is {2/Z, 6/Z}. Recall the definition of Matching and Prediction in Section 2.3.2.2:

$$Matching(M) = \frac{Obs \cap Sim}{Obs} \times 100\%$$

$$Prediction\ (P) = \frac{Obs \cap Sim}{Sim} \times 100\%$$

To look at the diagnosis result of Z stuck-0 when the actual fault is a stuck-open fault at transistor T1, *Obs* is compared with the *Sim* of Z stuck-at-0:

Matching (M) = |{#2:Z}| / |{#2:Z}| = 1/1 = 100%

Prediction (P) = |{#2:Z}| / |{#1:Z, #2:Z, #3:Z, #5:Z, #6:Z, #7:Z}| = 1/6 =17%

Alternatively, to look at the diagnosis result of A stuck-at-1 when the actual fault is a stuck-open fault at transistor T1 (the one that A directly connects to) *Obs* is compared with the *Sim* of A stuck-at-1:

Matching (M) = |{#2:Z}| / |{#2:Z}| = 1/1 = 100%

Prediction (P) = |{#2:Z}| / |{#2:Z, #6:Z}| = 1/2 =50%

The observation from this example shows when a stuck-open fault is present inside a gate, a stuck-at fault at the output (either stuck-at-0 or stuck-at-1) will always give Matching = 100% and Prediction ≤ 100%, and if the stuck-open transistor has a direct input (in Figure 3.1, A is the direct input of transistor T1), the stuck-at fault at this direct input (either stuck-at-0 or stuck-at-1) will also give Matching = 100% and Prediction ≤ 100%. The reason for this is given as following. Think about the situation when a stuck-open fault happens on a p-transistor, resulting in its inability to make a conducting path between Vdd and the Output. With an open fault in a p-transistor, the gate output cannot be pulled up to a logic one via that transistor. If the output was previously a zero, it will stay at a zero and appear as a stuck-at fault at the output. The stuck-at-0 signature of the output will contain those failing bits. However, if the previous output was a one, no failing bit is observed at the output, but the stuck-at-0 signature of the output still contains those failing bits. Alternatively, the output can be pulled up to a logic one via some other transistors, the output will be also correct, no failing bit is recorded while the stuck-at-0 signature of the output will still contain these failing bits. In other words, sometimes stuck-open results in stuck-at fault at the output but sometimes not. It therefore follows that the *Obs* of a stuck-open will always be a subset of, or the same as, the *Sim* of a stuck-at fault at the output. Not surprisingly, the previous example shows Matching = 100% and Prediction = 17% for Z stuck-at-0. The same argument applies for stuck-open fault happening at n-transistors when the output stuck-at-1 fault is at concern.

When a stuck-open p-transistor has a direct input (as A for T1 in Figure 3.1), the A stuck-at-1 fault has the same effect of turning off T1. On top of that, it also helps to turn off other pull-up paths (by turning off other p-transistor(s)) and opening pull down paths (by turn on other n-transistor(s)). So, regardless of the previous output value, when a vector is due to fail under T1 stuck-open model, it must fail under the A stuck-at-1 model, but not vice versa. It therefore follows that the *Obs* of a stuck-open will always be a subset of, or same as, the *Sim* of a stuck-at fault at the direct input. Therefore Matching = 100% and Prediction = 50% are diagnosed for A stuck-at-1 in the previous example. The same argument applies for stuck-open fault happening at n-transistors when the direct input stuck-at-0 fault is concerned.

In general, when a stuck-open fault occurs and the stuck-at model is used to diagnose the circuit, both the output of the faulty logic gate and the direct input (if there is one) of the stuck-open transistor will report Matching = 100%. The relationship of their Predictions is: Prediction (output) ≤ Prediction (direct input, if there is one) ≤ 100%.

Knowing how stuck-open faults behave in a stuck-at fault diagnosis provides us with a very promising direction for stuck-open diagnosis. Because the stuck-at fault diagnosis tools are widely available and extensively practiced in the IC testing industry, they can be implemented to give clues to the potential gates with stuck-open faults. Those gates with either output or input diagnosed as Matching = 100% and Prediction ≤ 100% under stuck-at fault diagnosis are the primary suspects for having

stuck-open faults. The conventional idea is that this is as far as the stuck-at fault diagnosis can be used to help stuck-open diagnosis [Li02], however, a novel transformation method is introduced in the next section in which the suspect gates are transformed to diagnose intra-gate stuck-open faults purely using stuck-at fault diagnosis tools.

3.3.2. Transforming stuck-open faults to stuck-at faults

The basic idea of the transformation is to represent a stuck-open fault with an equivalent stuck-at fault so the stuck-at diagnosis tool can be used, instead of building a new stuck-open fault diagnosis tool. Since the stuck-at fault diagnosis can be only conducted at gate level, the stuck-open fault has to be migrated to the gate level.

The following example explains how this is realized. Figure 3.2 is the schematic of an example logic gate with three inputs (*A B C*) and one output (*Z*). Each input signal is split into a number of branches, one for each transistor the input is connected to. For the gate from our example, input *A* will be split in *A_1, A_2, A_3* and *A_4,* each representing the transistor it connects to. The transformation from the transistor level schematic to the gate level is conducted by the following rules.

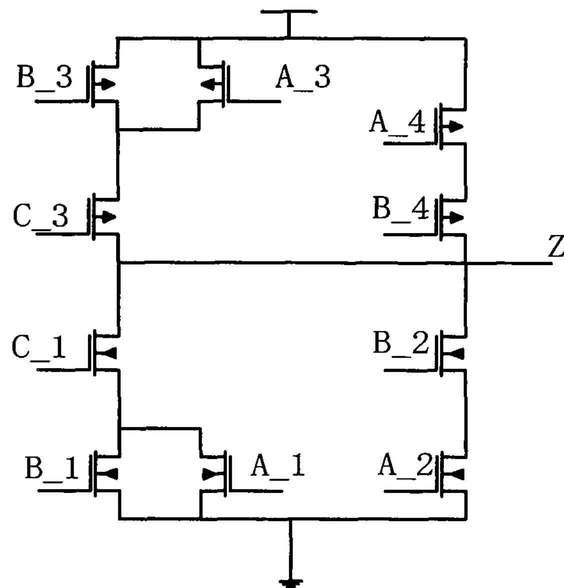


Figure 3.2 Schematic of the example circuit

For n-transistors:

A. Replace all the n-transistors with the element as shown in Figure 3.3. The idea is to guarantee that a **ZERO** voltage from the source will be transmitted to the drain when value on the input (*A* in our example) is **ONE** (transistor turned-on).

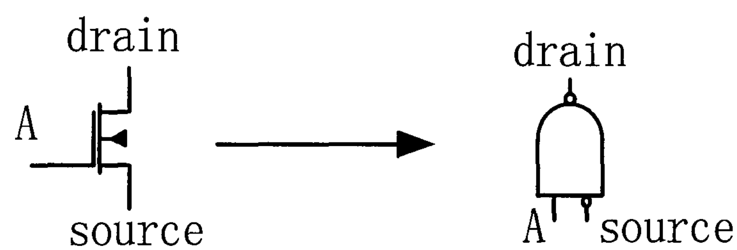


Figure 3.3 Replacement of n-transistor

B. Place an AND gate where a parallel connection between transistors is present, as Figure 3.4. The purpose is to guarantee that the **ZERO** signals from the source will be transmitted to the drain as long as either one of the transistors is turned-on. Therefore the AND gate makes sure if one of the drains is **ZERO**, the output will be **ZERO**. If

none of the drains is **ZERO**, the output will be **ONE**.

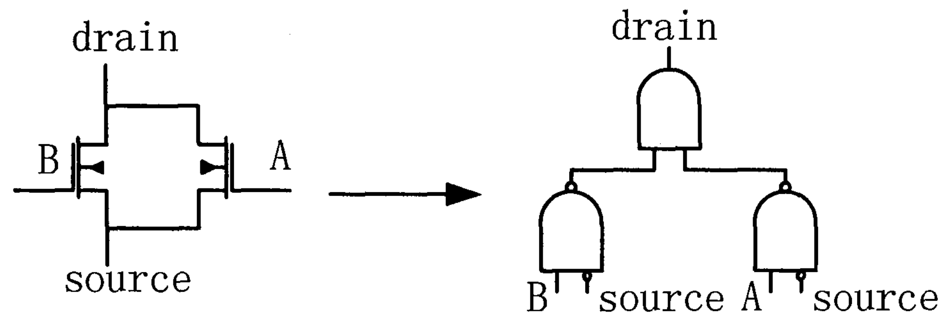


Figure 3.4 Replacement of parallel n-transistors

For p-transistors:

A. Replace all the p-transistors with the element as shown in Figure 3.5. The purpose is similar, to guarantee the propagation of logic **ONE** from the source. When the input *A* is **ZERO** (transistor turned-on) and the source signal is **ONE**, the drain will be **ONE**. Otherwise, the drain will be **ZERO**.

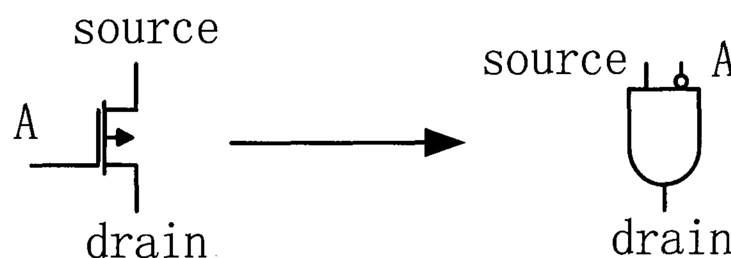


Figure 3.5 Replacement of p-transistor

B. Place an OR gate where a parallel connection is present, as Figure 3.6. The purpose is to guarantee that the **ONE** signals from the source will be transmitted to the drain as long as either one of the transistors is turned-on. Therefore the OR gate makes sure if one of the drains is **ONE**, the output will be **ONE**. If none of the drains is **ONE**, the output will be **ZERO**.

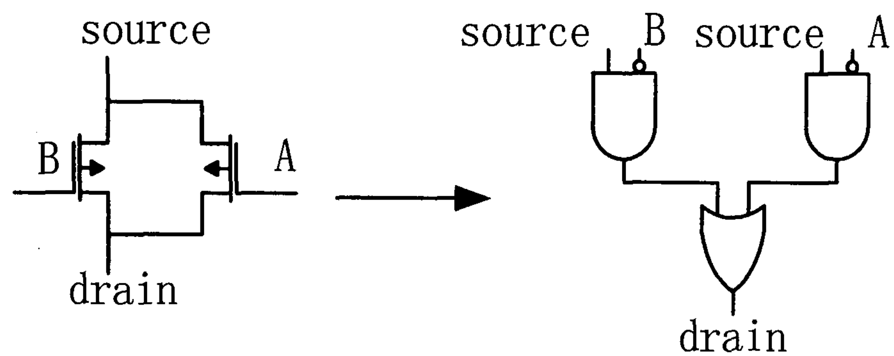


Figure 3.6 Replacement of parallel p-transistors

Analysis of first phase transformation:

To excite a stuck-open fault, two vectors are needed, the first vector sets the output to a certain value, the second vector has to make sure that the target transistor plays an indispensable role to drive the output to the opposite value. Thus when the target transistor is stuck open, the output fails to go to the opposite value and keeps the previous one. From the structural point of view, this means the stuck-open transistor has to control the only serially conducting path under the second vector.

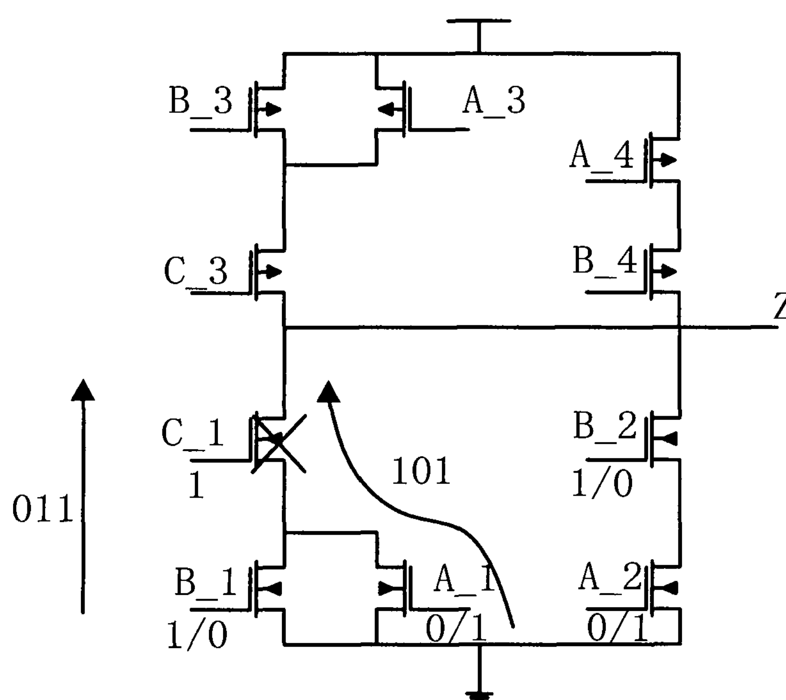


Figure 3.7 Stuck-open fault at C_1

For instance in Figure 3.7, to excite a stuck-open at *C_1*, the first vector (for instance *ABC*=000) sets *Z* to one, the second vector should be either *ABC*=011 or *ABC*=101. The expected pull-down paths for these two vectors are depicted by the arrows on Figure 3.7. In either case, *C_1* controls the only conducting path. The invalid propagation case s (*ABC=011, ABC=101*) will be introduce in Figure 3.8.

The placement of AND and OR gates makes sure that the stuck-at fault propagates only when there is no parallel conducting path, in other words, the stuck-at fault controls the only conducting path. For instance Figure 3.8 shows the transformed circuit of the n-transistors part in Figure 3.7. Under either of the second vectors (*ABC*=011 and *ABC*=101) that excite the *C_1* stuck-open in Figure 3.7, the stuck-at-0 fault at *C_1* in Figure 3.8 is also propagated to N_output. *C_1* controls the only conducting path under either of the second vectors depicted by the arrows. Therefore, when it is stuck-at-0 (SA0), the N_output is stuck-at-1, indicating the corresponding stuck-open fault at transistor *C_1* is propagated.

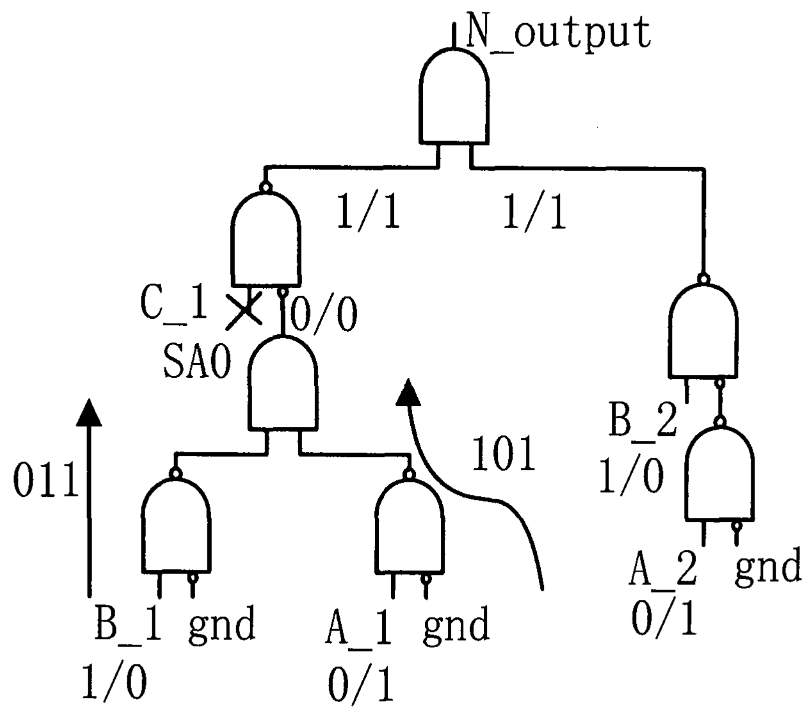


Figure 3.8 Stuck-at fault at C_1

Suppose the second vector is ABC = 111, stuck-open transistor *C_1* is only controlling the only conducting path, Figure 3.9. There is another conducting path through transistor *A_2* and *B_2*, thus the second vector is not a valid sensitization vector for this particular stuck-open fault. In Figure 3.9, the transformed circuit reacts in a way that there is ground signal 0 (shown in bold) propagating through the two gates on the right – *A_2* and *B_2*, thus blocking the propagation of the SA0 fault at *C_1*. The same observation holds for the p-transistors part too.

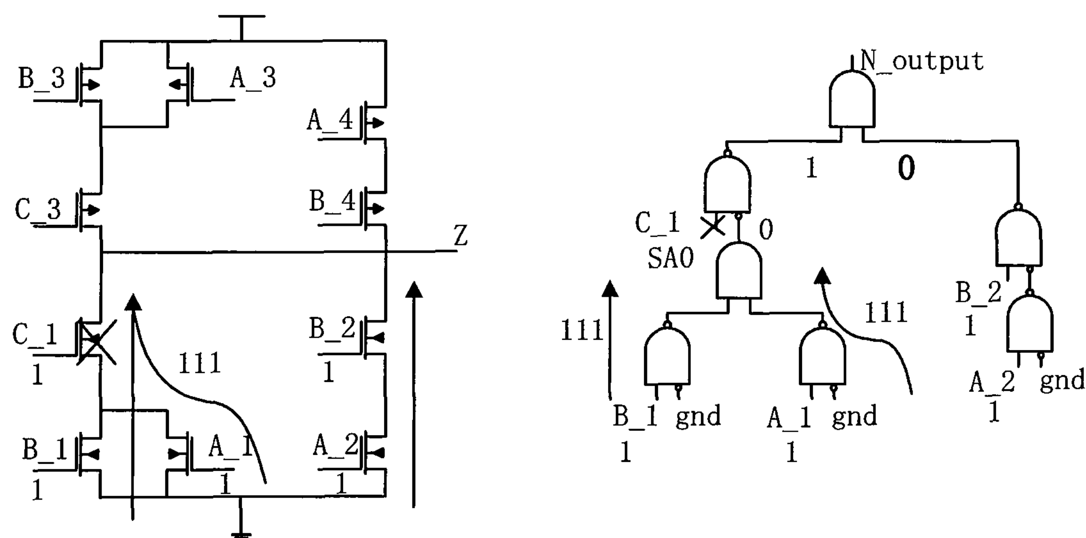


Figure 3.9 Invalid sensitization vector

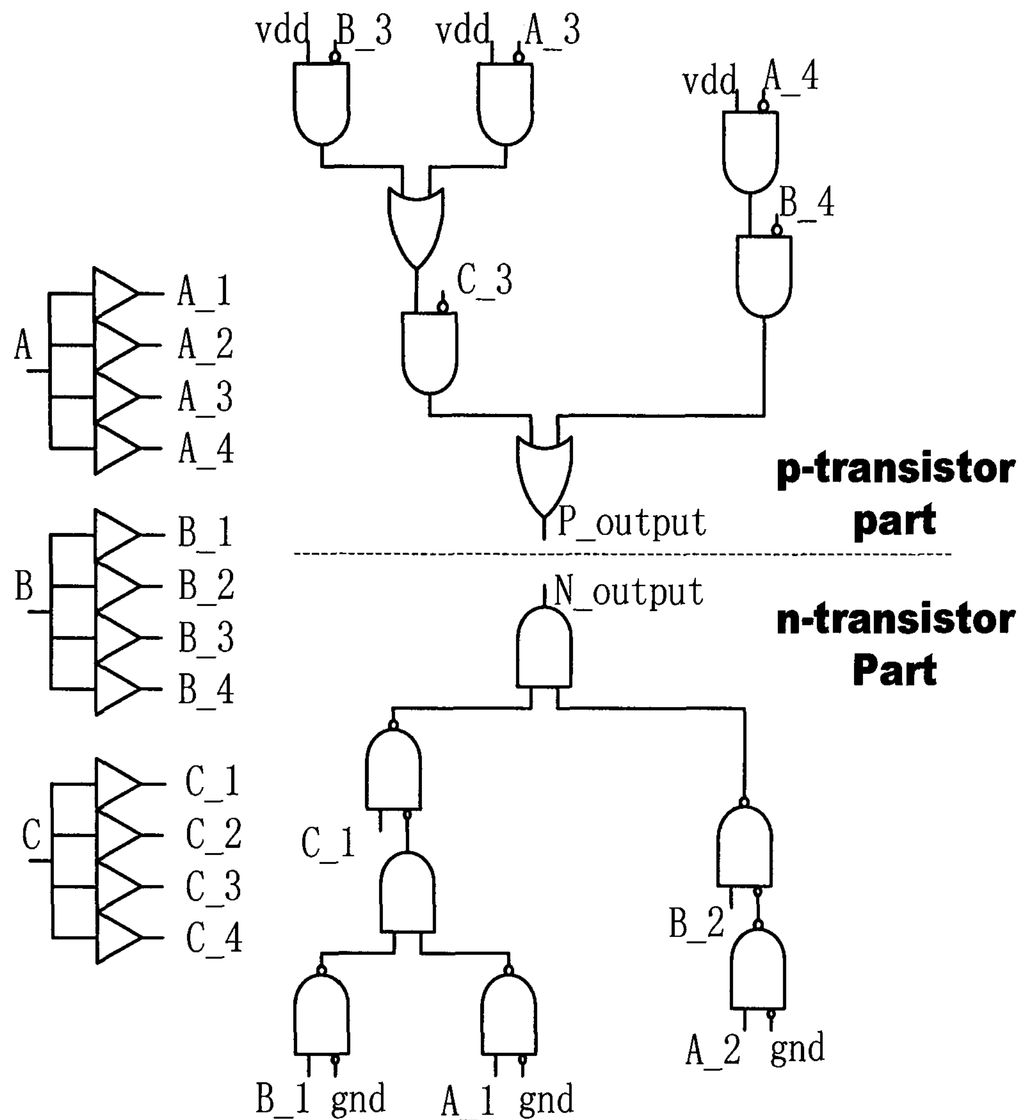


Figure 3.10 The first phase of transformation

The view after the whole transformation of the gate shown in Figure 3.2 is as Figure 3.10. The whole graph consists of a *P* Part and an *N* Part. Since this is virtually created for representing stuck-open faults, it is sensible to assume the only possible faults will be stuck-at-0 on branch inputs for the *N* Part and stuck-at-1 on branch inputs for the *P* Part. For instance, for our example, a stuck-open fault at *C_1* in Figure 3.2 will be represented by a stuck-at-0 fault at *C_1* in Figure 3.10 and a stuck-open fault at *A_3* in Figure 3.2 will be represented by a stuck-at-1 fault at *A_3* in Figure 3.10.

Final transformation:

Now, only when the stuck-open second vector sensitization requirement is met, will either P_output or N_output in Figure 3.10 fail. However, not only does the second vector sensitization requirement need to be met, but the first vector initialization requirement also needs to be met before a fail at the final output can be seen, *i.e.* the first vector needs to set the output to the opposite value to that expected from the second vector.

This leads to the final transformation shown in Figure 3.11. Two sub-circuits are used to generate the fault free value of the current Z and the previous Z. In the fault free case, the P_output and N_output give the same responses. They both give the result at Z in Figure 3.2. The P_output and N_output are connected by an XOR gate to generate a FaultFlag signal, which will be zero when there is no fault and one when there is a fault (*i.e.* when the P_output and N_output are different to each other).

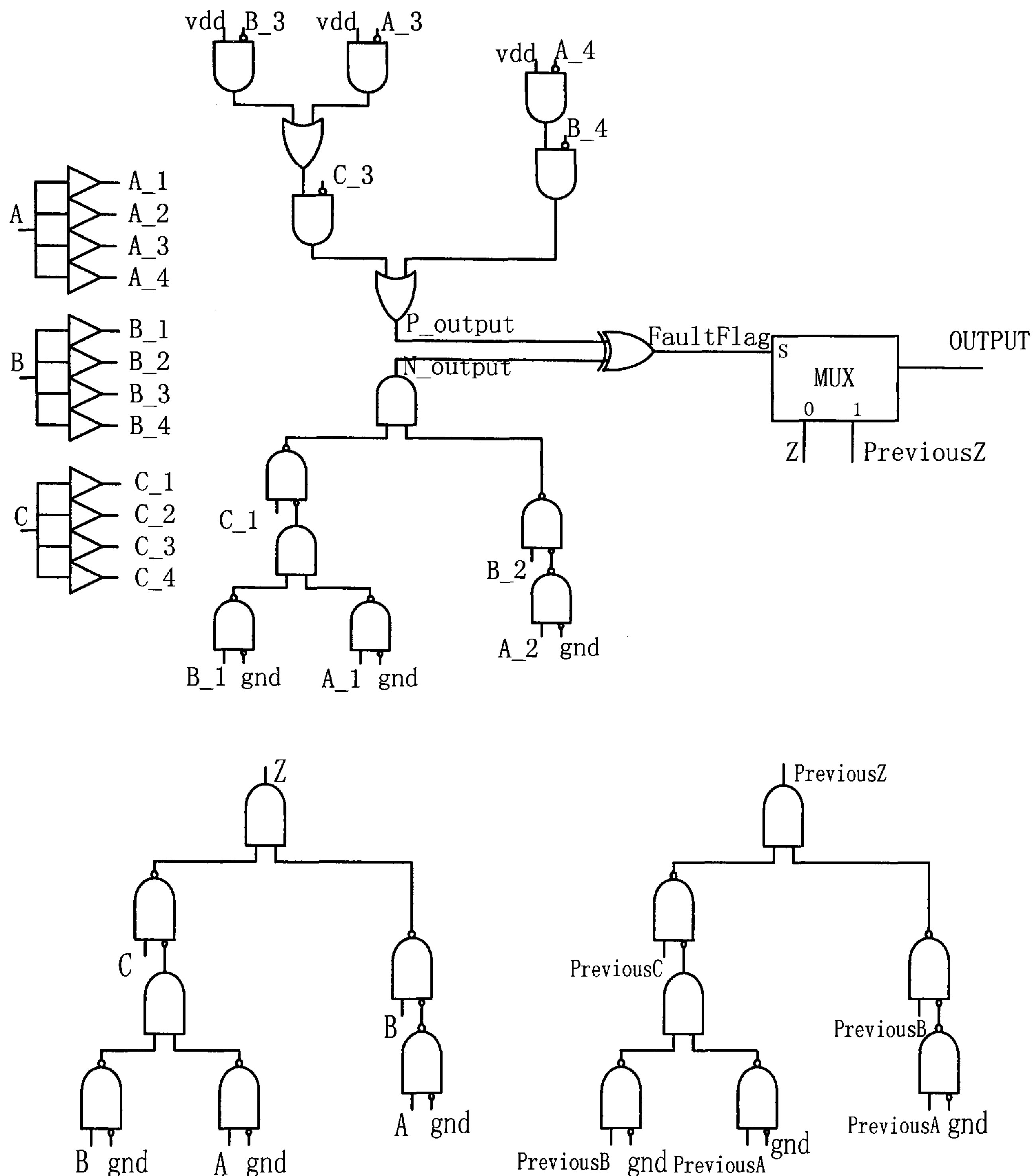


Figure 3.11 Final picture of transformation

When no stuck-at fault propagates, meaning there is no valid sensitization vector for stuck-open fault, P_output = N_output, and FaultFlag = 0, the MUX selects the correct current Z value, so the OUTPUT will not fail. When a stuck-at fault propagates, meaning the second vector requirement of stuck-open sensitization is met, P_output $\neq$ N_output, and FaultFlag = 1, the MUX then selects the previous Z value. It is then the

previous Z value which determines if the OUTPUT fails or not. If the previous *Z* value is different from the correct current *Z* value, the OUTPUT fails. If they are the same, the OUTPUT does not fail. By adding this extra MUX, the first vector requirement for stuck-open initialization is also met. Now, all the stuck-open faults in the *P* part are represented by the corresponding stuck-at-1 faults and all the stuck-open faults in the *N* part are represented by the corresponding stuck-at-0 faults.

Multiple blocks gate transformation:

The example above shows how a single block gate is transformed. It is also possible to extend this method to gates with multiple blocks like gate nds4ab in Figure 3.12. Each of the two blocks is transformed to a *P* part and an *N* part (P_1, N_1 for the first block and P_2, N_2 for the second block). The transformation procedure for each block is exactly the same as for a single block gate, with signal *Connect* taken as an input signal of the second block. Notice signal *PreviousConnect* is needed to generate the previous output of the second block in Figure 3.12. When the stuck-open fault occurs in the second block, the first block simply serves as an input value generator for *Connect* and *PreviousConnect.* The situation is no different from the single block transformation, where *Connect* is taken as a normal input and split into *Connect_1* and *Connect_2.* When the stuck-open fault happens in the first block, the second block has no fault and thus always selects the current output value which is generated by *Connect.* The first block decides under what initialization and sensitization vectors is the stuck-open fault effect propagated to *Connect.* The second block then only judges by the sensitization

vector to see if the logic combination of the second block allows the stuck-at fault of *Connect* to propagate to *Z*.

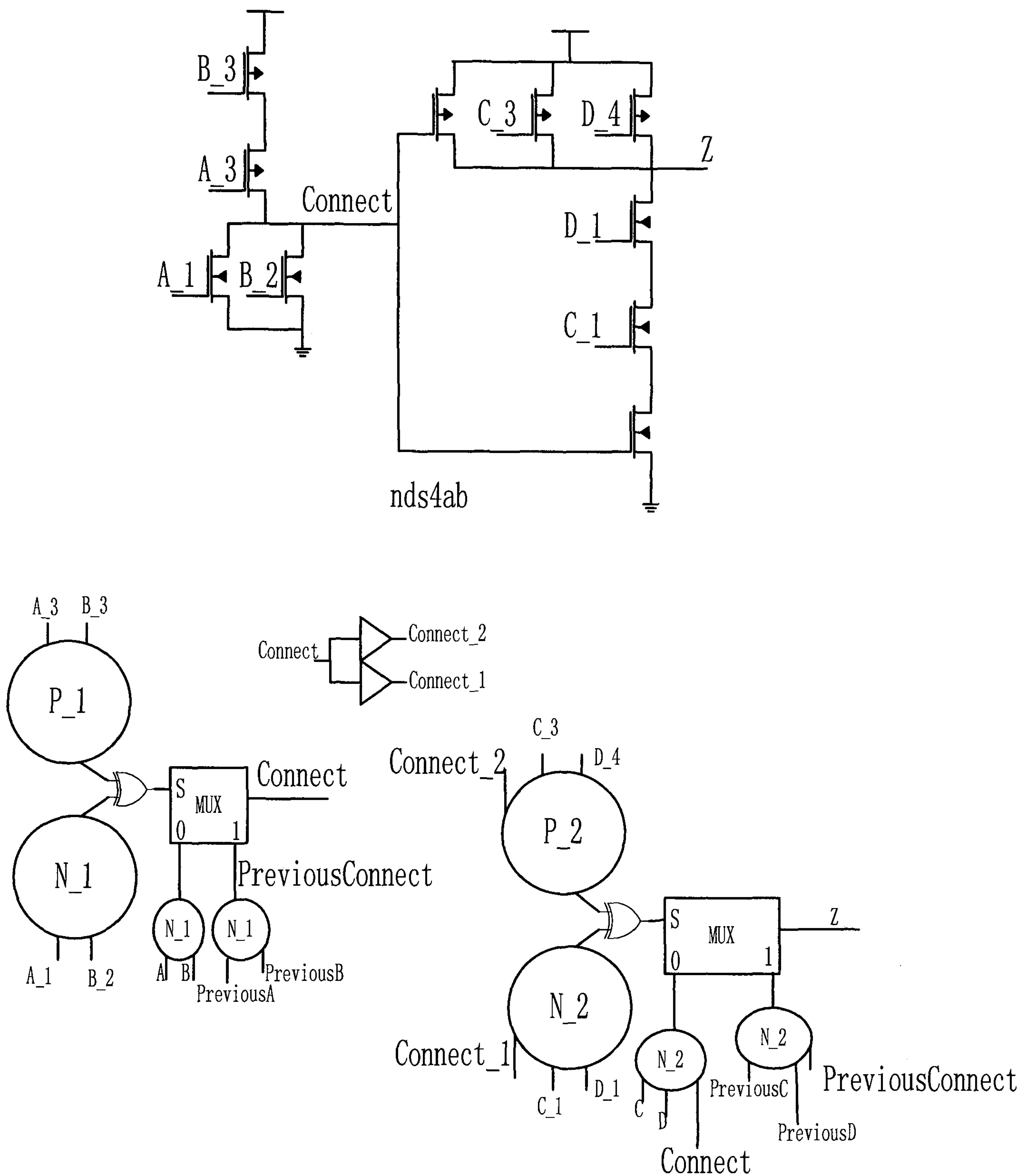


Figure 3.12 Multiple blocks gate transformation

3.3.3. Stuck-open fault diagnosis flow

The last two sections addressed the issues of how stuck-open faults perform under

stuck-at fault diagnosis and of how to transform stuck-open faults into gate level stuck-at faults. It is now possible to present the overall diagnosis flow for stuck-open faults. Figure 3.13 shows the overall diagnosis flow of stuck-open faults. An initial single stuck-at fault diagnosis has to be performed on the whole circuit in the first step. Given the relationship explained in Section 3.3.1, those gates with either an input or an output diagnosed as Matching = 100% and Prediction < 100% under the first round stuck-at fault diagnosis are selected as the top suspects of stuck-open gates.

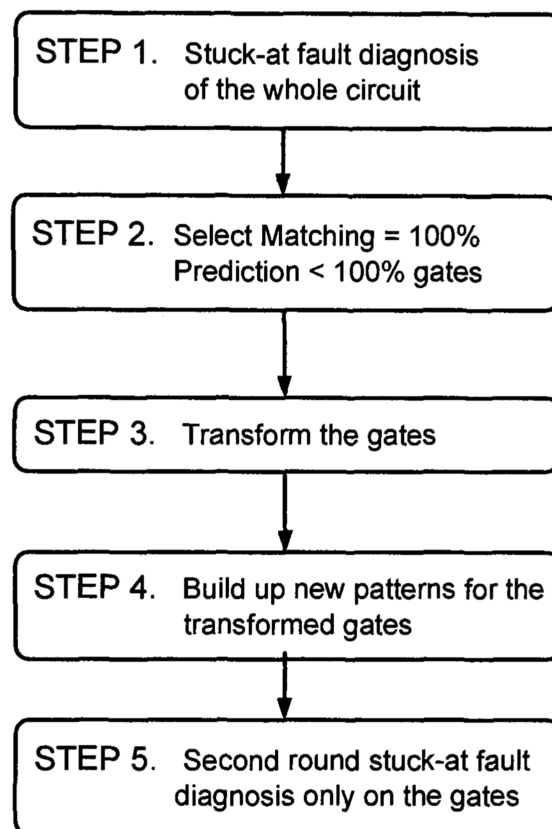


Figure 3.13 Overall flow of stuck-open fault diagnosis

The next step is to transform the gates to a new form where all the stuck-open faults are represented by stuck-at faults, as in the example given in Section 3.3.2. After the transformation, input vectors need to be constructed for the new circuit, by combining both the previous and current input values of the transformed gates. For every input vector that propagates the target gate's stuck-at fault to the output pins, its previous vector is determined. The two consecutive vectors that appear on the transformed gate

are concatenated to form a new vector. By now, the suspect gates are transformed with all possible stuck-open faults represented by corresponding stuck-at faults. The second round stuck-at fault diagnosis is performed on only those transformed circuits and provides the final verdict of whether or not the suspected gates actually contain a stuck-open fault. If there is one, the stuck-open fault will be diagnosed in the names of the corresponding stuck-at faults. Matching = 100% and Prediction = 100% are expected from those fully diagnosed stuck-open faults. (For those can not be diagnosed as Matching = 100% and Prediction = 100%, special examinations will apply to confirm the individual situation. This will be introduced later on.)

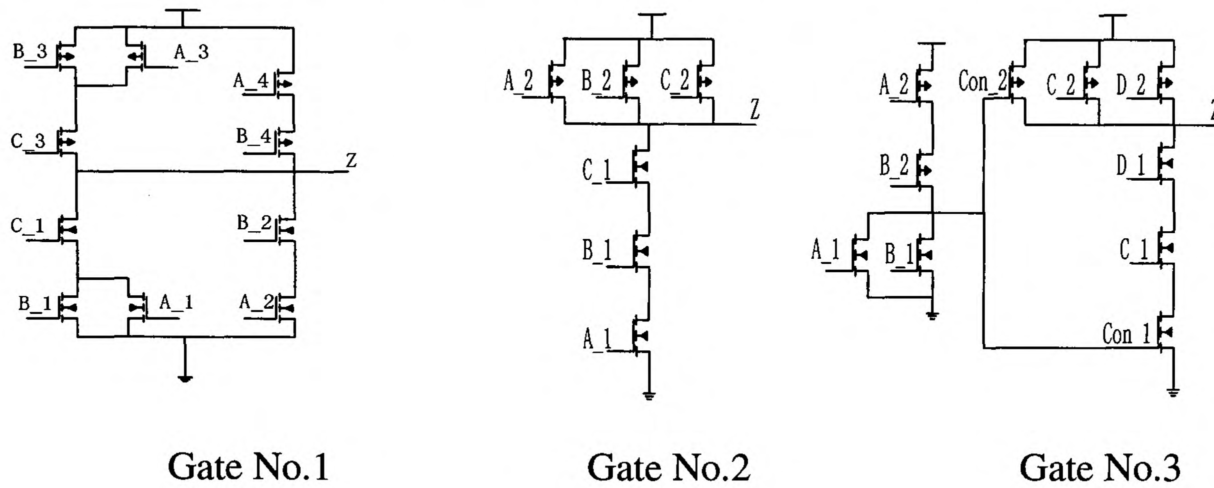
One of the benefits of having this transformation is that any stuck-at fault diagnosis tool will be able to diagnose the stuck-open fault directly and stuck-at fault diagnosis tools are commercialized and widely available. This automates the process that would otherwise have to be done manually, like creating excitation tables for each transistor stuck-open faults and modifying stuck-at fault signatures for stuck-open diagnostic purpose, matching and prediction calculation, *etc.* [Li02]

3.3.4. Implementation and experimental results

3.3.4.1. Simulation based experiment

Simulation experiments have been performed to examine the effectiveness of the stuck-open diagnosis. Based on the method, stuck-at faults with Matching = 100% and

Prediction < 100% are first selected. In Figure 3.14, three gates are assumed to be selected by the first round stuck-at diagnosis and their input vectors are given. Note that the fan-out branches are separated to each transistor to produce *A_1, A_2, etc.*



Gate No.1 Gate No.2 Gate No.3

Figure 3.14 Three experimental gates

Vectors#	Gate No.1 (ABC)		Gate No.2 (ABC)		Gate No.3 (ABCD)				
	Init	Sens	Init	Sens	Init	Sens	Vectors#	Init	Sens
1	111	000	100	000	1100	0000	17	0011	0000
2	111	001	100	001	1100	0001	18	0011	0001
3	111	010	100	010	1100	0010	19	0011	0010
4	111	011	000	011	1100	0011	20	0110	0011
5	111	100	000	100	1100	0100	21	0011	0100
6	111	101	100	101	1100	0101	22	0011	0101
7	111	110	100	110	1100	0110	23	0011	0110
8	110	111	100	111	1100	0111	24	0011	0111
9	110	000	111	000	1100	1000	25	0011	1000
10	000	001	111	001	1100	1001	26	0011	1001
11	000	010	111	010	1100	1010	27	0011	1010
12	000	011	111	011	1100	1011	28	0011	1011
13	000	100	111	100	0100	1100	29	0011	1100
14	000	101	111	101	1100	1101	30	0011	1101
15	000	110	111	110	1100	1110	31	0011	1110
16	000	111	001	111	1100	1111	32	0011	1111

Table 3.2 Initialization and sensitization vectors of experimental gates

Table 3.2 shows all possible initialization and sensitization input vector combinations of the three experimental gates. Sixteen pairs of initialization (Init) and sensitization (Sens) vectors are applied for Gate No.1 and Gate No.2, and for Gate No.3 which has four input nets, thirty-two pairs of vectors are applied. The sensitization vectors include all the possible combinations of input vectors for the three gates. The design and the parameters of the gates for electrical simulation are from Philips CMOS 0.18μm technology. The three gates are simulated by a SPICE level simulation tool and subjected to the vectors in Table 3.2. Results are taken as the real observed results (Obs) in this experiment. Every possible stuck-open defect is injected into the transistor schematics and simulated. The failed transistor is assumed to have a hard open defect. It is modeled with a resistor of 1GΩ connected to a net of the failed transistor in order that the simulator behaves properly. The vectors that fail in the electrical simulation are reported in Table 3.3. The three gates are then transformed by rules set up in the previous section, and fed into a stuck-at fault diagnosis tool to diagnose stuck-open faults. The stuck-at diagnosis tool used is a Philips internal tool – FALOC which has comprehensive gate level diagnosis features including the stuck-at diagnosis. Table 3.3 again shows the diagnosis results for the three gates.

Gate No.1			Gate No.2			Gate No.3		
Stuck-Open Transistor	Failed Vectors	Diagnosed Matching = 100% Prediction = 100%	Stuck-Open Transistor	Failed Vectors	Diagnosed Matching = 100% Prediction = 100%	Stuck-Open Transistor	Failed Vectors	Diagnosed Matching = 100% Prediction = 100%
A_1	**14**	**A_1 sa-0**	**A_1**	**8,16**	**A_1 sa-0**	**A_1**	**28**	**A_1 sa-0**
A_2	**15**	**A_2 sa-0**	**A_2**	**12**	**A_2 sa-1**	**A_2**	**4,20**	**A_2 sa-1**
A_3	**3**	**A_3 sa-1**	**B_1**	**8,16**	**B_1 sa-0**	**B_1**	**24**	**B_1 sa-0**
A_4	**2**	**A_4 sa-1**	**B_2**	**14**	**B_2 sa-1**	**B_2**	**4,20**	**B_2 sa-1**
B_1	**12**	**B_1 sa-0**	**C_1**	**8,16**	**C_1 sa-0**	**C_1**	**4,20**	**C_1 sa-0**
B_2	**15**	**B_2 sa-0**	**C_2**	**16**	**C_2 sa-1**	**C_2**	**18**	**C_2 sa-1**
B_3	**5**	**B_3 sa-1**				**D_1**	**4,20**	**D_1 sa-0**
B_4	**2**	**B_4 sa-1**				**D_2**	**19**	**D_2 sa-1**
C_1	**12,14**	**C_1 sa-0**				**Con_1**	**4,20**	**Con_1 sa-0**
C_3	**3,5**	**C_3 sa-1**				**Con_2**	**24,28, 32**	**Con_2 sa-1**

Table 3.3 Diagnosis results

For each gate, the first column indicates which transistors are supposed to be stuck open. The second column shows the failed vectors. The third column indicates whether the corresponding stuck-at faults are diagnosed as Matching = 100% and Prediction = 100%. The results show that all stuck-open faults are completely diagnosed.

In fact, the corresponding stuck-at fault will be diagnosed together with other logically equivalent faults. For example, for Gate No.1, when vectors 12 and 14 are set as the failed vectors feeding into FALOC, the report contains following faults diagnosed as

Matching = 100% and Prediction = 100%: net=*C* SA0, net=*C_3* SA0, net=*C_1* SA0. As mentioned earlier, the only stuck-at faults considered as possible are stuck-at-0 (SA0) faults in the n-transistor part and stuck-at-1 (SA1) faults in the p-transistor part. In this particular case, that includes *A_1, B_1, C_1, A_2, B_2* for SA0 and *A_3, B_3, C_3, A_4, B_4* for SA1. This is the rules made along with the transformation therefore all the other stuck-at diagnosis has to be filtered out. As to this diagnosis report, the only qualified candidate is *C_1* for SA0, the other two are against the rules and can be filtered out automatically. *C_1* stuck-open fault is exactly the right answer. Table 3.3 shows that all the stuck-open faults are diagnosed correctly by the corresponding stuck-at faults and these are the only feasible faults. Thus this simulation based experiment verifies the effectiveness of the transformation method.

3.3.4.2.Experiments based on wafer test data

In this section, experiments are performed using wafer test data from 25 wafers of a Philips 0.18μm design, which has ~500k logic gates. The same stuck-at diagnosis tool (FALOC) first diagnoses the faulty dies and suggests the likely stuck-at faulty nets. 43 dies are diagnosed as having stuck-at nets with Matching = 100% and Prediction < 100%. These results might be due to stuck-open transistors in one of these gates connected to the faulty nets and these gates are therefore taken as the targets for our transformation.

The design uses the full scan test strategy, so testing vectors are shifted in through the scan chains and applied at the normal cycle. The normal clock cycle thus has become the sensitization cycle in which the sensitization vector applies. The shift-in cycle exactly one cycle before the normal cycle thus becomes the initialization cycle in which the initialization vector applies. For all the normal cycle (sensitization) vectors that propagate the specific stuck-at fault to the primary output pins, the logic values applied on the inputs of the suspected stuck-open gates are recorded. These values will be applied at the corresponding (sensitization) inputs of the transformed circuit, for instance, inputs A, B and C in Figure 3.11. To get the initialization cycle input values on the suspected gates, vectors are traced backward, starting from the last shift in cycle, until the first instance of the input values on the suspected gates being different from the input values of the normal cycle. The speed of vectors being shifted into the scan chain is 100ns per cycle. Electrical simulation showed that the effect of a stuck-open fault can last for several microseconds. This means the effect of stuck-open fault can last for several tens of cycles if the input values of the suspected gates do not change. Therefore it makes no sense to have two identical vectors as initialization and sensitization vectors because strictly speaking the status of the stuck-open fault ultimately depends on the last vector transition. In this experiment, it turned out that no more than 10 cycles ever needed to be traced back before there was a change in the input values. These input values are taken as the previous values (like PreviousA, PreviousB, PreviousC in Figure 3.11), concatenated with current values from the normal cycle to form the new vectors for the transformed gate. Having built the new vectors for every suspected gate,

the transformed circuit and the new vectors are then compiled to be diagnosed by the stuck-at diagnosis tool again.

The 43 suspected dies were subjected to this procedure. Seven dies were fully diagnosed as having single stuck-open transistors and three of them were diagnosed by further analysis of timing skew effect or multiple stuck-open faults in one gate. The rest of the dies could not be identified as having stuck-open faults by our method. Below is the detail of the results.

#Die	Gate Type	Open Transistor	M (Matching) and P (Prediction) before transformation	M (Matching) and P (Prediction) after transformation
1	nd3	p-transistor at input C	C stuck-at-1 M = 100% P = 25%	C_4 stuck-at-1 M = 100% P = 100%
2	nd2	p-transistor at input A	A stuck-at-1 M = 100% P = 14%	A_3 stuck-at-1 M = 100% P = 100%
3	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 75%	A_1 stuck-at-0 M = 100% P = 100%
4	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 75%	A_1 stuck-at-0 M = 100% P = 100%
5	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 75%	A_1 stuck-at-0 M = 100% P = 100%
6	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 75%	A_1 stuck-at-0 M = 100% P = 100%
7	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 79%	A_1 stuck-at-0 M = 100% P = 100%

Table 3.4 Seven fully diagnosed single stuck-open faults

Successful diagnosed single stuck-open fault:

Seven dies have been diagnosed as having single stuck-open fault with Matching = 100% and Prediction = 100%. Table 3.4 shows the type of the gate and the location of open transistors. Column 4 and column 5 compare the Matching and Prediction before and after the transformation.

Figure 3.15 shows the transistor schematics of the gates and their open transistors. Note that the fan-out branches are separated to each transistor to produce A_1, A_2, *etc.* The stuck-at fault in the last column of Table 2 indicates the corresponding transistor as stuck-open. Similar to what happened in the simulation based experiment, the stuck-at fault in the last column of Table 3.4 is diagnosed together with other logically equivalent faults in the transformed circuit. The transistors are named according to the branches they lie in. For gate ao36, the report contains the following faults diagnosed with Matching = 100% and Prediction = 100%: net=*A_2* SA0, net=*A_1* SA0, net=*E_1* SA0, net=*E_4* SA0 and some extra internal nets created during the transformation.

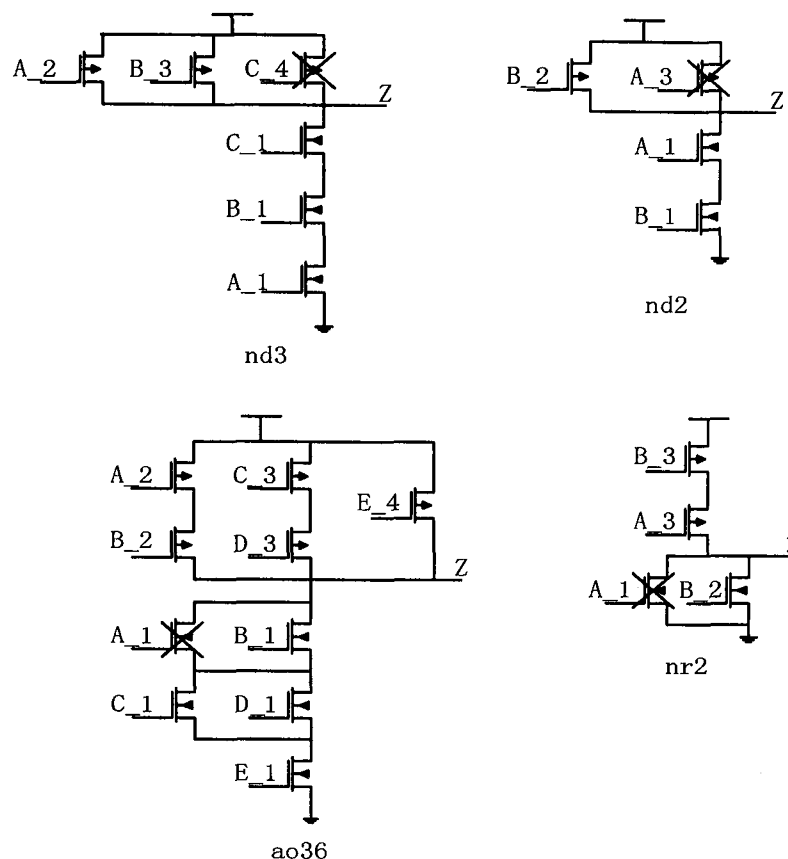


Figure 3.15 Schematics of stuck-open gates

As mentioned earlier, the only stuck-at faults considered as possible are stuck-at-0 (SA0) faults in the n-transistor part and stuck-at-1 (SA1) faults in the p-transistor part. In this particular case, that includes *A_1* and *E_1* for SA0, the rest are the redundant faults which can be removed. *E_1* is also not a realistic candidate, because if *E_1* was stuck open, *Z* would be completely detached from the ground and would thus give out Z stuck-at-1 with Matching = 100% and Prediction = 100% in the first round stuck-at fault diagnosis, rather than the result of Matching = 100% and Prediction < 100% that actually seen. The reason *E_1* is also reported is that under the assumption *E_1* is stuck-open, the previous values drawn out from the shift-in cycles are not valid in the sense that whatever the input values are, Z is always 1. Thus, in our experiment, those transistors that can singly decide the conducting of p-transistor part or n-transistor part are not considered as valid stuck-open candidates. As a result, *A_1* SA0 remains to be the only valid diagnosis result, indicating *A_1* is the open transistor.

Stuck-open faults affected by timing skew:

The timing skew problem can arise when two inputs change their values between the initialization vector and the sensitization vector. Different intermediate vectors may be formed depending on the timing of the transition. When the timing skew problem takes control, the real initialization vectors at the inputs of the suspected gates are in fact the intermediate values instead of the values drawn from the shift in cycles. In some cases, perfect diagnosis is not achieved but Matching < 100% and Prediction = 100% is seen after the transformation. This means some failing bits observed on the tester (in *Obs*) are not predicted by the fault model. One of the possible explanations is that the timing skew problem has invalidated some of the vectors that would otherwise be able to meet the initialization requirement. This successfully explains two gates in Table 3.5 (both are nr2 in Figure 3.15). For these two gates, values at the inputs of the gate are examined and the key finding is that the loss of Matching is always due to a particular sequence of input vectors, {initialization AB=01; sensitization AB=10}. Assuming B transitions earlier than A, *i.e.* the intermediate value is AB=00, the original sequence can be replaced with the effective sequence {initialization AB=00, sensitization AB=10}. The transformed circuit was simulated again with this new sequence to perform stuck-at fault diagnosis. Under this assumption, both gates give Matching = 100% and Prediction = 100% as shown in the last column of Table 3.5. A similar phenomenon is also reported in [Li02]. However, without detailed timing simulation, there is no particular reason to assume that the B transitions earlier than A rather than *vice versa.* The timing skew effect can be far more complicated than the two samples

explained here. More vector sequences can be affected when the number of inputs increases, influencing both the Matching and Prediction.

#die	Gate type	Open Transistor	M and P before transformation	M and P after transformation	M and P after timing skew modification
8	nr2	n-transistor at input A	A stuck-at-1 M=100%P=53%	A_1 stuck-at-1 M=60%P=100%	A_1 stuck-at-1 M=100%P = 100%
9	nr2	n-transistor at input A	A stuck-at-1 M=100%P=52%	A_1 stuck-at-1 M=80%P=100%	A_1 stuck-at-1 M=100% P =100%

Table 3.5 Stuck-open faults affected by timing skew

Another more informed way of judging the transition order is to perform exact timing simulation of the whole circuit, but this is extremely time consuming. The accuracy of the timing simulation largely depends on how closely the delay specification of the library cell actually matches with the real situation. There are also other issues such as the coupling effect between the neighbouring lines that are not taken into account by the timing simulation but do have impact on the timing behaviour of the circuit. Therefore one can not guarantee that the transition order derived from the timing simulation will always be correct and the value of performing timing simulation to inform the transition order is questionable.

Diagnosis result of multiple stuck-open faults in one gate

There is one sample, named die #10, for which only the output of a suspected gate is diagnosed as stuck-at-0 Matching = 100%, Prediction < 100% in the initial stuck-at fault diagnosis (Figure 3.16.A). The gate is transformed and diagnosed again. Two

input nets, *C_3* and *D_4*, report Matching < 100%, Prediction = 100% after the transformation.

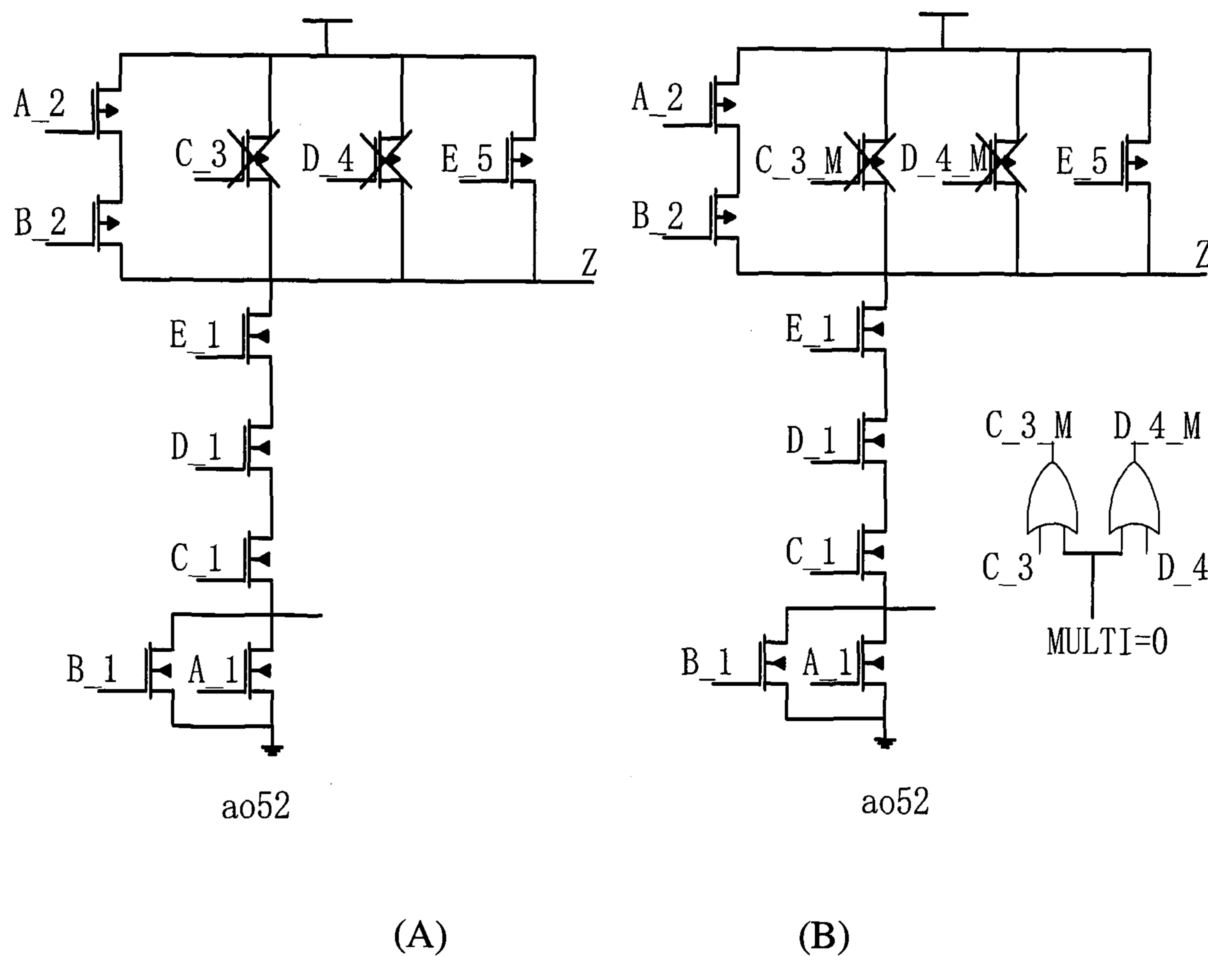


Figure 3.16 Multiple stuck-open faults

Some thought reveals that multiple stuck-open faults could lead to a phenomenon like this. With only one fault modeled, some of the failing bits in *Obs* caused by the other fault(s) and those only caused by the simultaneous presence of multiple faults are not predicted in *Sim*, leading to a low Matching for both of the single fault diagnosis candidates.

The transistors *C_3* and *D_4* are the main suspects of multiple stuck-open faults in this particular gate. To confirm the suspicion, the multiple stuck-open effect has to be modelled in the transformed circuits, *i.e.* in the form of multiple stuck-at faults. A small

additional circuit controlled by signal MULTI is devised to represent the multiple stuck-at effects in the transformed circuit, shown as Figure 3.16.B. The injected signal 'MULTI' is set to ZERO and therefore does not influence the normal behaviour of the gate. If 'MULTI' is stuck-at-1, both *C_3_M* and *D_4_M* are also stuck-at-1, representing the effect of multiple stuck-open of p-transistor C_3 and D_4. The same strategy also applies for the multiple n-transistors opens, with the OR gate replaced by AND gate and 'MULTI' given the input value of ONE instead.

After this modification, the circuit is applied with another round of stuck-at fault diagnosis. 'MULTI' is diagnosed as stuck-at-1 with Matching = 100% and Prediction = 100%, indicating p-transistor C_3 and D_4 are both stuck-open. Results before and after the transformation are given in Table 3.6.

#die	Gate type	Open Transistor	M and P before transformation	M and P after transformation	M and P after multiple stuck-open modification
10	ao52	n-transistor at input C and D	Z stuck-at-0 M = 100% P = 22%	C_3 stuck-at-1 M = 29% P = 100% D_4 stuck-at-1 M = 57% P = 100%	MULTI stuck-at-1 M = 100% P = 100%

Table 3.6 Multiple stuck-open faults in one gate

To further investigate if this diagnosis is realistic, layout of gate ao52 is analyzed and it reveals a possible explanation of the multiple open. An open contact illustrated in Figure 3.17 can disconnect both p-transistor C_3 and D_4 from the output Z.

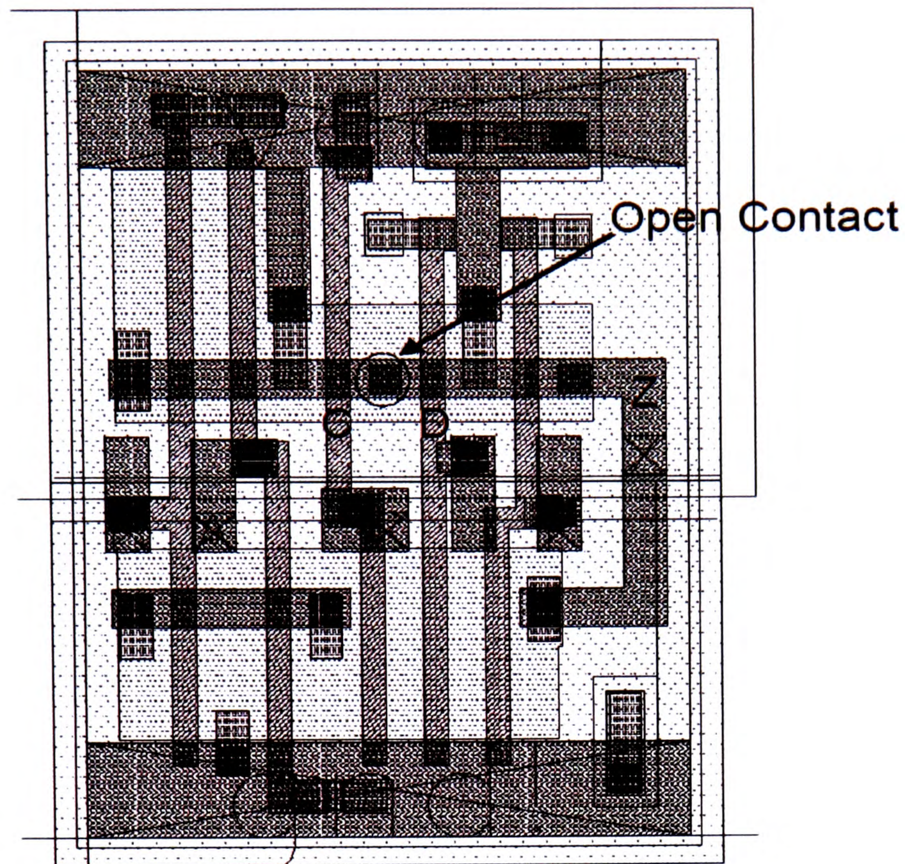
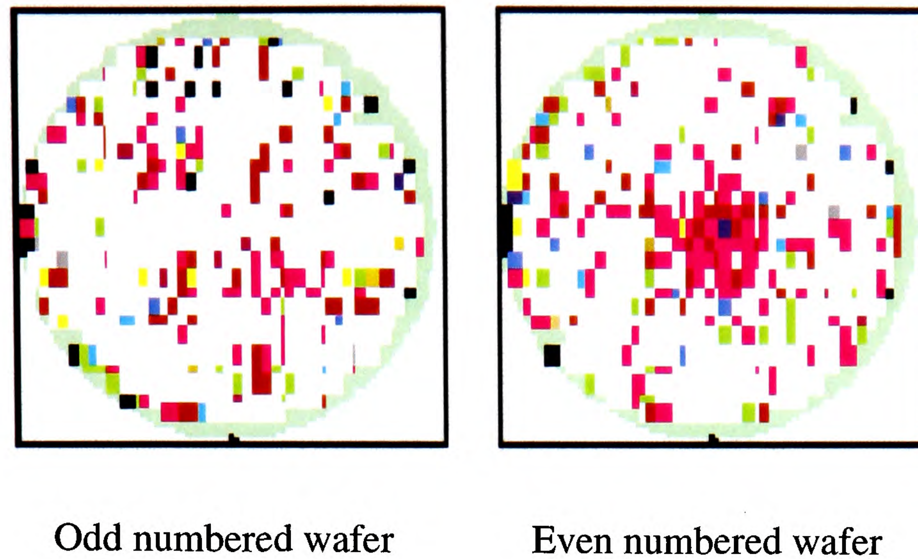


Figure 3.17 Layout analysis of ao52

Some further corroboration:

Unfortunatley the ten dies diagnosed above are no longer available to perform FA to confirm the stuck-open faults diagnosed above. However, after obtaining the diagnosis results, analysis was done on the process data that is still available. It turns out that this is a lot which shows an odd-even effect: The odd numbered wafers have scattered fails across the wafer, but in all the even numbered wafers, faulty dies cluster in the centre of the wafer as shown in Figure 3.18.



Odd numbered wafer Even numbered wafer

Figure 3.18 Wafer map

In-line inspection on one of the even number wafers revealed patterning problems in the centre of the wafer that had previously been found to result in open contact/via defects, as in the example shown in Figure 3.19. It turns out that all the samples diagnosed above belonged to even number wafers and that they were located in the centre of the wafers with known open contact problems. This strong correlation of our diagnosis results and the in-line inspection provides a useful corroboration of our work.

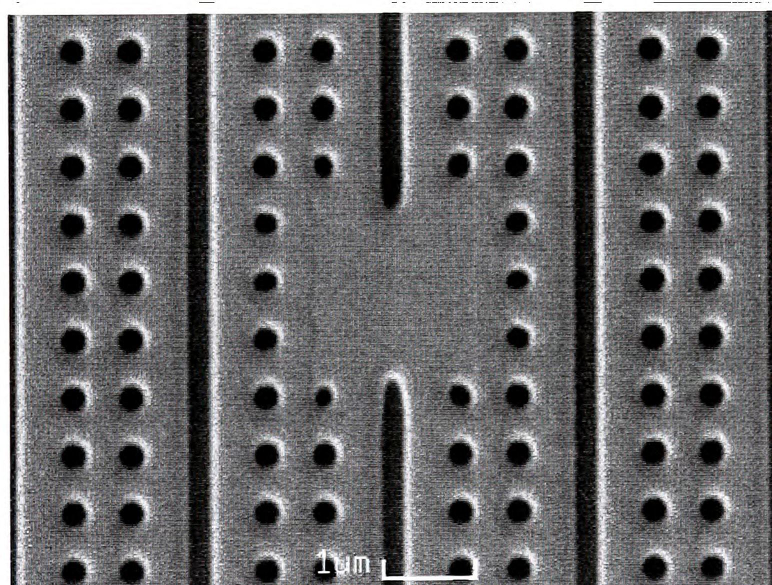


Figure 3.19 Missing Patterning

3.3.5. Consecutive high impedance vectors

High impedance vectors are defined here as those input vectors that result in a high impedance status at the output under the presence of a particular stuck-open fault, *i.e.* those vectors that meet the sensitization requirement for stuck-open.

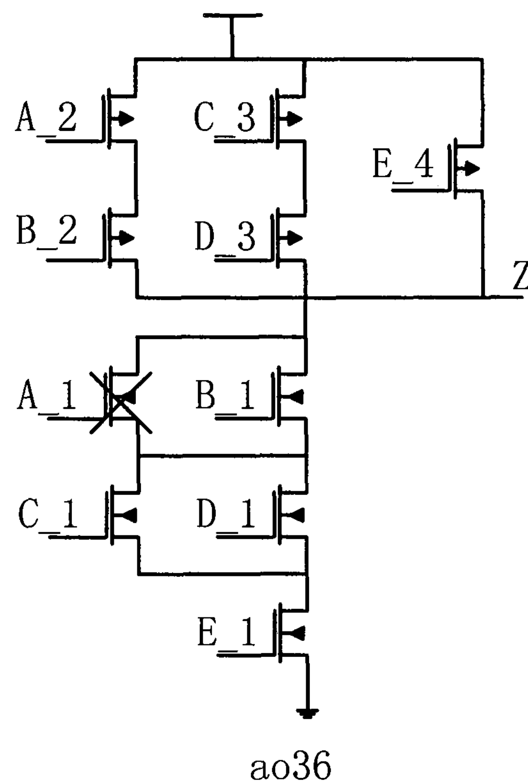


Figure 3.20 Gate ao36 with stuck-open fault at transistor A_1

For instance, the gate ao36 in Figure 3.20, suppose A_1 is a stuck-open transistor. Vector ABCDE = 10111 is a valid sensitization vector. When applied, neither the p-transistor part nor the n-transistor part drive the output Z because the stuck-open transistor A_1 controls the only conducting path from GND to Z. As a result Z keeps at the high-impedance status. A high-impedance vector is a valid sensitization vector for that particular stuck-open. The problem is that there are sometimes more than one high-impedance vector. For instance, vectors ABCDE = 10101 and ABCDE =10011 are also high-impedance vectors. If two of these vectors are applied to the gate in a consecutive way, it is no longer correct to take the first vector as the initialization vector

as it does not set the output to a particular value but lets it be high-impedance. This means that the strategy of stopping tracing back in the shift-in cycle once the first different vector appears may pose a risk of having consecutive high-impedance vectors.

While the consecutive high impedance patterns issue does not affect the successfully diagnosed dies introduced in earlier sections, it does influence the diagnosis result of one particular device from Philips Vector4 design. For this particular device, six different sets of test vectors are applied. The device fails on five of the test vector sets and passes the other one. The whole diagnosis process starts as usual with a first round stuck-at fault diagnosis on the circuit scale.

Vector set	Number of failing bits	First run stuck-at diagnosis
1	56	M = 100% P = 58%
2	Pass	----
3	61	M = 100% P = 61%
4	19	M = 100% P = 27%
5	59	M = 100% P = 47%
6	45	M = 100% P =75%

Table 3.7 Result from the first run stuck-at diagnosis

Table 3.7 gives the number of failing bits of each test vector set and the first round stuck-at diagnosis results. The first round stuck-at diagnosis of the six test vector sets all point to a particular gate shown in Figure 3.21, with Matching = 100%, Prediction < 100%. Therefore this gate is taken as the primary suspect to be transformed. The new input vectors are created and then applied on the transformed circuit to perform the

second round stuck-at fault diagnosis.

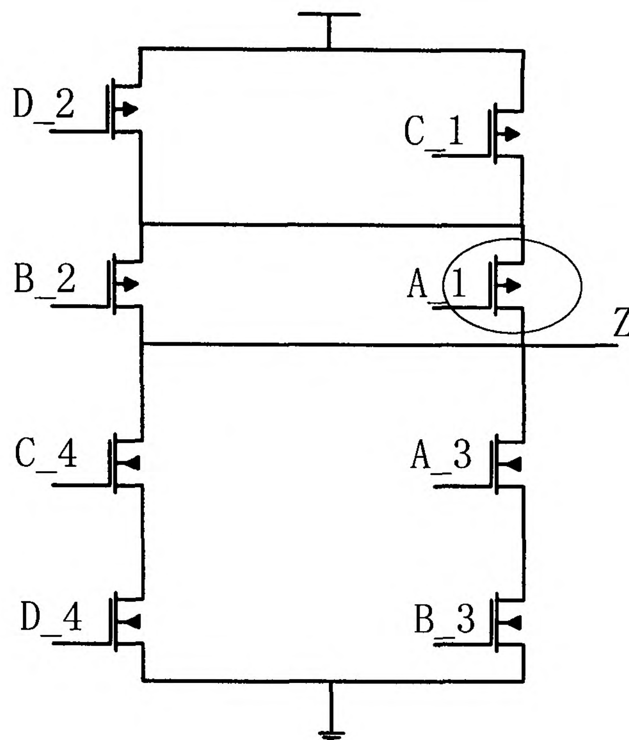


Figure 3.21 Suspected gate

The last column of Table 3.8 shows the second round stuck-at fault diagnosis after the transformation. A clear pattern of low Matching but high Prediction is seen in the second round diagnosis. One possible explanation for this observation is that the gate has consecutive high impedance vectors.

Vector set	Number of failing bits	First run stuck-at diagnosis	Normal stuck-open transformation
1	56	M = 100% P = 58%	M=19% P=100%
2	Pass	----	----
3	61	M = 100% P = 61%	M=86% P=100%
4	19	M = 100% P = 27%	M=57% P=100%
5	59	M = 100% P = 47%	M=79% P=100%
6	45	M = 100% P =75%	M=68% P=100%

Table 3.8 Diagnosis result after normal stuck-open transformation

The two consecutive high-impedance vectors are taken as a pair of initialization vector and sensitization vector while actually the first vector does not initialize the circuit properly. For instance, assume there is a sequence of vectors applied on this gate ABCD = {1100, 0100, 0101}, as shown in Figure 3.22. In the normal procedure, the normal cycle 0101 will be selected as the sensitization vector and the one cycle before it – cycle#-1 will be selected as the initialization vector. 0100 does not meet the stuck-open initialization vector requirement therefore this sequence of test vectors does not predict a failing bit in *Sim*. However what really happens is 0100 is also a high-impedance vector thus the output value depends on the previous vector – cycle#-2. 1100 is a valid initialization vector which sets output to 0. The above vector sequence will fail the test. Therefore this sequence will be a failing bit in *Obs* but not *Sim*, resulting in a loss of Matching. This could explain the low Matching but high Prediction phenomenon in the second round stuck-at fault diagnosis.

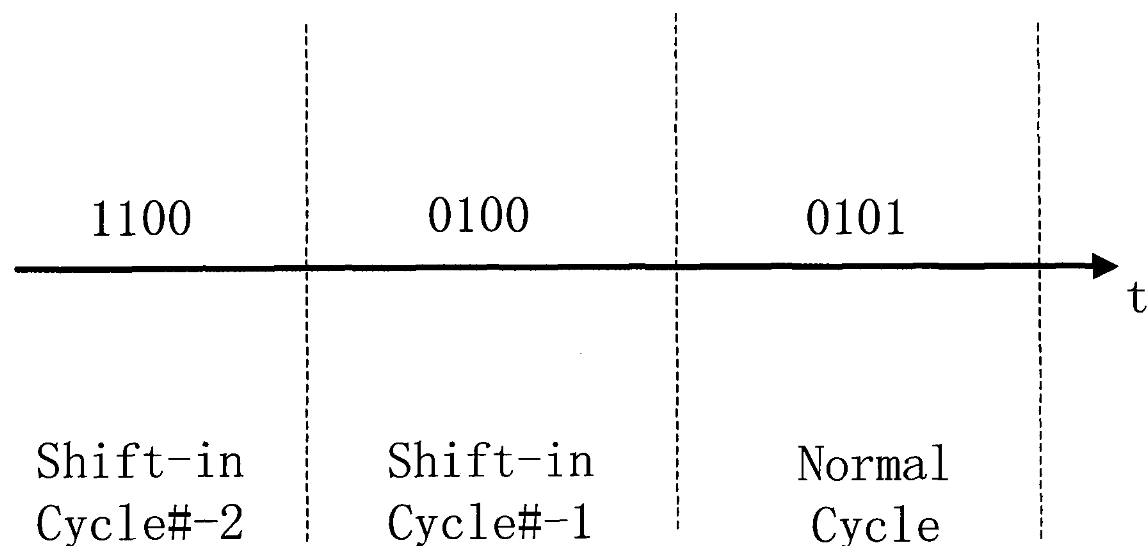
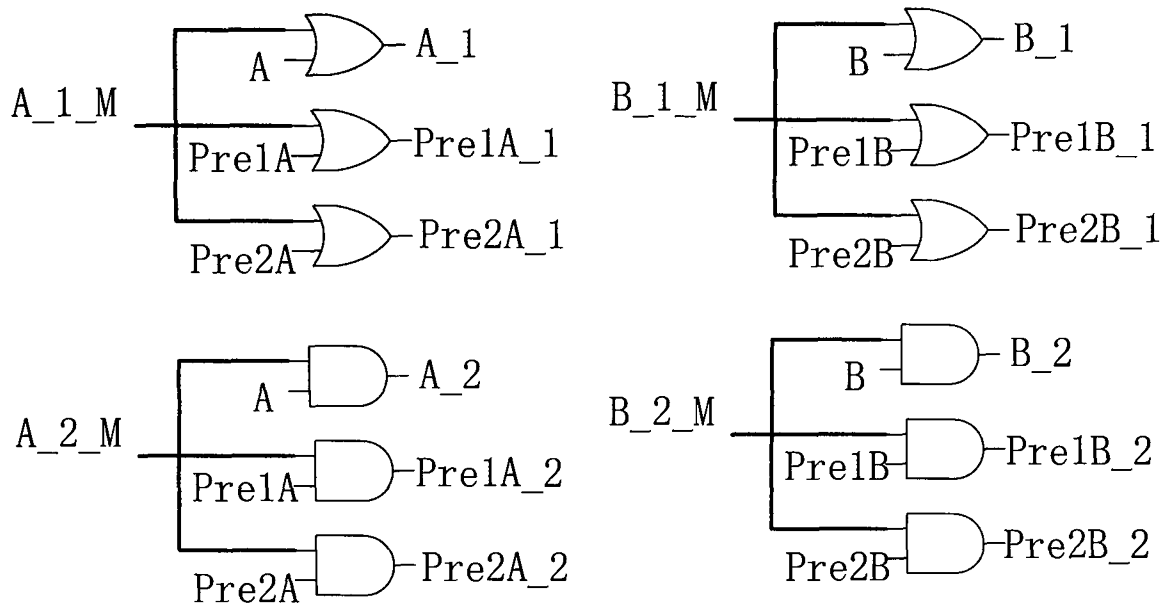


Figure 3.22 A sequence of test vectors

The transformation method introduced earlier is not able to handle this complication. However, the problem can be solved by an extended version of transformation which traces further back into shift-in cycle until the first non high impedance vector is found.

Figure 3.23 shows the extended version transformation.



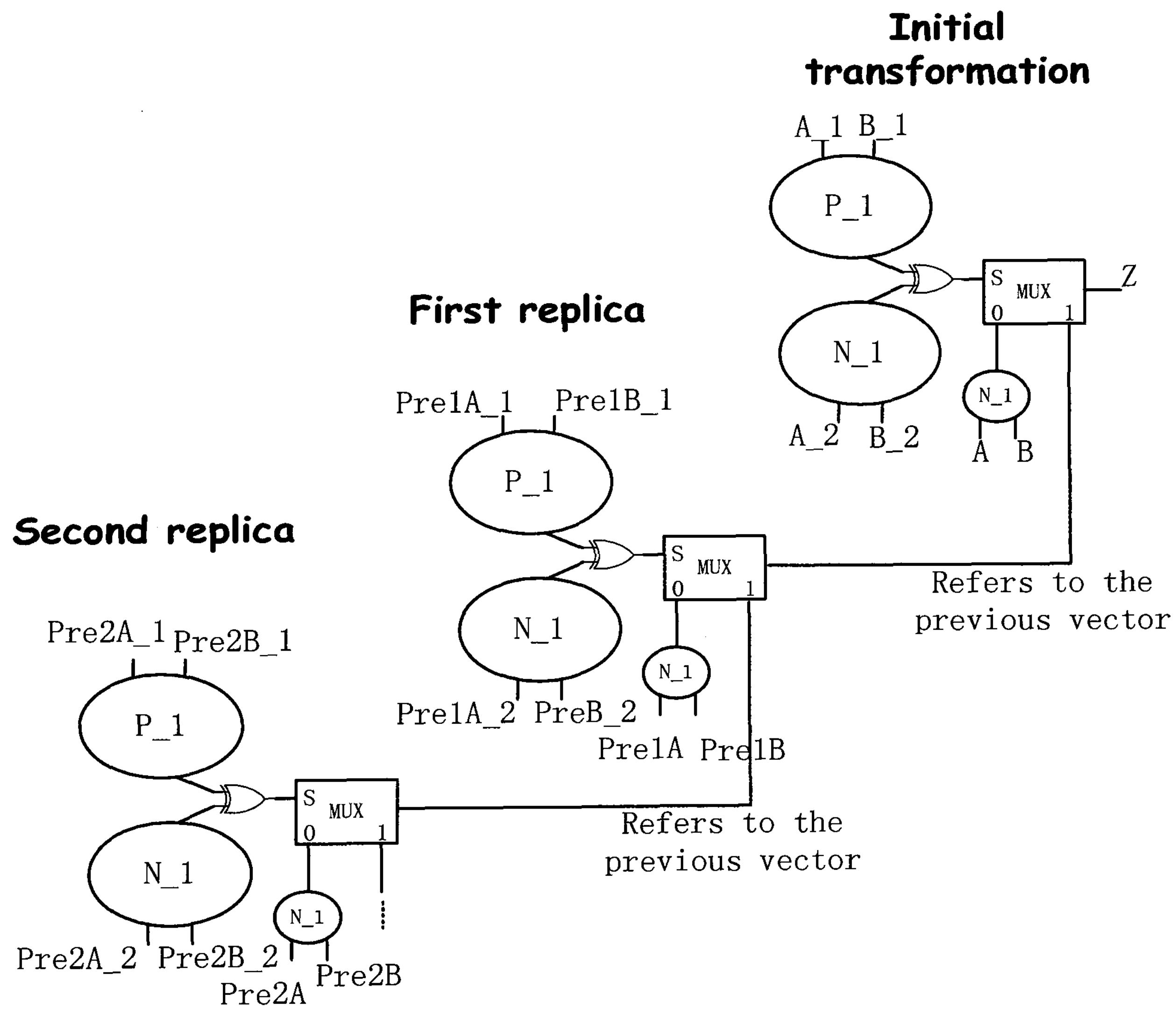


Figure 3.23 An extended version of transformation for consecutive high-impedance issue

The basic idea of this extended version transformation is to replicate the initial version of the transformation so that the replicated transformation circuit can be used to determine if the previous vector (Pre1) is also a high-impedance vector. If it is, the next replica is used to determine if the vector previous to last one (Pre2) is a high-impedance vector and so on.

The stuck-open effect should be seen on each of the replicas therefore signals like A_1_M, A_2_M, B_1_M and B_2_M are designed to represent the stuck-open faults in the extended transformation. A_1_M and B_1_M are designed to represent the stuck-open on p-transistors, they are set to 0 in the fault free case therefore all the previous vector values like A, Pre1A, Pre2A, B, Pre1B, Pre2B are passed through to the A_1, Pre1A_1, Pre2A_1, B_1, Pre1B_1 and Pre1B_2 and fed into the replicas respectively. The correct logic of the circuit is not affected in the fault free case. When, for example, A_1_M is stuck-at-1, A_1, Pre1A_1, Pre2A_1 are all stuck-at-1, each representing the p-transistor as stuck-open in the corresponding replica. Therefore A_1_M stuck-at-1 will be representing the p-transistor stuck-open fault in this extended transformation. The same arguments apply for the n-transistor too.

When the normal cycle vector is not a high impedance vector the Mux in the initial transformation selects the normal cycle value which would be fault free. When the normal cycle vector is a high impedance vector, the Mux refers to the previous vector which is generated by the first replica. The first replica examines if the shift-in cycle#-1

is a high-impedance vector. If it is, the Mux of the first replica puts the value of shift-in cycle#-1 through to the request from the Mux of the initial transformation. If not, the Mux refers to the shift-in cycle#-2, and this process goes on as necessary. One obvious question raised here is how many replicas need to be made that can be regarded as sufficient. The real answer relates to how long the stuck-open fault effect can be sustained in the shift-in cycles. Electrical simulation shows the stuck-open fault effect can last for several microseconds, while the speed of shift-in is 100ns per cycle. Therefore, in the limit, several tens of the replicas have to be made. However, examination of a number of examples shows that the consecutive high-impedance vectors rarely lasts for more than ten cycles.

Going back to the example of Table 3.8 and Figure 3.21, this particular gate is transformed again up to ten replicas. It is then subjected to stuck-at diagnosis again. Results in Table 3.9 show that after the extended transformation Matching (M) has been increased substantially but has not achieved 100%.

Vector set	Number of failing bits	First run stuck-at diagnosis	Initial transformation	Extended transformation
1	56	M = 100% P = 58%	M=19% P=100%	M=69% P=100%
2	Pass	----	----	----
3	61	M = 100% P = 61%	M=86% P=100%	M=93% P=100%
4	19	M = 100% P = 27%	M=57% P=100%	M=89% P=100%
5	59	M = 100% P = 47%	M=79% P=100%	M=95% P=100%
6	45	M = 100% P =75%	M=68% P=100%	M=84% P=100%

Table 3.9 Diagnosis results after the extended transformation

This non-100% Matching results might be explained by the timing skew problem which generates intermediate vectors that serve as valid initialization vectors and therefore brings more failing bits than predicted by the extended transformation. This is only one speculation of what could cause non-100% Matching. Nonetheless, the clear improvement in Matching from the initial transformation to the extended transformation indicates the likelihood of having a stuck-open fault at transistor A_1 in Figure 3.21 with high-impedance complications.

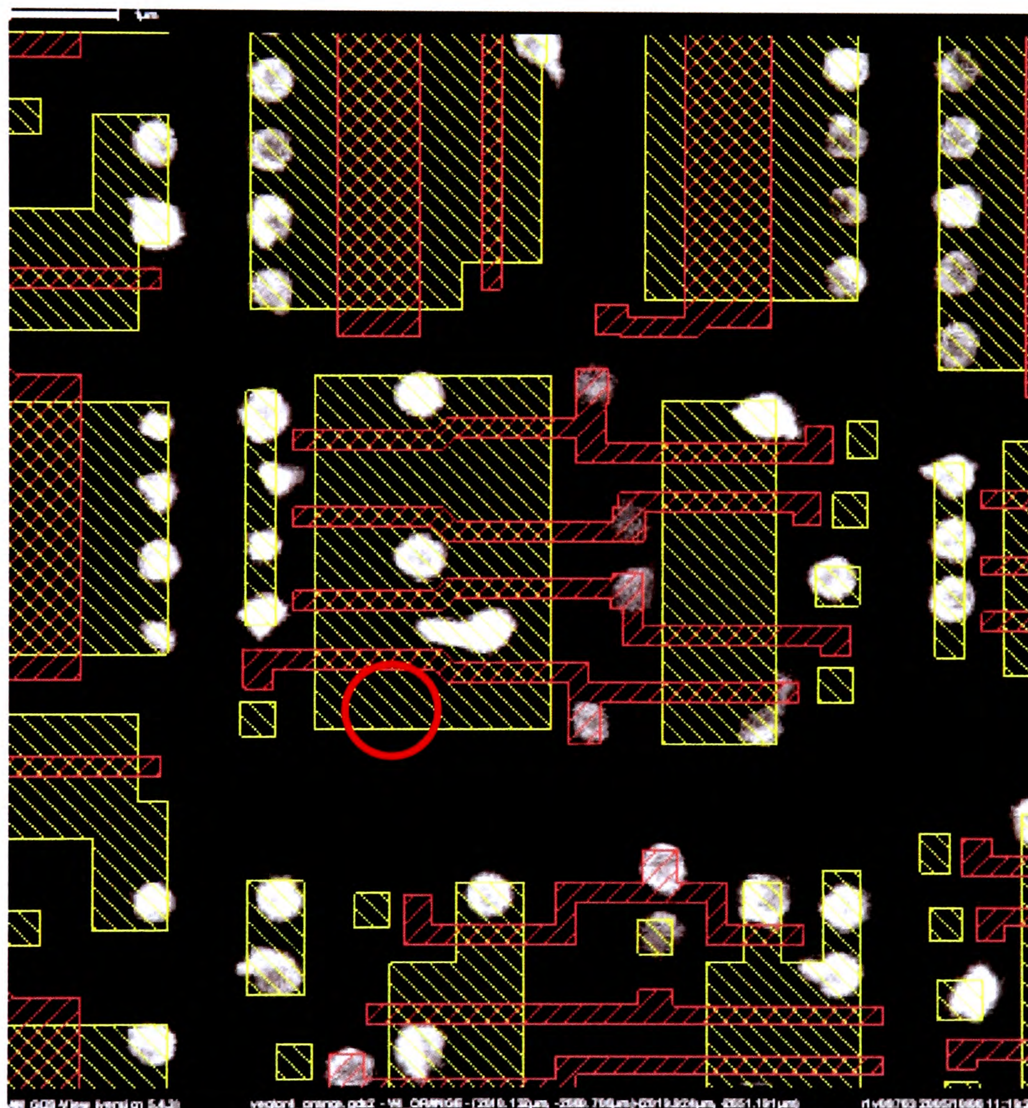


Figure 3.24 FA result of stuck-open device from Philips FA lab

To confirm this diagnosis, failure analysis (FA) was performed on this device. After de-processing to contact level, FA revealed the absence of the top contact on exactly the

transistor diagnosed by the extended transformation method (the missing white spot on the location circled in red).

3.3.6. Short summary

To diagnose stuck-open faults, a method of transforming from transistor level schematic to a logic gate level description has been proposed. By this method, a stuck-open fault in the transistor level can be represented by a stuck-at fault at the logic gate level. A commercial stuck-at diagnosis tool can be brought in to pick out the stuck-open fault directly. This is the most efficient and cost-effective method that has been introduced for stuck-open fault diagnosis so far. Also, because only those cells identified by a first-pass stuck-at fault diagnosis are transformed, the method is independent of the size of the circuit, which means that little extra time is spent on the diagnosis. The method is also adaptable to diagnosis of open affected by timing skew and to the diagnosis of multiple stuck-open faults and to incorporate consecutive high-impedance vectors situation.

3.4. Intra-gate resistive open fault diagnosis

Section 3.1 developed a new method for the diagnosis of hard stuck-open faults, but it is known that resistive opens also occur. A different method needs to be developed to diagnose the intra-gate resistive open fault in order to incorporate the delay effects

caused. A resistive open is defined as an imperfect connection that can be modelled as a resistor. Since CMOS circuits are charge controlled, resistive opens increase circuit delays and raise timing related issues [Li01] [Tahoori02]. It has been shown that the delay increases linearly with the open resistance value [Li03]. As discussed in Section 2.3.3.4, resistive open defects cause slow-to-rise and slow-to-fall faults [Li01] [Krishnamachary02]. The problem of testing and diagnosis of resistive open defects therefore fits in the scope of delay fault testing.

3.4.1. Delay fault testing

Delay faults model defects that affect (most often slow down) the timing behaviour of a combinational circuit without changing their functionality [Sivaraman98]. In order to trigger the effect of a delay fault, two consecutive test vectors <V1, V2> must be applied [Abramovici90]. The test scheme basically works as Figure 3.25, where vector V_1 is applied at t_1 to set the combinational circuit at a certain stable status, and V_2 is applied at t_2 to sensitize the delay fault. The test response is latched into the scan chain at $t_2 + T_c$ and shifted out to compare with the faulty free response: once different, the delay fault is detected. A fault-free circuit has its normal delay as well, but it should be smaller than T_c. The normal delay plus the delay added by defects causes the circuit to stabilize after the sampling time $t_2 + T_c$, in which case it fails in the delay testing.

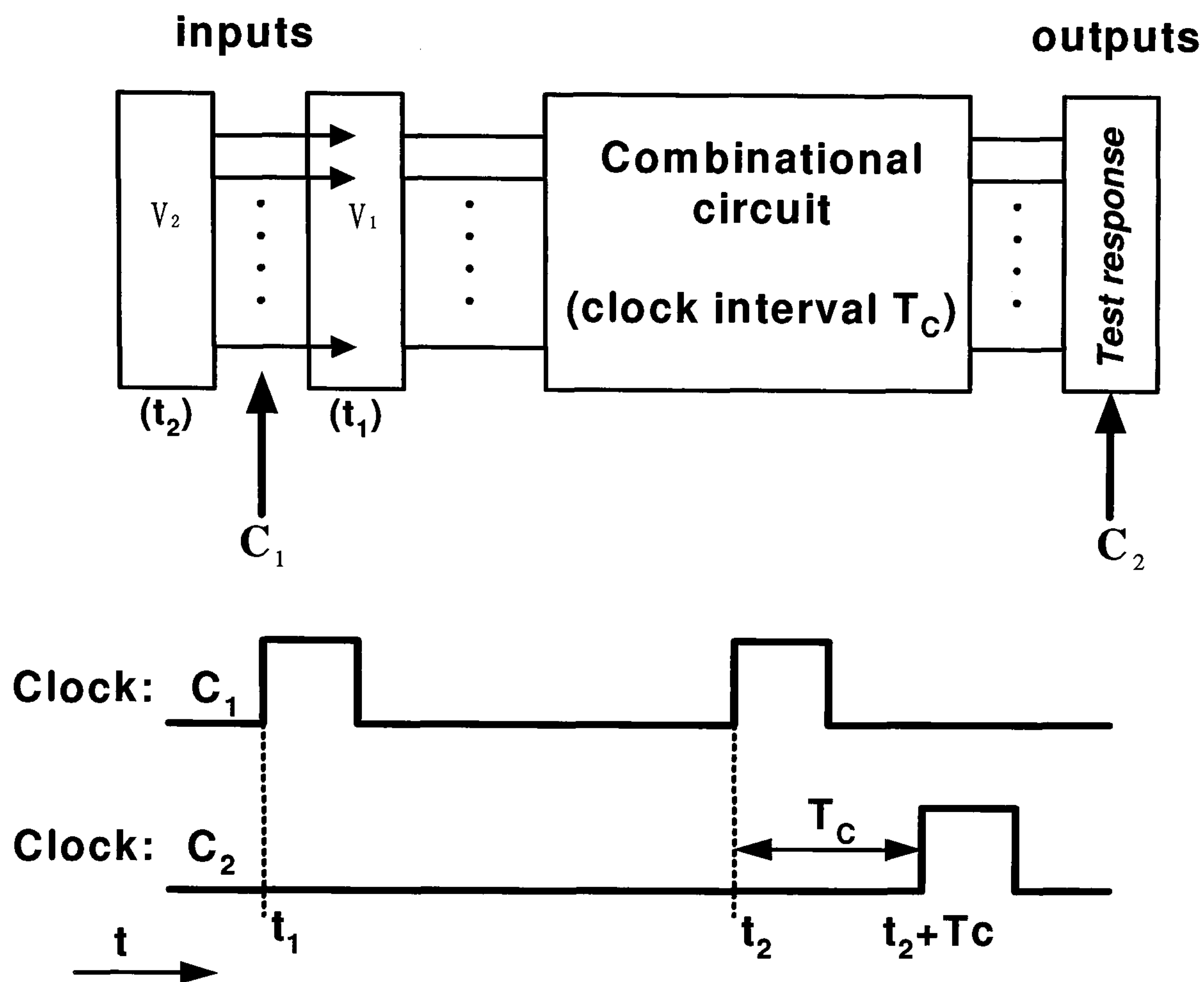


Figure 3.25 Basic structure of delay fault testing

An important and widely used delay fault model is the transition fault model [Waicukauski87], where the delay fault is assumed big enough that all paths through this gate will carry the effects to the outputs and break the timing constraint at the outputs. This idea has intrinsic connection with the stuck-at fault model in that the second vector V2 is exactly a stuck-at vector: a slow-to-rise transition fault can be modelled as a stuck-at-0 and a slow-to-fall transition fault can be modelled as a stuck-at-1. The difference between the stuck-at model and transition delay model is that transition delay model needs one vector to initialize the value of the faulty node, i.e. 0 for the slow-to-rise and 1 for the slow-to fall, while the stuck-at model does not.

Very much like the way the stuck-open fault signature is composed, transition delay

fault signatures are constructed by examining a sequence of two vectors, which makes sure there is a transition on the faulty net and that the second vector sensitizes and propagates the stuck-at fault to the output. *Sim* is formed by these pairs of failing vectors and compared with *Obs* of a delay fault testing. The same diagnostic procedure applies and gate level transition delay fault diagnosis tools are well developed. The following section discusses how to utilise the gate level transition delay fault diagnosis tools to diagnose intra-gate resistive open faults which manifest as transitional delay faults.

3.4.2. Intra-gate resistive open fault diagnosis

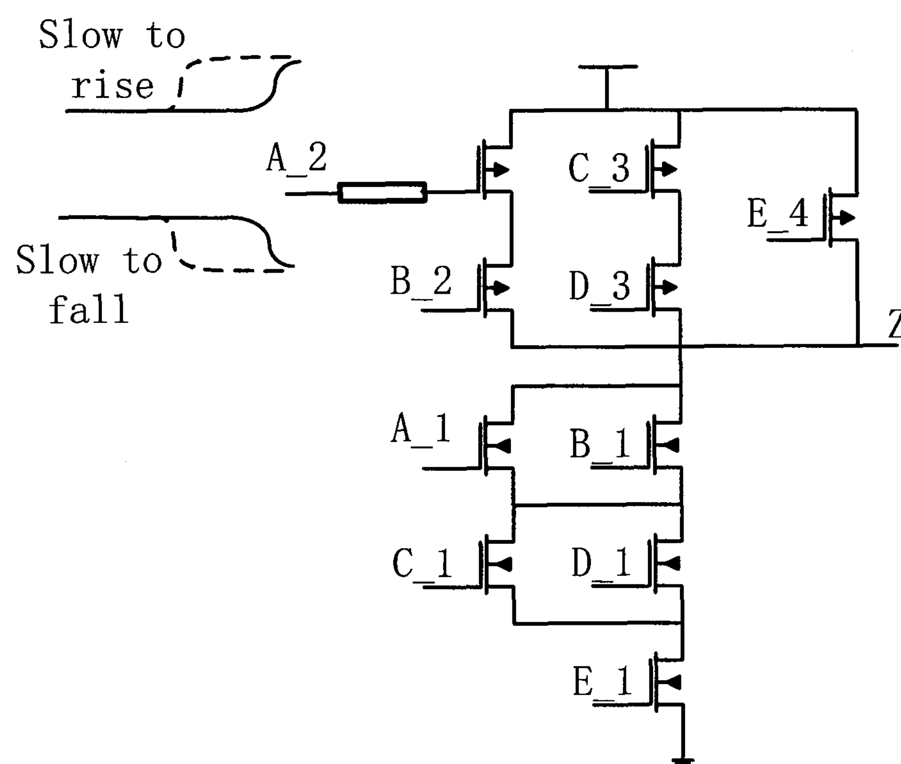


Figure 3.26 Transitional delay fault caused by resistive open at A_2

Figure 3.26 shows a gate with a resistive open at transistor A_2. This open has caused a transitional delay fault at its input net for both slow-to-rise and slow-to-fall faults. This has a direct effect on the p-transistor itself, causing it to be slow-to-turn-off and

slow-to-turn-on respectively. These complementary effects are considered separately below.

The slow-to-turn-on effect has much similarity with the stuck-open fault model in that they both keep the faulty transistor from conducting. Both require an initialization vector to set the output to the complementary value before a failing bit can be observed. The extra requirement of the slow-to-turn-on effect is that the initialization vector also has to make sure there is a transition at the faulty gate when it changes to the second vector, otherwise no delay fault appears at the transistor and no fault can be observed at the output. This requirement is inherently examined in transition delay fault diagnosis tools, therefore the stuck-open transformation of Section 3.1 can be applied directly to represent the slow-to-turn-on effect with the transition delay fault in the gate level.

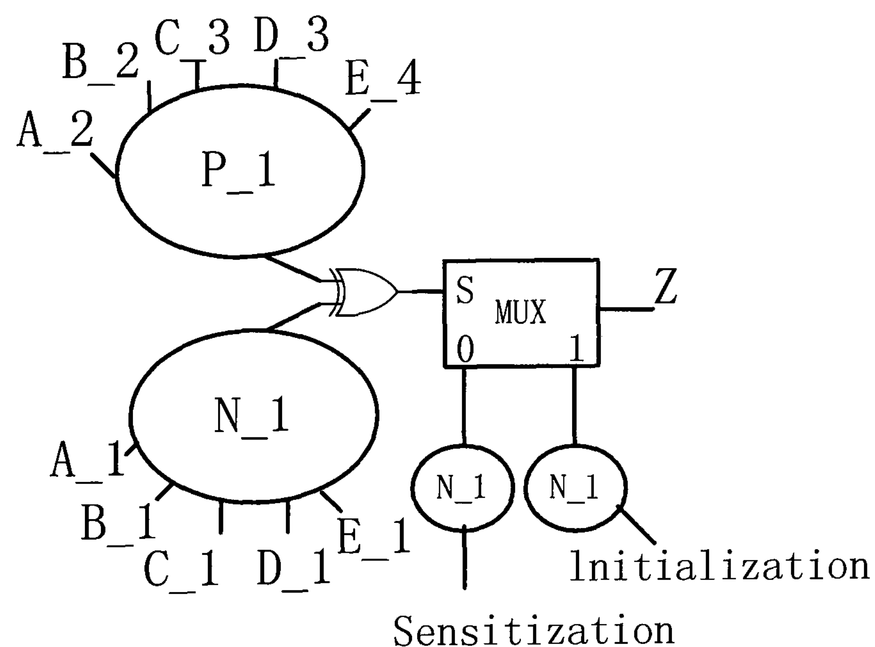


Figure 3.27 Transformation of slow-to-turn-on

Figure 3.27 transforms the gate of Figure 3.26 in the same way as a stuck-open transformation. Each slow-to-0 (STO) fault of those input nets connected to

p-transistors (A_2, B_2, C_3, D_3, E_4) and each slow-to-1 (ST1) fault of those input nets connected to n-transistors (A_1, B_1, C_1, D_1, E_1) represents the slow-to-turn-on effect of the relevant transistor. The slow-to-turn-off effect literally keeps one unwanted transistor conducting temporarily. It will not affect the output unless there is a conducting path through this transistor, nor will it affect the output if there are other conducting paths that could replace the effect of this transistor. Therefore, the necessary condition for this slow-to-turn-off effect to be seen at the output is that the temporary conducting transistor **creates** the **only path** from either the Vdd (if it is a p-transistor) or Gnd (if it is an n-transistor) to the output. Remember that the transformation rules set for the stuck-open fault in Section 3.1 also indicate that the faulty transistor **cut off** the **only conducting path**. This similarity means the same transformation can be implemented to represent the slow-to-turn-off effect, Figure 3.28.

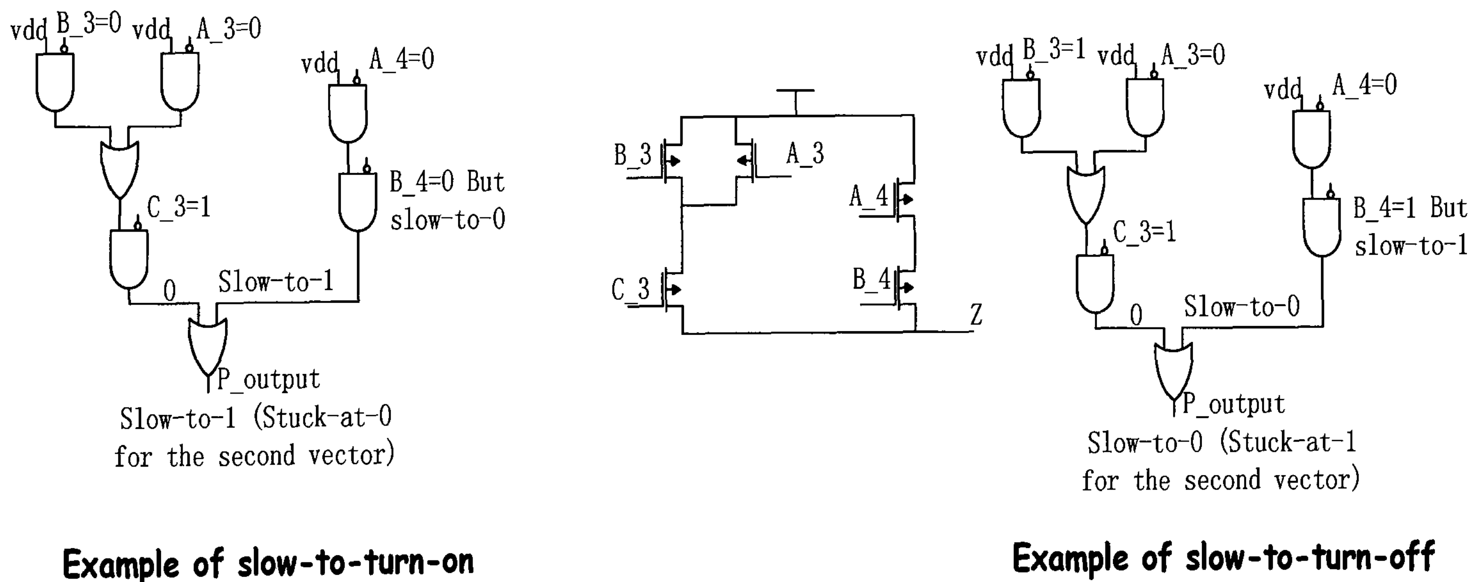


Figure 3.28 Comparison of slow-to-turn-on and slow to turn-off

The only difference is the slow-to-turn-off effect is represented by each ST1 fault at transistor input. However there is a major difference in how the slow-to-turn-off effect

behaves. The faulty effect of slow-to-turn-off comes from this extra conducting path competing with the conducting path on the other side. If the extra conducting path wins, a faulty effect can be observed, if not, the gate behaves as if it was fault free. Unlike the slow-to-turn-on fault, electrical simulation shows that this kind of competing has little to do with the previous output value set by the initialization vector, *i.e.,* it does not require the initialization vector to set the output to the complementary value, therefore the stuck-open like transformation in Figure 3.27 is not applicable.

As resistive opens cause both slow-to-turn-on and slow-to-turn-off faults, the difficulty now lies in how to integrate and represent the two types of faults in one schematic. The solution is given in Figure 3.29.

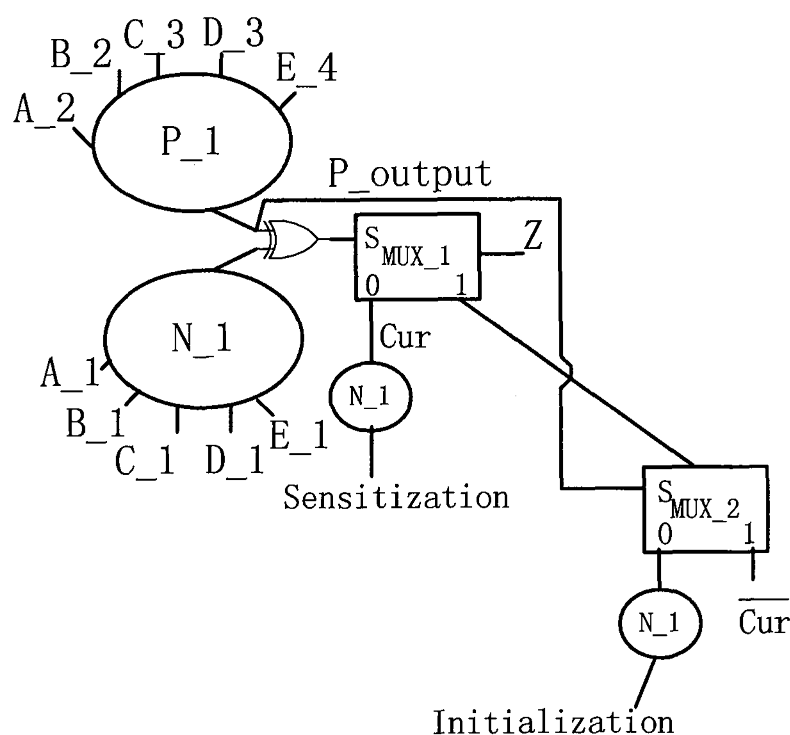


Figure 3.29 Integrated transformation for intra-gate transitional delay fault

As illustrated in Figure 3.28, when a slow-to-turn-on fault takes effect, the second vector always puts the P_output to stuck-at-0, whereas slow-to-turn-off fault always

puts the P_output to stuck-at-1. This is an indicator of exactly what type of transition faults happens and it controls the MUX_2 that replaces the initialization part of Figure 3.27. When a fault occurs, if the P_output is stuck-at-0, *i.e.*, slow-to-turn-on happens, the initialization part is passed on, resulting in the same circuit as Figure 3.27. If the P_output is stuck-at-1, *i.e.*, a slow-to-turn-off fault, the opposite value of Cur (Cur is the output value of sensitization vector) is passed on by MUX_2 and selected by MUX_1, resulting in a fail at the output. This is in line with the behaviour of slow-to-turn-off which fails regardless of the previous output value as long as the fault is triggered.

Simulation based experiments were performed on the gate of Figure 3.26, using the transition delay fault diagnosis function of FALOC. 5M Ω resistance is inserted on each transistor's gate connection in turn and simulated. The simulation results are subject to transitional delay fault diagnosis on the transformed circuit.

Transistors	Typed of fault discovered by simulation	Diagnosed with Matching=100% and Prediction=100%
A_1, B_1, C_1, D_1, E_1, E_4	**Slow-to-turn-on, Slow-to-turn-off**	**A_1 ST1&ST0, B_1 ST1&ST0, C_1 ST1&ST0, D_1 ST1&ST0, E_1 ST1&ST0, E_4 ST0&ST1,**
A_2, B_2, C_3 D_3	**Slow-to-turn-on**	**A_2 ST0 B_2 ST0 C_3 ST0 D_3 ST0**

Table 3.10 Simulation based experiment

Simulation reveals that both slow-to-turn-on and slow-to-turn-off faults are observed when resistive opens are assumed on six of the transistors. They are perfectly diagnosed by the proposed method with multiple transitional delays fault reported for each net. The other four transistors do show the slow-to-turn-off effect, because the output is determined by the strong conducting paths from the n-transistor pull-down network. However, their resistive opens are also successfully diagnosed as single ST0 faults as shown in Table 3.10.

3.5. Conclusion

This chapter introduces a novel transformation method developed to address the intra-gate open fault diagnosis. Given the void of intra-gate open fault diagnosis tools and the widely availability of gate-level fault diagnosis tools, a method of transforming open faults from transistor level schematic to a logic gate level description has been proposed. By this method, a stuck-open fault in the transistor level can be represented by a stuck-at fault at the logic gate level. A commercial stuck-at diagnosis tool can be brought in to pick out the stuck-open fault directly without the need to build up excitation tables and rewrite the diagnosis tool. The method is also adaptable to diagnosis of opens affected by timing skew and to the diagnosis of multiple stuck-open faults. Experiments have been performed on the wafer test data of 25 wafers of a Philips 0.18μm design. Seven single stuck-open faults and one multiple stuck-open fault are completely diagnosed. Another two single stuck-open faults are also diagnosed based on the assumption of timing skew effects. An extended version transformation is able to

handle more complicated situation of consecutive high-impedance vectors. One real die is diagnosed of having a stuck-open fault affected by this complication. This diagnosis is successfully confirmed by physical failure analysis. Another type of intra-gate open is resistive open, which causes extra delays in transistor switching status. A slightly modified version transformation is able to represent the transistor slow-to-turn-off and slow-to-turn-on faults with transitional delay faults in the gate level, and thus to be diagnosable in the gate level transitional delay fault diagnosis tools.

To summarize this chapter, a comprehensive transformation package is made available for intra-gate open faults to be diagnosed by gate level diagnosis tools. Meanwhile, because only those cells identified by the initial round stuck-at fault diagnosis are transformed and diagnosed again, these methods are independent of the size of the circuit and take little extra running time. The next chapter will continue on the same general idea of migrating intra-gate fault diagnosis to gate level, but will examine bridging faults instead.

Chapter 4. Intra-gate Bridging Fault Diagnosis

4.1. Introduction

While diagnosis of inter-gate level bridging faults has been extensively studied, as shown in Section 2.3.3.4, little has been mentioned in the IC testing arena about the intra-gate bridging fault diagnosis. There are no available intra-gate bridging fault diagnosis tools in contrast to the fact that inter-gate level bridging fault diagnosis tools like POIROT (Intel) and FALOC (Philips) are well developed [Venkataraman01] [Hora02]. However, the reality is that complex gates are often used in modern CMOS designs and the chances of a bridging fault occuring as an intra-gate fault are high. The inter-gate level diagnosis tools are unable to handle this type of fault as the bridged intra-gate nodes are not represented in the netlist. Following the same thinking of the previous chapter, this chapter exploits the availability of inter-gate level bridging fault diagnosis tools. By migrating the intra-gate bridging fault to the inter-gate level through transformation, it enables a commercial bridging fault diagnosis tool to diagnose the bridging faults inside gates.

This chapter is organized as follows. Section 4.2 shows the method to shortlist intra-gate bridging suspects. Section 4.3 explains a transformation method to diagnose

the intra-gate bridging faults. Section 4.4 gives the overall diagnosis flow, followed by Section 4.5 where experimental results are given. Section 4.6 is the conclusion.

4.2. Method to shortlist possible gates with intra-gate bridging faults

Early on in section 2.3.3.4 a fault model called the net diagnosis model was introduced for the gate level open fault diagnosis. The basic reason for using this model is that it is hard to predict when the open net behaves like a stuck-at-0 fault and when it behaves like stuck-at-1. In order to differentiate the hard open net from the others by the highest Matching result, the net diagnosis model combines both the stuck-at-0 and stuck-at-1 signature of this particular net. Similarities can be drawn between the gate level hard open faults and intra-gate faults. Imagine that a fault occurs inside the gate (not

#Vector	ABC	Fault-free Z	Z Under B(bridge)Net74
#1	000	0	1
#2	001	0	1
#3	010	0	0
#4	011	1	0
#5	100	0	1
#6	101	1	1
#7	110	1	0
#8	111	1	0

Table 4.1 Vectors of sample gate

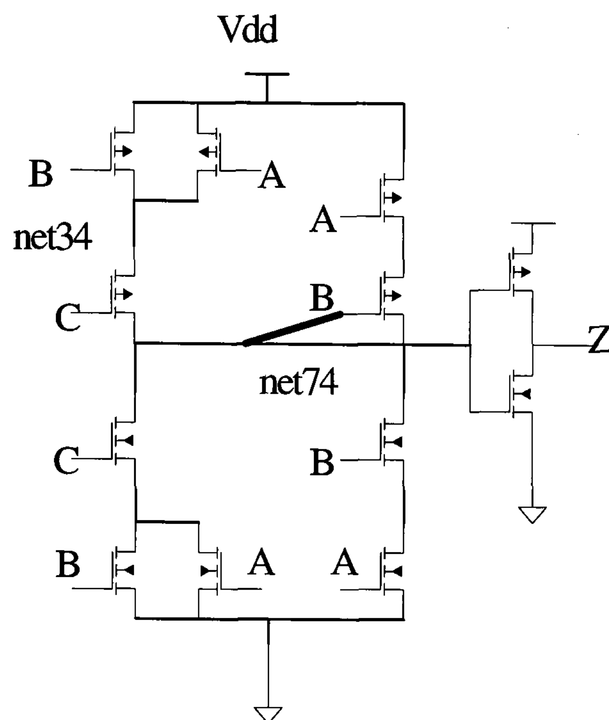


Figure 4.1 Sample gate

necessarily an intra-gate bridging fault). Without sufficient knowledge of the mechanism of the fault, one can not predict that the fault effect on the gate output would be stuck-at-0 or stuck-at-1. To some extent, the output of a gate with an internal fault can be seen as a hard open net, for which both stuck-at-0 and stuck-at-1 faults are possible but in an unpredictable way at the gate level. Therefore the net diagnosis model also applies for the intra-gate fault diagnosis.

Table 4.1 and Figure 4.1 show what happens when a sample gate with internal bridging fault is subjected to a set of input vectors. Supposing that the resistance of the bridging fault is 1 ohm between B and net74, electrical simulation is performed and compared with fault-free Z to generate the observed result, *Obs* = {#1:Z, #2:Z, #4:Z, #5:Z, #7:Z, #8:Z}.

On the other hand, the net diagnosis model combines both the stuck-at-1 and the

stuck-at-0 signatures. The Z stuck-at-1 signature is {#1:Z, #2:Z, #3:Z, #5:Z} and the Z stuck-at-0 signature is {#4:Z, #6:Z, #7:Z, #8:Z}, so the net signature of Z is *Sim* = {#1, #2, #3, #4, #5, #6, #7, #8}. According to the definitions of Matching and Prediction:

$$\text{Matching (M)} = \frac{|Obs \cap Sim|}{|Obs|} \times 100\%$$

$$= \frac{|\{\#1{:}Z,\#2{:}Z,\#4{:}Z,\#5{:}Z,\#7{:}Z,\#8{:}Z\} \cap \{\#1{:}Z,\#2{:}Z,\#3{:}Z,\#4{:}Z,\#5{:}Z,\#6{:}Z,\#7{:}Z,\#8{:}Z\}|}{|\{\#1{:}Z,\#2{:}Z,\#4{:}Z,\#5{:}Z,\#7{:}Z\}|}$$

$$= 100\%$$

$$\text{Prediction (P)} = \frac{|Obs \cap Sim|}{|Sim|} \times 100\%$$

$$= \frac{|\{\#1{:}Z,\#2{:}Z,\#4{:}Z,\#5{:}Z,\#7{:}Z,\#8{:}Z\} \cap \{\#1{:}Z,\#2{:}Z,\#3{:}Z,\#4{:}Z,\#5{:}Z,\#6{:}Z,\#7{:}Z,\#8{:}Z\}|}{|\{\#1{:}Z,\#2{:}Z,\#3{:}Z,\#4{:}Z,\#5{:}Z,\#6{:}Z,\#7{:}Z,\#8{:}Z\}|}$$

$$= 75\%$$

Clearly the output net will always have Matching=100% under the net diagnosis model even though the fault is actually inside the gate. However like the net diagnosis model for hard open faults, a low Prediction is expected since no real defect will cause the output net to be both stuck-at 0 and stuck-at 1. Therefore the gates diagnosed as Matching = 100% and Prediction < 100% under the net diagnosis model are taken as the suspect gates for further analysis to confirm if there is an intra-gate bridging fault.

It may also happen that intra-gate bridging faults will only result in the fault signature either from the output stuck-at-1 or from the output stuck-at-0 appearing in its *Obs*. For instance in Figure 4.1, when net34 is bridged with Vdd (the fault can be also modelled

as net34 stuck-at-1), the faulty effect could only be sometimes net74 stuck-at-1 and, consequently, Z stuck-at-0. So, for these faults too, the faulty gates will be diagnosed under the stuck-at model as Matching = 100%, Prediction < 100%.

Therefore, to cover all the intra-gate bridging faults, those gates with output diagnosed as Matching = 100%, Prediction < 100% under either net diagnosis model or stuck-at model should be taken as our primary intra-gate bridging fault suspects. Then efforts have to be made to raise the Prediction to confirm that there are intra-gate bridging faults rather than something else. The only convincing evidence would be a Prediction also of 100%. To confirm if the gates contain intra-gate bridging faults (as well as diagnosing which bridging faults have occurred), the most direct solution would be to use electrical level simulation to predict the fault signature (*Sim*) of every possible intra-gate bridging pair and to compare them with the *Obs* extracted from the gate output. However, the number of all the possible intra-gate bridging pairs and the numerous vectors to be simulated means that time becomes a critical issue. Also, for every simulation, a certain resistance value is assumed which does not guarantee the same faulty effect as the real defect. On top of these factors, the whole diagnosis software has to be rewritten.

As has been shown in Chapter 3, the capability of traditional gate-level bridging fault diagnosis tools can be extended to the intra-gate domain. To use the gate-level bridging fault model for a transistor-level circuit, the transistor-level description has

to be transformed to the gate-level while having all the potential bridging nets represented. The biggest problem is that whereas some fault simulation tools do allow transistor-level descriptions, the transistor primitives are normally unidirectional and are thus not able to represent the values of the internal nodes of CMOS circuits The next section will explain the details of a suitable intra-gate bridging fault transformation.

4.3. Intra-gate bridging fault transformation

The available gate-level bridging diagnosis tools analyze the values of possible bridging nets under each test vector. If the values between two possible bridging nets are different, then the bridging fault signature is created from certain stuck-at fault signatures, depending on the bridging model used. For example, if a potential bridging between net A and net B is examined, under the Wired-And model, those failing bits from A stuck-at-0 and B stuck-at-0 signature that put A and B to different values are taken to form the Wired-And signature. This has been extensively discussed in Section 2.3.3.3. To fit in with this process, during the course of transformation, two principles have to be followed:

1) **Every net (for instance in Figure 4.2: A, B, C, D, E, net1, net2, net3, net4, net5, Z) in the transistor-level schematics will be represented in the transformed circuits, and their values at every test vector will remain the same as those before the transformation.**

2) If a stuck-at effect propagates to the output, *i.e.* it is not blocked by any off transistors, the corresponding stuck-at fault in the transformed circuit will also propagate.

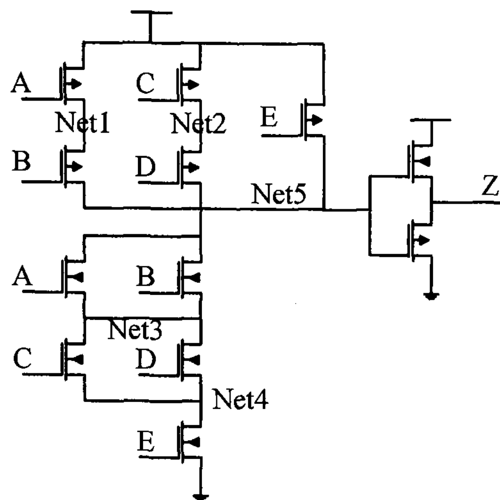


Figure 4.2 Gate to be transformed

4.3.1. The transformation from Principle 1

The transformation method is demonstrated through the example gate in Figure 4.2. There are five internal nets in this particular gate (Net1-5). According to the first principle, all the five internal nets must be represented in a gate-level description where their values are kept. The following rules are set to justify the values of the internal nets.

For n-transistor part:

For the n-transistors part (Net3 and Net4), because the transistors are bi-directional, an internal net's value is either decided by the GND or by the logic 1 coming from p-transistor part. Two rounds of justification are needed, one from the GND and one from the net connecting the n-transistors and p-transistors, in this case Net5. In each

round of justification, the following steps apply.

A. Replace all the n-transistors with the element as shown in Figure 4.3. The purpose is to guarantee that the zero value from the source will be transmitted to the drain when value on the gate is one (transistor turned-on).

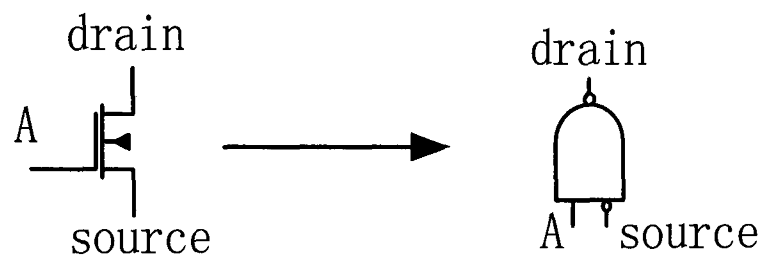


Figure 4.3 Replacement of n-transistor

B. Replace with an AND gate where a parallel connection between transistors is present, as Figure 4.4. The AND gate makes sure if one of the drains is zero, the output will be zero, thus guaranteeing the propagation of zero value from the source.

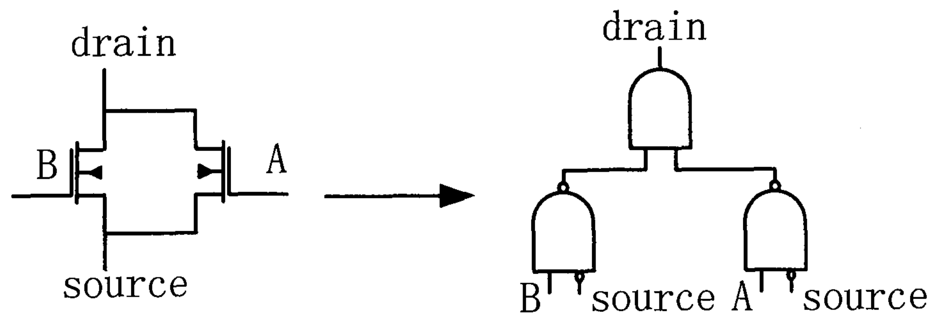


Figure 4.4 Replacement of parallel n-transistors

Figure 4.5 shows the first round justification after the replacement steps. The thing that needs to be mentioned is that Net3_1 and Net4_1 are just indicators of whether Net3 and Net4 are connected with GND, they do not represent the values of Net3 and Net4. The value of internal nets are just 1 or 0, they can also have high-impedance

values in certain circumstances when the nets are not connected to anywhere. Therefore connection indicators like Net3_1 and Net4_1 are needed to tell if the nets are connected to GND or not. The other connection indicators Net3_2 and Net4_2 are also needed to judge if Net3 and Net4 are connected to Net5 or not. The four indicators all together show if the two nets are connected to GND or Net5 or neither of them, in which case high-impedance status appears. Net3_2 and Net4_2 are generated in the same way in the second round justification from the other direction, as shown in Figure 4.5. A zero value shows it is connected, a one value shows it is disconnected.

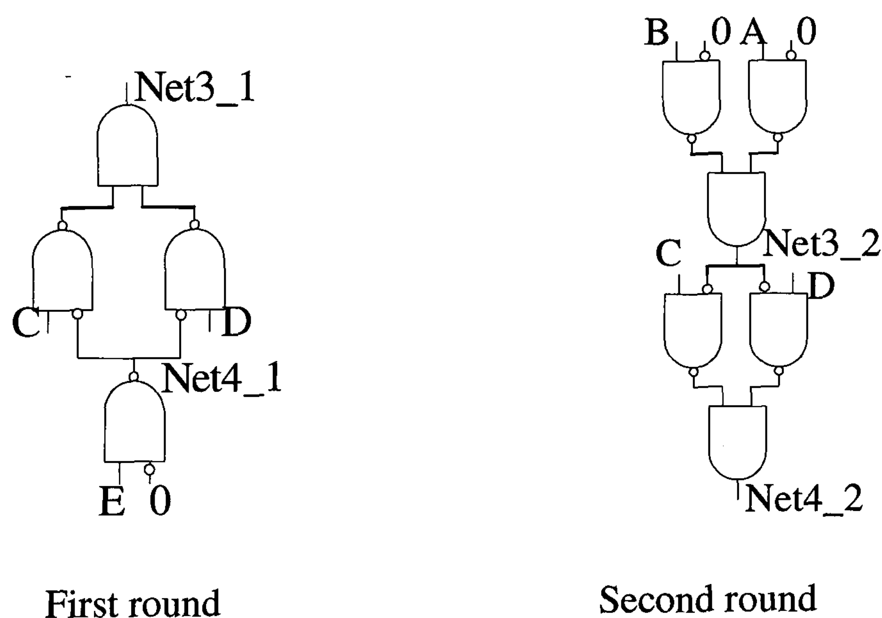
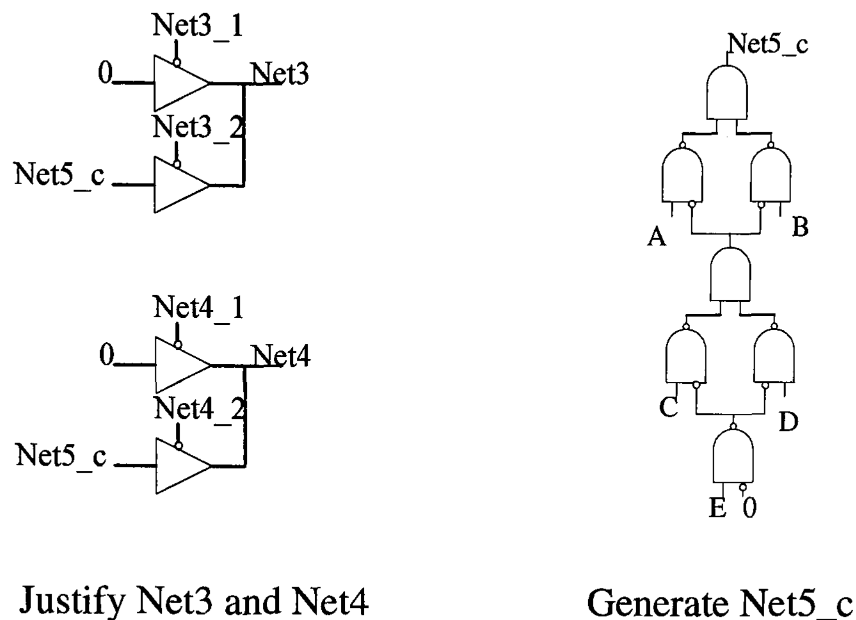


Figure 4.5 First round and second round justifications

Given the four connection indicators, the justification for Net3 and Net4 is done by the four tri-state bus drivers in Figure 4.6, each controlled by a connection indicator and each individually decides if the value of GND or Net5 should be loaded to the Net3 and Net4. Net5_c is the correct value of Net5 and is generated by logic combinations following the same rules as in Section 3.3.2. Net3 and Net4 are now fully justified.



Justify Net3 and Net4 Generate Net5_c

Figure 4.6 Final justifications

For p-transistor part:

For the p-transistor part (Net1 and Net2) of Figure 4.2, two rounds of justifications also have to be performed, one from the Vdd and one from Net5. The rules of transformation are similar to those for the n-transistor part, except when replacing the p-transistor, the element in Figure 4.7 instead of the one in Figure 4.3. The reason for this is that the p-transistor conducts only when the gate node is at 0.

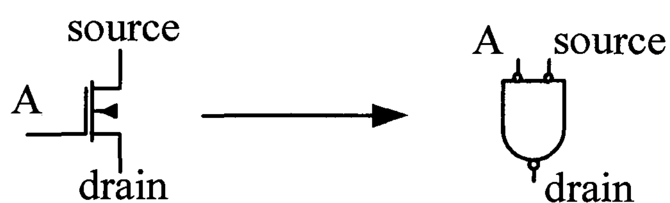


Figure 4.7 Replacement of p-transistor

The rest of the transformation for p-transistor part is just the same as the n-transistor part where N1_1, Net2_1, Net1_2 and Net2_2 are the four connection indicators, as shown in Figure 4.8. Note that there is no need to transform the transistor that directly

connects either Vdd or GND to Net5, like the p-transistor controlled by E, for there is no internal net.

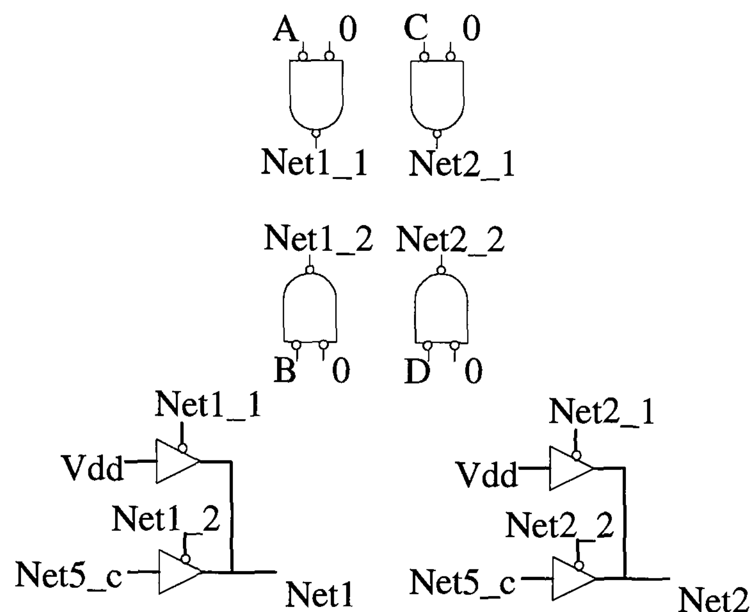


Figure 4.8 The transformation of P transistors

At this stage, all the internal nets have been represented in the transformed circuit where their original values are kept. Principle 1 is satisfied.

4.3.2. The transformation from Principle 2

Principle 2 requires that the internal nets are connected in a way that all the stuck-at signatures are propagated when they are not completely blocked by the off transistors, *i.e.*, they are not completely disconnected from the Net5. Since the connection indicators of every internal net (Net1_2, Net2_2, Net3_2, Net4_2) to Net5 have already been derived, the only thing to do is to add tri-state bus drivers controlled by the connection indicators, whose values decide if a stuck-at signature would be propagated to Net5 and subsequently to the output Z. Figure 4.9 is the whole view after the final transformation. Transistors like the p-transistor controlled by E in

Figure 4.2, which directly connect Vdd or GND to the p-n junction net, have to be replaced by tri-state bus drivers to help Net1-4 to justify the value of Net5.

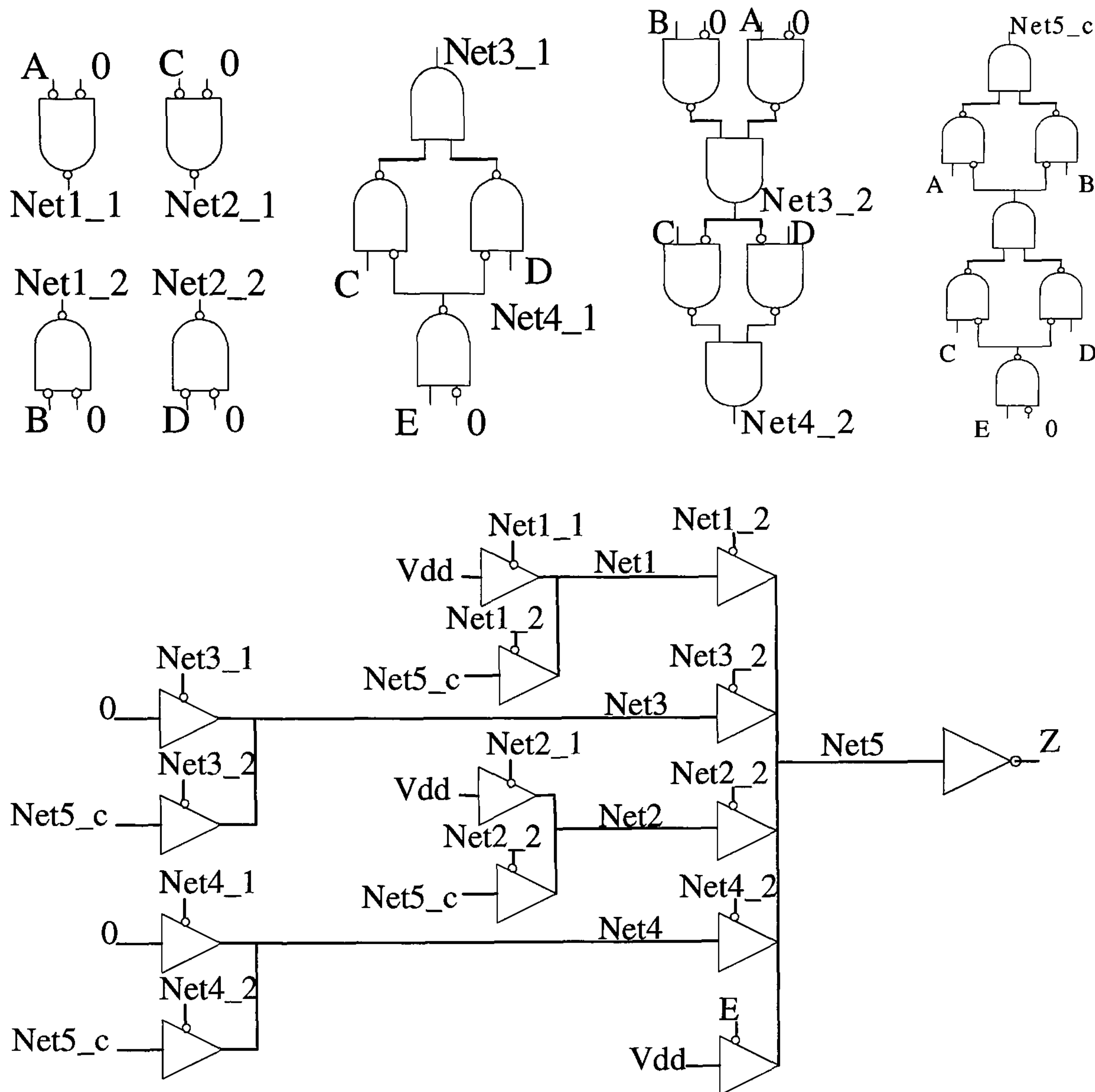


Figure 4.9 Final transformation

With this step, both principles are now met.

4.4. Intra-gate diagnosis flow

Having introduced the crucial step of transformation, the intra-gate diagnosis flow can be summarized as the four steps illustrated in Figure 4.10. First, a preliminary stuck-at fault diagnosis and failing net diagnosis is performed on the whole circuit to shortlist the possible faulty gates with intra-gate bridging. Those gates whose outputs are diagnosed as Matching = 100% and Prediction < 100% under either stuck-at model or net diagnosis model are the primary suspects. The second step is to transform only these gates according to the rules set in Section 4.3. (If desired, transformations can be precomputed for every cell in the library.) In the third step, for those original test vectors that propagate the failure signature of the gate's output, input values of this particular gate are extracted and composed into new test vectors for the transformed gate. In the final step, the gate-level bridging diagnosis is applied on this transformed gate only, which takes very little time.

Currently all pairs of two nets are set as possible intra-gate bridging faults when the final gate-level bridging diagnosis are performed on the transformed circuit, although if relevant tools are available, extraction of realistic intra-gate bridging pairs from the gate layout would be useful to enhance the diagnostic resolution.

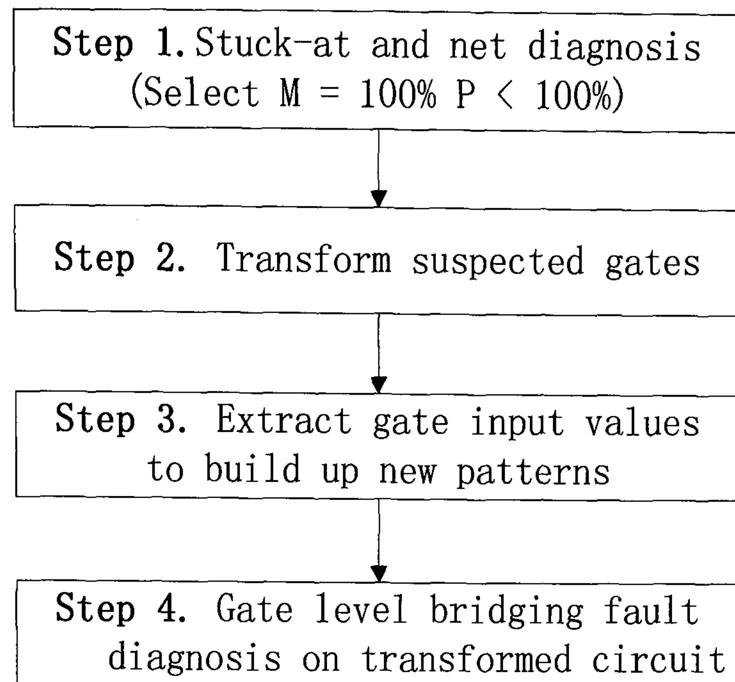


Figure 4.10 Intra-gate diagnosis flow

4.5. Experimental results from wafer testing data

Experiments were performed again with FALOC, which has the capacities of not only stuck-at fault but also net fault and bridging fault diagnosis. The wafer testing data are from three different Philips' designs. Table 4.2 shows seven successfully diagnosed dies. The second and the third columns are the type of faulty gate and the intra-gate bridging fault identified. The fourth column shows the Matching and Prediction results of this particular gate when the first round diagnosis is performed (Step 1). After the transformation, the second round diagnosis is performed and the results are given in column five. The last column shows the number of intra-gate bridging faults that have been diagnosed as Matching = 100%, Prediction = 100%. Sometimes the number is more than one and in this case, layout information has to be referred to judge if one of them is a realistic bridging fault.

Die	Gate Type	Bridged Nets	M (Matching) and P (Prediction) before transformation	M (Matching) and P (Prediction) after transformation	Number of MP = 100% faults
#1	ao32	C and D	Z stuck-at-1 M = 100% P = 63%	C D wired-AND M = 100% P = 100%	1
#2	fa1	B and Net74	CO net model M = 100% P = 61%	B dominates Net74 M = 100% P = 100%	1
#3	an2	B and Net5	Z net model M = 100% P = 42%	B Net5 wired –AND M = 100% P = 100%	1
#4	mx21	SN and Net8	Z net model M = 100% P = 48%	SN dominates Net8 M = 100% P = 100%	2
#5	mx21	SN and Net8	Z net model M = 100% P = 54%	SN dominates Net8 M = 100% P = 100%	2
#6	ao36	Net2 and Z	Z stuck-at-0 M = 100% P = 3%	Net2 dominates Z M = 100% P = 100%	1
#7	mx41x4	S1 and Net74	Z net model M = 100% P = 33%	S1 dominates Net74 M = 100% P = 100%	2

Table 4.2 Successfully diagnosed dies

As shown in Table 4.2, for all the seven faulty dies, Predictions are enhanced to 100% after the gate is transformed. Layout information reveals only one likely site of bridging fault for those multiple diagnosis (Die #4, Die #5 and Die #7). For those singly diagnosed bridging fault (Die #1, Die #2, Die #3, Die #6), a likely layout explanation is also found. (This step would be unnecessary should a layout extraction tool for intra-gate bridging faults be available.) Figure 4.11 illustrates the schematic locations of diagnosed bridging pairs.

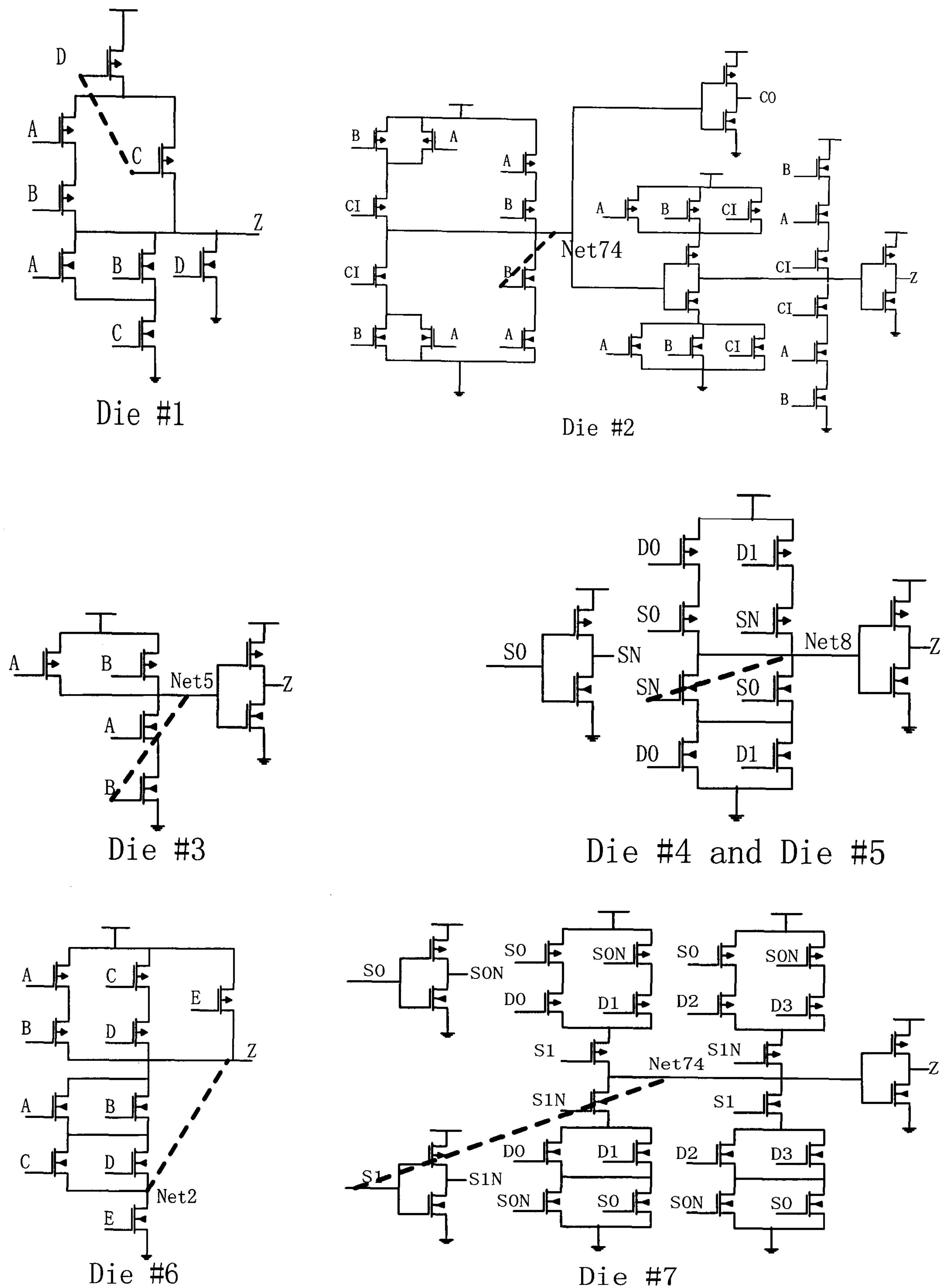


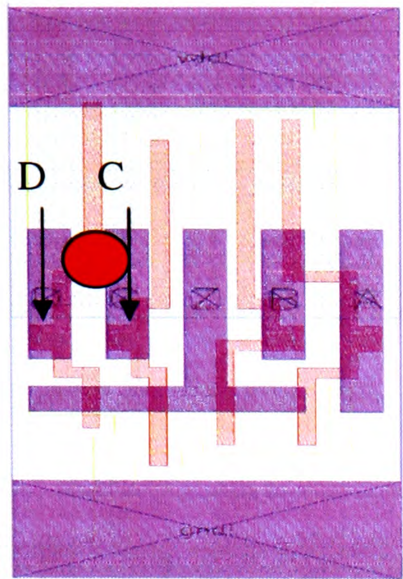
Figure 4.11 Seven intra-gate bridging faults

To further prove our diagnosis, electrical transistor-level simulations are performed for all the seven intra-gate bridging faults. The resistance of the bridging fault is

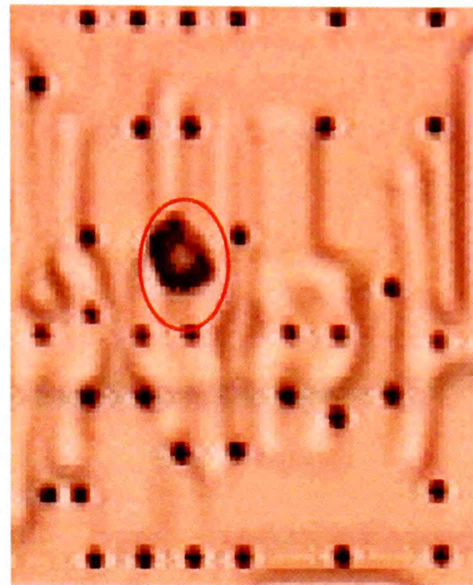
assumed to be very low (1 Ω) and the simulation results confirm the modelled behaviour of all the bridging faults.

It is worth mentioning that technically speaking the fault in Die#1 C(bridge)D is actually a gate level bridging fault, it is not diagnosed by the gate level bridging fault because this bridging pair is not the potential bridging fault pair list. The two nets come out to the different metal layer therefore are not picked by the layout extraction tools as the potential bridging fault on the gate level. The two are close enough to form a bridging fault only when look into the gate.

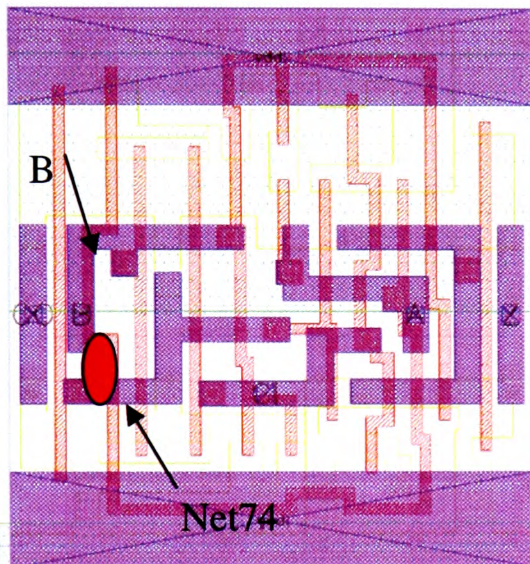
For Die #1, Die #2 and Die #3, inline inspection information has been made available as further proof of the diagnosis. The inline inspection pictures in Figure 4.12 reveal bridging faults that match exactly with the diagnosis results, C and D (Figure 4.12, Die1#.a), B and Net74 (Figure 4.12, Die#2.a), B and Net5 (Figure 4.12, Die#3.a).



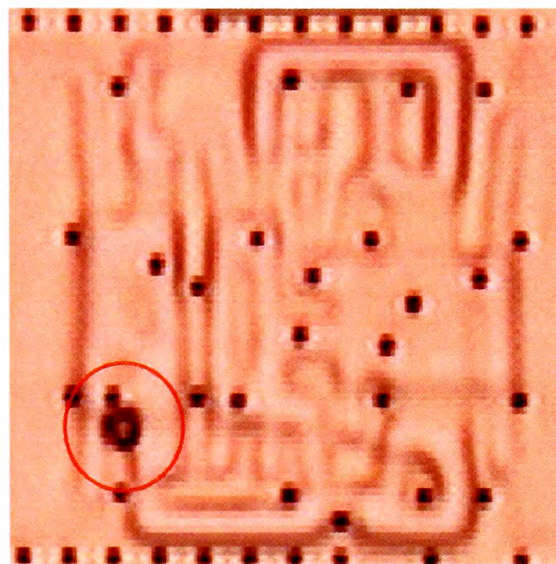
Die#1.a



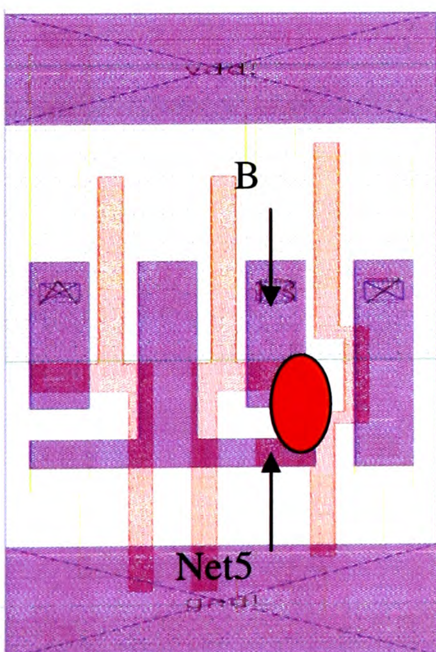
Die#1.b



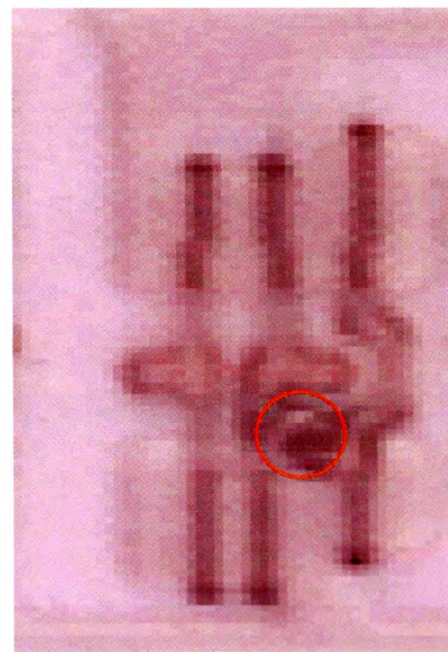
Die#2.a



Die#2. b



Die#3.a



Die#3.b

Figure 4.12 Failure analysis of Die #1, Die #2, Die #3

4.6. Conclusion

This chapter follows a similar development to Chapter 3 and shows how transistor-level circuits can be transformed to the gate level to allow intra-gate fault to be diagnosed with well developed gate level fault diagnosis tools. The new intra-gate bridging fault transformation method costs little extra time in addition to the initial stuck-at and net model diagnosis, because the extra round of bridging fault diagnosis is performed only on the suspect gates. Experiments have been done on the wafer testing data from three of Philips' IC designs. Seven dies are successfully diagnosed as having intra-gate bridging faults. Gate layout information is needed in order to prune out the unrealistic intra-gate bridging faults in three cases and narrow the diagnosis to just one possible candidate. All seven diagnoses are subjected to electrical simulation and their faulty behaviour is confirmed under the low resistance assumption. Out of the seven diagnoses, three dies have inline inspection results, which again prove the correct diagnosis of this method. This intra-gate bridging diagnosis transformation, together with the intra-gate open diagnosis method introduced in Chapter 3, provides an effective and comprehensive solution for intra-gate fault diagnosis.

Chapter 5. Diagnosis Flow for Intra-gate faults

This chapter brings together the results on intra-gate open fault diagnosis in Chapter 3 and intra-gate bridging fault diagnosis in Chapter 4 and shows how they fit together to make a complete diagnosis flow for the various types of intra-gate faults. Figure 5.1 (on the next page) shows how the techniques relate to each other; software tools based on the ideas in this thesis are under commercial development at Philips so Figure 5.1 can also be seen as a guiding map for real intra-gate fault diagnosis practice.

All the devices can not be diagnosed as Matching = 100% and Prediction = 100% at the gate level diagnosis are the target for further analysis. The first round of stuck-at diagnosis is conducted on the whole circuit. When the stuck-at fault diagnosis indicates gates with Matching = 100% and Prediction < 100%, these gates are suspected of having an intra-gate fault. Matching = 100% means that the simulation of an assumed stuck-at fault at the gate-level matches all the failing results that are observed on the tester but Prediction < 100% means that not all the failing results that the simulation predicts are actually observed on the tester. In other words our assumed gate-level fault is in some sense more gross that the actual fault. Following this indication, these gates are examined under different fault assumptions to see if their faulty behaviour fits into

one of the intra-gate fault models. This intra-gate fault diagnosis is performed only on those cells implicated by the gate-level diagnosis and uses only test data already available from the gate-level diagnosis.

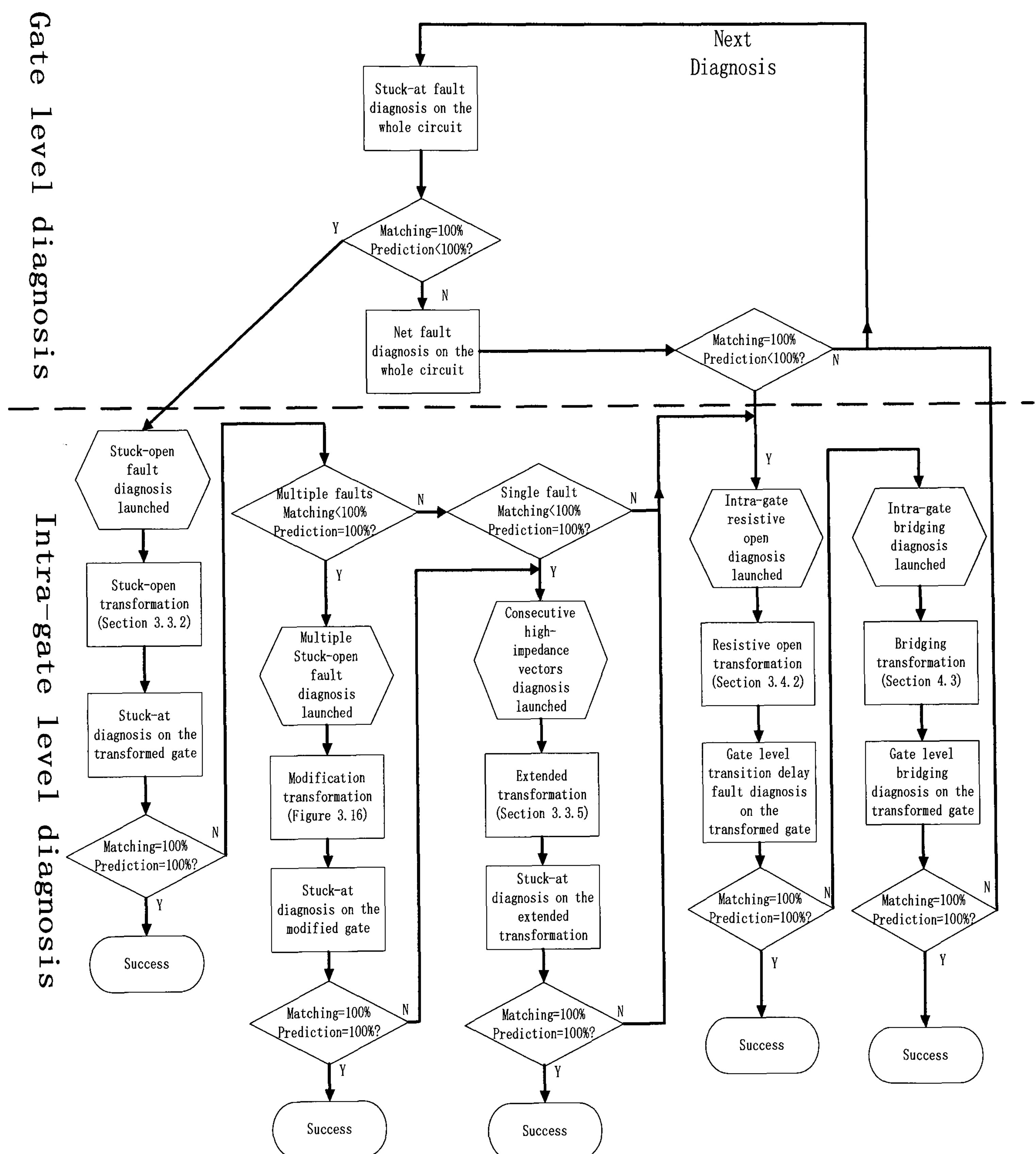


Figure 5.1 Diagnostic map of intra-gate faults

The intra-gate diagnosis is performed by making a series of transformations on the suspect gates in order to express their internal structure as a gate-level circuit. As can be seen from Figure 5.1, stuck-open faults are examined first using the circuit transformation described in Section 3.3.2. If successful diagnosis (Matching = 100% and Prediction = 100%) is not achieved, the gate is then subjected to multiple stuck-open fault and consecutive high-impedance vectors examination, where modified and extended versions of the transformation are applied as in Figure 3.16 (Section 3.3.4.2) and Section 3.3.5. If there is still no successful diagnosis, the assumption of an intra-gate resistive open fault is made and the specific intra-gate resistive open transformation method in Section 3.4.2 is used to help identify the resistive opens. If this still does not lead to a successful diagnosis, the intra-gate bridging fault diagnosis is launched. Suspected gates are subjected to intra-gate bridging fault transformation explained in Section 4.3. Gate level bridging fault diagnosis tool is used subsequently to identify the intra-gate bridging fault. If none of these analyses yield a success, the process gives up and proceeds to the next diagnosis. The reason for this device being faulty could be another issue such as a hard open gate output or a timing related problem (we have managed to identify two examples possibly affected by timing skew problem in Section 3.3.4.2), for which no accurate prediction is available and thus no perfect diagnosis can be achieved.

Since intra-gate resistive opens can cause both slow-to-turn-on and slow-to-turn-off effects on the victim gate, they are not necessarily only indicated by the gate level

stuck-at fault diagnosis, but also the net fault diagnosis model. The same arguments apply for the intra-gate bridging faults. Therefore those gates that do not give Matching=100% and Prediction<100% at the first round gate-level stuck-at fault diagnosis are subjected to the net fault diagnosis. Those gates having Matching=100% and Prediction<100% under net fault diagnosis are also the likely suspects of both intra-gate resistive opens and intra-gate bridging faults. Therefore these gates join the diagnosis flow from the step of intra-gate resistive open diagnosis in Figure 5.1.

To sum up, the first round diagnosis gives the direction of the type of intra-gate fault to look for. Different transformations are developed for each type of intra-gate fault and various special circumstances. The gate level diagnosis tools are applied on these different transformed circuits to examine if the assumptions of the type and location of the faults are correct.

Chapter 6. Conclusions and Future Work

6.1. Conclusions

Failure analysis is an important practice in IC testing to find the root cause of the failure of CMOS circuits so that modifications can be made to the design and/or fabrication to minimize the chance of similar types of failure in the future. In this competitive semiconductor market, where the time of getting product yield up to an acceptable level is crucial to a company's success, an efficient and accurate failure analysis process is very much needed to ensure immediate adjustment of design and manufacturing recipes to increase the yield quickly.

The technique of modern IC design has developed to the point where billions of transistors can be integrated onto one chip. The growing size and the number of layers of the modern IC devices bring a tough question for failure analysis engineers when they solely rely on physical failure analysis measures such as Liquid Crystal Analysis, Photon Emission Microscopy, *etc.* More and more efforts have been focused on the electrical failure analysis process (also known as fault diagnosis), as a vital step in the physical failure analysis process. Fault diagnosis tools analyze the testing data and report the most likely fault candidates to the failure analysis engineers. It is then for

the failure analysis engineers, to decide whether to use the information from fault diagnosis tools and which reported candidate to choose to perform the physical failure analysis. Although only the visual proof from the physical failure analysis stands as the ultimate evidence of the defect, the process is usually destructive and cannot be repeated. Therefore fault diagnosis result must be convincing enough to push forward to the physical failure analysis stage. Two diagnosis measurements are used in this thesis as indicators of the diagnosis quality – Matching and Prediction. The most convincing diagnosis result is when both Matching and Prediction equal 100%. Unfortunately this cannot always be achieved. Most of the existing fault diagnosis tools are developed to address the gate level fault with no consideration of intra-gate faults. However, modern IC design tends to use more complex gates which result in a high chance of intra-gate faults. As can be seen from the experimental results of Chapter 3 and Chapter 5, intra-gate faults are commonly seen.

This thesis provides a major step forward in the diagnosis of intra-gate faults, which have never been systematically addressed before. One important advantage of the methods introduced in this thesis is that they are based on the traditional gate level diagnosis tools. In the current situation when no intra-gate level diagnosis tools are available but gate level diagnosis tools are well developed and widely used, this thesis provides a way to allow any failure analysis engineer who has access to the gate level diagnosis tools to also tackle the intra-gate faults.

The core of the approach is a series of circuit transformations in which intra-gate

faults are replaced by equivalent faults at the gate level. Chapter 3 develops a novel circuit transformation for stuck-open faults, when one of the transistors is permanently turned-off. Chapter 3 also shows that special cases like the timing-skew problem, multiple stuck-open transistors, and the consecutive high-impedance test vectors problem can be handled either with minor adjustments or replicated extensions of the transformation. Successful diagnosis results are given from the real wafer testing data and are corroborated by in-line inspection data. The other major type of open faults is resistive open fault that can cause delay effects. A special transformation is also developed in Chapter 3 to tackle them. It is able to diagnose the slow-to-turn-off and slow-to-turn-on effect at the same time, which could be the simultaneous faulty effects from a resistive open transistor. The other fault category which is common in practice is the bridging fault. Chapter 4 develops a unique transformation which models intra-gate bridging faults at the gate level where existing line-dominant, Wired-OR and Wired-AND models can be used. The method is implemented in gate level bridging diagnosis tools, and yields several successful diagnosis results from the real wafer testing data. Physical failure analysis has been performed on some of the diagnosed dies and this confirms the diagnosis.

With comprehensive coverage of many types of intra-gate fault and the solid experimental results confirmed not only by simulation but also by real wafer testing data and physical failure analysis, the new method for intra-gate fault diagnosis has been proven to be effective and efficient. It also generates the optimal outcome with

minimum cost by using a currently available fault diagnosis platform. These new methods of stuck-open fault diagnosis and intra-gate bridging fault diagnosis are in the process of being integrated into Philips' FALOC system. More positive results are expected to come when diagnosis can then be performed on a massive scale.

6.2. Future work

Although this thesis offers an effective method in dealing with various types of intra-gate faults, there is still a lot of work that needs to be done in the future.

A. Confirmation from the wafer test data and subsequent FA results for the intra-gate resistive open would provide the ultimate proof of the effectiveness of the resistive open transformation. Given a little more time, there will be progress in getting the real intra-gate resistive open diagnosis results like those achieved in stuck-open and intra-gate bridging fault diagnosis.

B. The timing skew problem is still playing a damaging effect in diagnosis. Efforts should be made to find an efficient way to identify the correct transition order of the gate inputs.

C. This transformation method has the potential to be used in ATPG (Automatic Test Pattern Generation) process. A set of specific test patterns can be generated for the transformed gate level representation by the gate level ATPG tools, to ensure high fault coverage for the intra-gate faults.

References

Abd-El-Barr99 M. H. Abd-El-Barr et al, "Transistor Stuck-Open Fault Detection in Multilevel CMOS Circuits", Great Lakes Symposium on VLSI, pp. 388-392, 1999.

Abramovici80 M. Abramovici, et al "Multiple Fault Diagnosis in Combinational Circuits based on an effect-cause analysis", IEEE Tans. On Computing, Vol C-29, pp. 451-460, 1980.

Abramovici90 M. Abramovici, et al "Digital systems testing and testable design". Freeman, 1990.

Acken91 J. M. Acken and S. D. Millman, "Accurate Modelling and Simulation of Bridging Faults", Proceeding of Custom Integrated Circuit Conference, pp. 17.4.1-17.4.4, 1991.

Aitken95 R.C. Aitken and P.C. Maxwell, "Better Models or Better Algorithms? Techniques to Improve Fault Diagnosis", Hewlett-Packard Journal, Vol. 46, No. 1, pp. 110-116, 1995

Allen92 R.W. Allen, M.M. Ervin-Willis and R.E. Tullose. "DORA: CAD Interface to Automatic Diagnostics" 19th Design Automation Conference, pp. 559-563, 1982.

Amerasekera97 E. A. Amerasekera and F. N. Najm, "Failure Mechanisms in Semiconductor Devices", John Wiley & Sons, 1997.

Baker97 K. Baker and J. van Beers, "Shmoo Plotting: The Black Art of IC Testing", IEEE Design & Test of Computers, pp. 90-97, July-September, 1997.

Burgess84 N. Burgess and R. I. Damper, "The Inadequacy of he Stuck-at Fault Model for Testing MOS LSI Circuits: A Review of MOS Failure Mechanisms and Some Implications for Computer Aided Design and Test of MOS LSI Circuits", Software and Microsystem, Vol. 3, pp. 30-36, April 1984.

Camurati90 P. Camurati et al., "Diagnosis Oriented Test Pattern Generation", European Design Automation Conference, pp. 470-474, 1990.

Chakravarty99 S. Chakravarty and V. Gopal, "Techniques to Encode and Compress Fault Dictionaries", IEEE VLSI Test Symposium, pp. 195-200, 1999.

Champac94 V. H. Champac et al, "Electrical Model of the Floating Gate Defect in CMOS ICs: Implication on IDDQ Testing", IEEE Transactions on Computer-Aided-Designs, Vol.13, no.3, pp.359-369, 1994.

Chandramouli85 R. Chandramouli and H. Sucar, "Defect Analysis and Fault Modelling in MOS Technology", IEEE International Test Conference, pp. 313-321, 1985.

Chess94 B. Chess et al "On Evaluating Competing Bridge Fault Models for CMOS ICs", VLSI Test Symposium, pp.446-451, 1994.

Chess95 B. Chess, et al, "Diagnosis of realistic bridging faults with single stuck-at information" International Conference on Computer-Aided Design, pp. 185—192, 1995.

Colvin98 J. Colvin "Review Paper ESD Failure Analysis Methodology" Microelectronics Reliability, Vol.38, pp. 1705-1714, 1998.

De95 K. De and A. Gunda, "Failure analysis for Full-Scan Circuits", IEEE International Test Conference, pp. 636-645, 1995.

Elziq81 Y. M. Elziq, "Automatic test generation for stuck-open faults in CMOS VLSI", Proceedings of the 18th conference on Design automation, pp.347-354,1981.

Fantini85 F. Fantini and C. Morandi, "Failure Modes and Mechanisms for VLSI IC's – A Review", IEE Proceedings, Vol. 132, Pt G, No. 8, pp 74-81, June 1985.

Ferguson88 F. J. Ferguson and J. P. Shen, "A CMOS Fault Extractor for Inductive Fault Analysis", IEEE Trans. on Computer-Aided Design, Nov. 1988, pp. 1181-1194.

Ferrer90 C. Ferrer et al, "A new switch-level test pattern generation algorithm based on single path over a graph representation", European Design Automation conference, pp. 402-406, 1990.

Fujiwara83 H. Fujiwara and T. Shimono, "On the Acceleration of Test Generation Algorithms", IEEE Transactions on Computers, Vol. C-32, No. 12, pp. 1137-1144, December 1983.

Fujiwara90 H. Fujiwara "Computational Complexity of Controllability /Observability Problems for Combinational Circuits" Trans. On Computers, 1990, pp.762-767.

Geol81 P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits", IEEE Transactions on Computers, Vol. C-30, No. 3, pp. 215-222, March 1981.

Greenstein92 G. Greenstein and J. Patel, "EPROOFS: A CMOS Bridging Fault Simulator", Proc. Int'l Conf. On Computer-Aided Design, pp. 268-271, 1992.

Hartanto97 I. Hartanto, et al., "Diagnostic Test Pattern Generation for Sequential Circuits", IEEE VLSI Test Symposium, pp. 196-202, 1997

Hawkins90 C. F. Hawkins et al., "The Use of Light Emission in Failure Analysis of CMOS ICs", International Symposium for Testing and Failure Analysis, pp. 55-67, 1990.

Hawkins94 C. F. Hawkins et al, "Defect Classes- An Overdue Paradigm for CMOS IC Testing", International Test Conference, pp. 413-425, 1994

Hiatt81 J. Hiatt, "A method of Detecting Hot Spots on Semiconductors Using Liquid Crystals", International Reliability Physics Symposium, pp. 130-135, 1981.

Hora02 C. Hora, "On Diagnosing Faults in Digital Circuits", PhD Thesis, Technische Universiteit Eindhoven, 2002

Hsu98 Y. C. Hsu and S. K. Gupta, "A New Path-Oriented Effect-Cause Methodology to Diagnose Delay Failures", IEEE International Test Conference., pp. 758-767, 1998.

Hurst98a S. L. Hurst, "VLSI Testing, digital and mixed analogue/digital techniques", The Institution of Electrical Engineers, 1998

Hurst98b S. L. Hurst, " VLSI Custom Microelectronics, Digital, Analog, and Mixed-Signal", Marcel Dekker, 1998.

Ishizuka90 H. Ishizuka et al, "Advanced Method of Failure Analysis Using Photo Spectrum of Emission Microscopy". International Symposium for Testing and Failure Analysis, pp. 13-19, 1990.

Jee93 A. Jee and F. J. Ferguson, "Carafe: An Inductive Fault Analysis Tool for CMOS VLSI. Circuits," Proceedings of VLSI Test Symposium, pp. 92-98, 1993

Jha03 N. Jha and S. Gupta, "Testing of Digital System", Cambridge University Press, 2003.

Konuk97 H. Konuk and F. J .Ferguson, "Oscillation and Sequential Behavior Caused by Interconnect Opens in Digital CMOS Circuits". International Test Conference, pp.597-606, 1997.

Krishnamachary02 A. Krishnamachary et al, "Test Generation for Resistive Opens in CMOS", Proceedings of the 12th ACM Great Lakes Symposium on VLSI, pp. 65-70, 2002

Kunda93 R. P. Kunda, "Fault Location in Full-Scan Designs." Int'l Symposium for Test and Failure Analysis, pp.121-126, 1993.

Lavo96 D. B. Lavo and T. larrabee and B. Chess, "Beyond the Byzantine Generals: Unexpected Behavior and Bridging-Fault Diagnosis", International Test Conference, pp. 61-619, 1996

Lavo97 D. B. Lavo et al, Bridging Fault Diagnosis in the Absence of Physical Information". IEEE Int'l Test Conf , pp. 887-893, 1997.

Lavo98 D.B.Lavo, et al "Diagnosing Realistic Bridging Faults with Single Stuck-at Information", IEEE Transaction On CAD and Design of Integrated Circuits and Systems, vol.17, no.3, pp.255-268, 1998.

Lavo01 D. B. Lavo and T. Larrabee, "Making Cause-Effect Cost Effective: Low-Resolution Fault Dictionaries", International Test Conference, pp. 278-286, 2001.

Lee89 H.K. Lee et al, "Test generation of stuck-open faults using stuck-at test sets in CMOS combinational circuits", Proceedings of the 26th ACM/IEEE conference on Design automation, pp. 345 – 350, 1989.

Lee92 S. Y. Lee et al "An Algorithms to reduce test application time in full scan designs" Int'l Conf. Computer Aided Design, pp.17-20, 1992.

Li00 J. Li and E. J. McClusky, "Testing for Tunneling Opens", International Test Conference, pp.85-94, 2000

Li01 J. Li et al, "Testing for Resistive Opens and Stuck Opens", International Test Conference, pp.1049-1058, 2001.

Li02 J. Li et al, "Diagnosis for Sequence Dependent Chips", IEEE VLSI Test Symposium, pp.187-192, 2002

Li03 Z. Li, et al, "A Circuit Level Fault Model for Resistive Opens and Bridges", ACM Transactions on Design Automation of Electronic Systems, Volume 8 , Issue 4, pp. 546 – 559, 2003.

Madhukar85 K. Madhukar et al, "Transistor level test generation for MOS circuits", IEEE Design Automation Conference, pp. 825-828, 1985.

Mei74 K. Y. Mei, "Bridging and Stuck-At Faults", IEEE Trans. On Computers. Vol. C-23(7), 1974, pp.720-770

McCluskey03 McCluskey, et al "Defects vs. Faults: Some Data from the ELF35 and Murphy Chips," submitted to the Int'l. Test Conf., Sep. 30-Oct 2, 2003.

Menon93 S.M. Menon et al, "Testable Design of BiCMOS Circuits for Stuck-Open Fault Detection using Single Patterns", VLSI Test Symposium, pp. 296-302, April 1993.

Millman89 S. D. Millman and E. J. McCluskey "Detecting Stuck-Open Faults with Stuck-At Test Sets", IEEE Custom Integrated Circuits Conference, pp. 22.3.1-22.3.4, 1989.

Millman90 S.D. Millman, et al "Diagnosis of Bridging Faults with Stuck-at Fault Dictionaries", International Test Conference, pp.860-870, 1990.

Mourad00 S. Mourad and Y. Zorian, "Principles of Testing Electronic Systems", Wiley Inter-Science, 2000.

Plummer00 J. D. Plummer et al, "Silicon VLSI Technology, Fundamentals, Practice and Modeling", Prentice Hall, 2000.

Pomeranz92 I. Pomeranz and S. M. Reddy, "On the Generation of Small Dictionaries for Fault Location", IEEE International Conference on Computer-Aided Design, pp. 272-279, 1992.

Rafiq98 S. Rafiq, et al., "Testing for Floating Gates Defects in CMOS Circuits", Proceeding of Asian Test Symposium, pp.228-236, 1998.

Rajski87 J. Rajski and H. Cox, "A method of test generation and fault diagnosis in very large combinational circuits." IEEE International Test Conference, pp. 932-943, 1987.

Ratford86 V. Ratford and P. Keating, "Integrating guided probe and fault dictionary: an enhanced diagnostic approach" International Test Conference, pp. 304-311, 1986.

Renovell92 M. Renovell and G. Cambon, "Electrical Analysis and Modeling of Floating Gate Faults in MOS Circuits", IEEE Transaction on Computer-Aided-Designs, pp.1450-1458,1992

Richards83 B. P. Richards and P. K. Footner, "Failure Analysis in Semiconductor Devices – Rationale, Methodology and Practice", The GEC Journal of Research, Vol 1, No 2, pp. 74-91, 1983

Richards92 B. P. Richards and P. K. Footner, "The Role of Microscopy in Semiconductor Failure analysis", Oxford Science Publications, Oxford University Press, Royal Microscopical Society, 1992.

Renovell92 M. Renovell and G. Cambon, "Electrical Analysis and Modeling of Floating Gate Faults in MOS Circuits", IEEE Trans. On Computer-Aided-Designs, pp. 1450-1458. 1992

Roth66 J. P. Roth, "Diagnosis of Automata Failures: A Calculus and a Method", IBM Journal of Research and Development, Vol. 10, pp. 278-281, 1966.

Sachdev97 M. Sachdev, "Open Defects in CMOS RAM Address Decoders," IEEE Design and Test of Computers, vol. 14, no. 2, pp. 26-33, 1997.

Sar-Desai99 V. R. Sar-Dessai and D. M. H. Walker, "Resistive Bridge Fault Modelling, simulation and Test Generation", International. Test Conference, pp. 596-605. 1999.

Segura04 J. Segura and C. F. Hawkins, "How It Works, How It Fails", Wiley-Interscience, 2004.

Sheppard96 J. W. Sheppard and W. R. Simpson, "Improving the Accuracy of Diagnostics Provided by Fault Dictionaries", IEEE VLSI Test Symposium, pp. 180-185, 1996.

Sivaraman98 M. Sivaraman et al, "A Unified Approach for Timing Verification and Delay Fault Testing" Kluwer Academic publishers, 1998.

Stroud00 C. E. Stroud et al, "Bridging Fault Extraction from Physical Design Data for Manufacturing Test Development," International. Test Conference, pp. 760, 2000.

Tahoori02 M. B. Tahoori et al, "Improving Detectability of Resistive Open Defects in FPGA", MAPLD International Conference, September 2002

Tangyunyong96 P. Tangyunyong et al. "Localizing Heat Generating Defects Using Fluorescent Microthermal Imaging", Int'l. Symposium for Testing and Failure Analysis, pp. 55-62, 1996.

Taur98 Y. Taur and T. H. Ning, "Fundamentals of Modern VLSI Devices", Cambridge University Press, 1998.

Timoc83 Timoc et al, "Logical Models for Physical Failures", Int'l Test Conf. pp. 546-553. 1983.

Turino97 J. Turino, "Test Economics in the 21st Century", IEEE Design and Test of Computers, Vol 14, No.3, 1997.

Uyemura02 J. P. Uyemura, "Introduction to VLSI Circuits and Systems", John Wiley & Sons, 2002.

Vallett97 D. P. Vallett, "IC Failure Analysis: The Importance of Test and Diagnostics.", IEEE Design and Test of Computers, Vol 14, No.3, 1997.

Veendrick98 H. Veendrick, "Deep-Submicron CMOS ICs", Kluwer BedrijfsInformatie, 1998.

Venkataraman97 S. Venkaratarnan and W. K. Fuchs, "A Deductive Technique for Diagnosing of Briding Faults", IEEE International Test on Computer Aided Design, pp. 467-475, 1997.

Venkataraman01 S. Venkataraman et al, "Poirot: Applications of a logic fault diagnosis tool", IEEE Design and Test of Computers, Vol.18,no.1, pp. 19-30, 2001.

Waicukauski87 J. A. Waicukauski et al, "Transition Fault Simulation" IEEE Design and Test of Computers, Vol.4, No. 2, pp.32-38, 1987

Waicukauski89 J. A. Waicukauski and E. Lindbloom, "Failure Diagnosis of Structured VLSI", IEEE Design & Test of Computers, Vol. 6, no. 4, pp. 49-60, 1989.

Wagner99 L. C. Wagner, "Failure Analysis of Integrated Circuits, Tools and Techniques", Kluwer Academic Publishers, 1999.

Woodhall87 B. W. Woodhall et al, "Empirical Results on Undetected CMOS Stuck-Open Failures", International Test Conference, pp. 166-170, 1987.

Wilkins86 B. R. Wilkins, "Testing Digital Circuits, An introduction", Van Nostrand Reinhold, 1986

Zantye04 P. B. Zantye et al, "Chemical mechanical planarization for microelectronics applications" Materials Science and Engineering R 45 pp. 89–220, 2004.

Appendix A Failure Analysis Techniques

IC defects are unfortunately frequently occurring in many possible ways during the IC manufacturing process. Advanced fault modelling and test pattern generation and DFT technologies allow us to detect most of these IC defects, but to eliminate the defects and increase the reliability of the IC products in the future requires improvements to the design and the manufacturing process. The knowledge has to come from a deep understanding of the failure mode, failure mechanism and the root cause of failures. Failure Analysis (FA) refers to the process of locating the defects and characterizing their properties. The information from failure analysis is used to understand how these defects are generated and how to eliminate the root causes in support of process improvement impacting product yield.

Typical failure analysis flow

A typical failure analysis flow is shown in Figure 1[Hora02] [Wagner99]. Beginning with visual inspection, FA engineers have to look for bent leads, package cracks, signs of heat damage or lead corrosion to prune out the bulk defects before the next step – electrical characterization. Electrical testing plays a very significant role in failure analysis which provides a general understanding of how the device is failing

electrically.

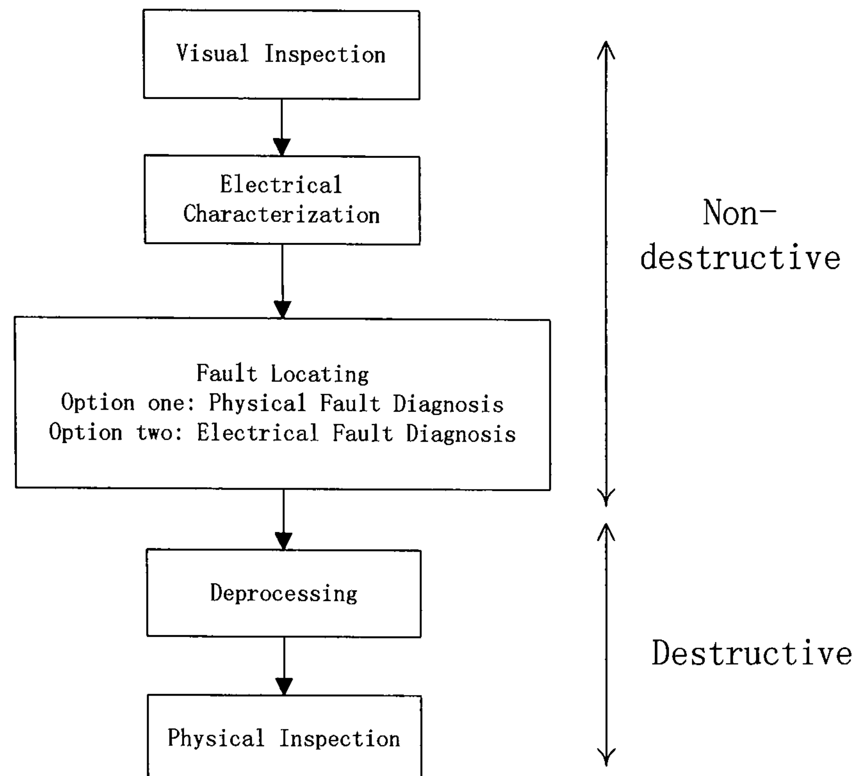


Figure 1 Typical flow of failure analysis

The faulty devices are then subjected to the fault locating process for narrowing the focus of diagnostic investigations. Modern IC devices may have millions of transistors and interconnections. Therefore it is essential to narrow down to a few suspects from the millions in order to be able to perform physical examination. Two main types of fault locating techniques can be used – physical fault diagnosis and electrical fault diagnosis. Once the suspected locations of the defects are found, the device is deprocessed to expose the suspected faulty layers and locations to be subjected to the following physical inspections. From the deprocessing step onward, the failure analysis becomes a destructive process, which includes etching and polishing steps to be able to get through the layers on top of the suspected locations. It is important that the electrical tests and non-destructive analysis have been completed before destructive techniques are employed. The object of destructive testing is to

verify the hypothesis of the defect locations and mechanisms as obtained from the fault locating step [Wagner99] [Amerasekera97] [Hora02]. In the rest of the appendix, a brief review is given to each of the failure analysis steps starting from the electrical characterization.

Electrical Characterization

Electrical characterization is the real starting point of the failure analysis process. It plays a very important role in the success of the whole failure analysis, by providing an initial ability to narrow down the scope of the analysis. Electrical failures can be categorized into three areas: continuity failures, parametric failures and functional failures. Continuity failures are caused by the input and output connections with the external devices. It is relatively easy to characterize and only require a simple measurement of opens and shorts on the external pins. Parametric failures require more complex measurements to characterize. If one or more of a device's parametric specifications are not met, like power supply current, bias current, transistor threshold voltages, frequency response, *etc.,* a parametric failure has occurred. Functional failure characterization is performed by giving a known stimulus and measuring the responding output. If the measured output is the same as the expected results, the device is said to be functional. Otherwise, a functional failure occurs. It is not always possible to draw a distinct line between the parametric test and functional tests. For example, power supply current parameters test, normally referred as I_{ddq} test, may require a very specific functional state in order to measure the leakage current. If the

failure effect is achieved regardless of the functional state of the circuit, this failure is more like a parametric failure. If the failure effect is quite functional state dependent, its characterization is more like functional failures [Wagner99].

For simple devices the analysis can be done with bench test electronics including power supplies, function generators, pattern generators, logic analyzers. But for complex devices, ATE (Automatic Test Equipement) is the main tool used.

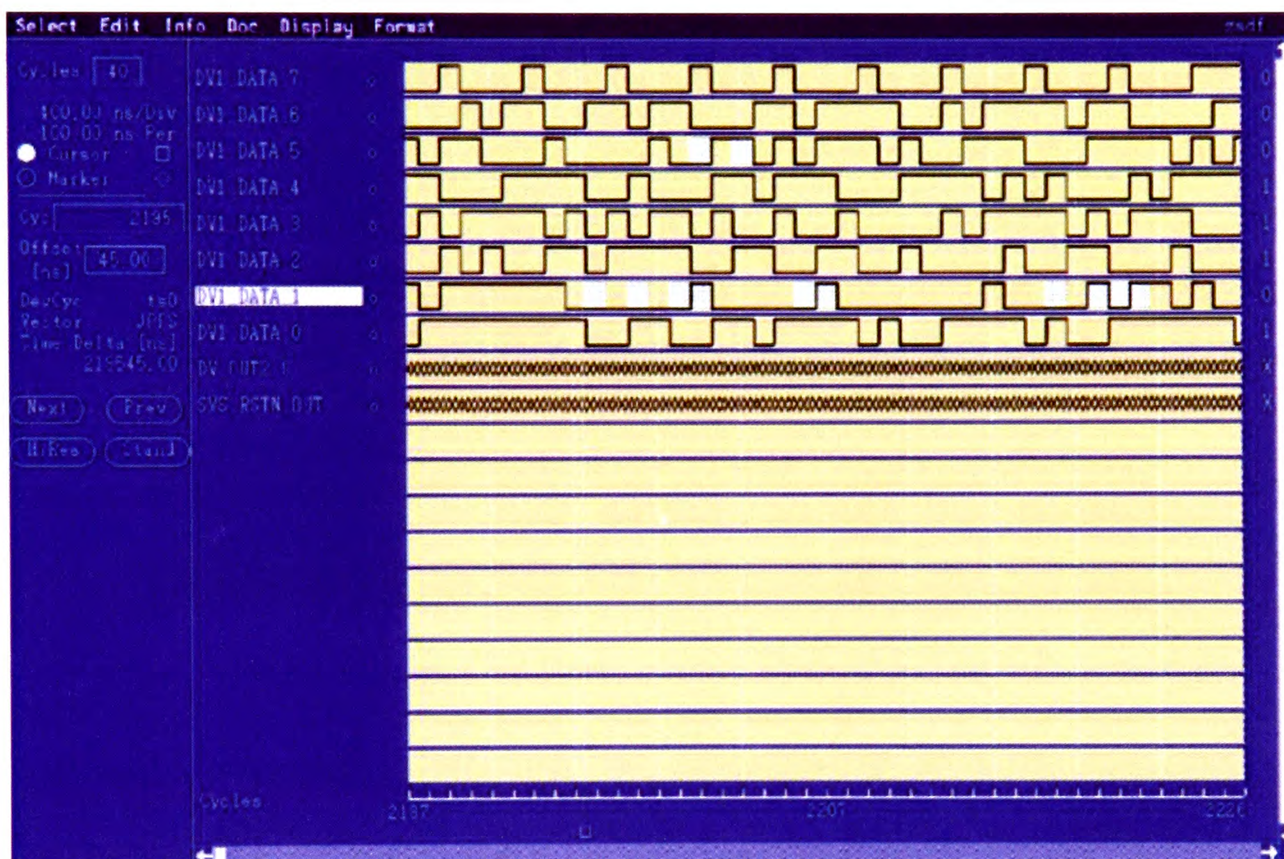


Figure 2 Graphic output waveform tool [Philips Research]

Today's ATE incorporate advanced debug features that can be graphically shown, such as wave, vector, pattern, shmoo plots. The wave tool is a like a digital oscilloscope which displays both the input and output waveform of the device, Figure 2.

A Shmoo plot is a useful tool to characterize the device's performance in relation to

the changes of different parameters such as temperature, clock frequency and power supply voltage [Baker97], Figure 3. The knowledge gained from Shmoo plotting can be used to optimize the process, design and final test program.

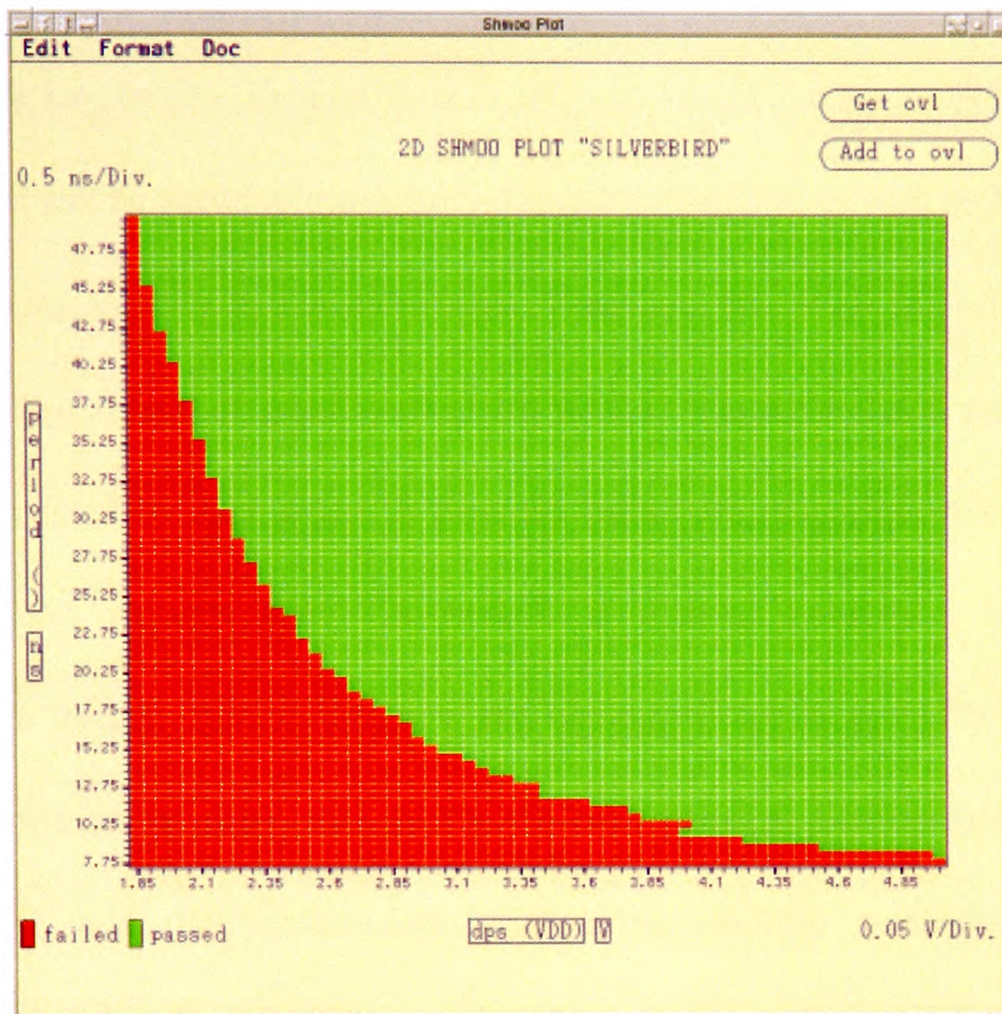


Figure 3 Shmoo plot Voltage versus Period [Philips Research]

Failure analysis normally begins with an examination of the datalog measured from the production test program. Most production test programs stop at the time when the first indication of failure happens. Therefore this feature must be changed to generate a full size datalog of all of the production tests. By examining the full datalog, the FA engineer is able to tell if one failure mode is dominant. Interpretation of this full datalog has to be as accurate as possible and the quality of the interpretation determines the complexity in subsequent failure analysis.

Fault locating

Fault locating refers to the act of finding out the real defect location. It is essential to narrow down the suspected defect locations into a few before the destructive deprocessing step begins, because there is only one chance during the destructive step. The locating can be performed precisely down a logic block, a logic node or, in some cases, pointing directly to the defect site. There are two ways in performing fault location: physical fault diagnosis and electrical fault diagnosis. These two techniques are supplemented by each other to gain the maximum diagnostic results. Physical fault diagnosis requires heavy investment in relevant equipment and qualified personnel in testing, chip architecture and equipment operating. Although physical fault diagnosis can in many cases lead to the exact physical location of the defects, it is expensive and time consuming because the package normally has to be decapsulated and circuit features may then be examined using microscopes [Tangyunyong96]. Alternatively, electrical fault diagnosis can be performed by applying tests to the inputs of the circuit, capturing and analyzing the response at the outputs. The performance of electrical fault diagnosis largely depends on the accuracy of the fault model and the simulation. The next two sections give a brief look at the two locating techniques respectively.

Physical Fault Diagnosis

Physical examinations can start with a package analysis, utilizing radiographic techniques like X-rays, Scanning Acoustic Microscopy (SAM) and thermal imaging.

An X-ray scan provides a non-destructive method for obtaining a graphical representation of the chip's internal structure. It helps locate die attach voids, lead frame shorts, damaged or missing bond wires [Wagner99] [Amerasekera97]. SAM is based on the focusing of an acoustic pulse at an interface within the package to examine the package cracking, voiding. After the package analysis, the chip has to be decapsulated to allow the necessary exposure of the die surface to perform the next step of physical fault diagnosis.

Most of the physical fault diagnosis measures take advantages of the secondary effect of the failure, like thermal or optical effects [Hiatt81] [Ishizuka90] [Hawkins90]. Many defects result in a leakage power supply which would generate hot spots during the chip operation. Imaging techniques such as liquid crystal analysis, Fluorescent Microthermal Imaging (FMI) use this effect to locate the defects that cause extra heat emission. In liquid crystal analysis, the chip is placed on a temperature controlled chuck whose temperature depends on the phase transition temperature of the type of liquid crystal used and is typically in the range of 50°C to 70°C. After the liquid crystal is applied to the exposed die surface, the chip is powered up in to operation. The liquid crystals are substances that change their state from a crystalline solid to an opaque liquid and then to a clear liquid if their temperature increases. Thus hot spots on the die surface will be observed as black areas indicating where the damage is, Figure 4.

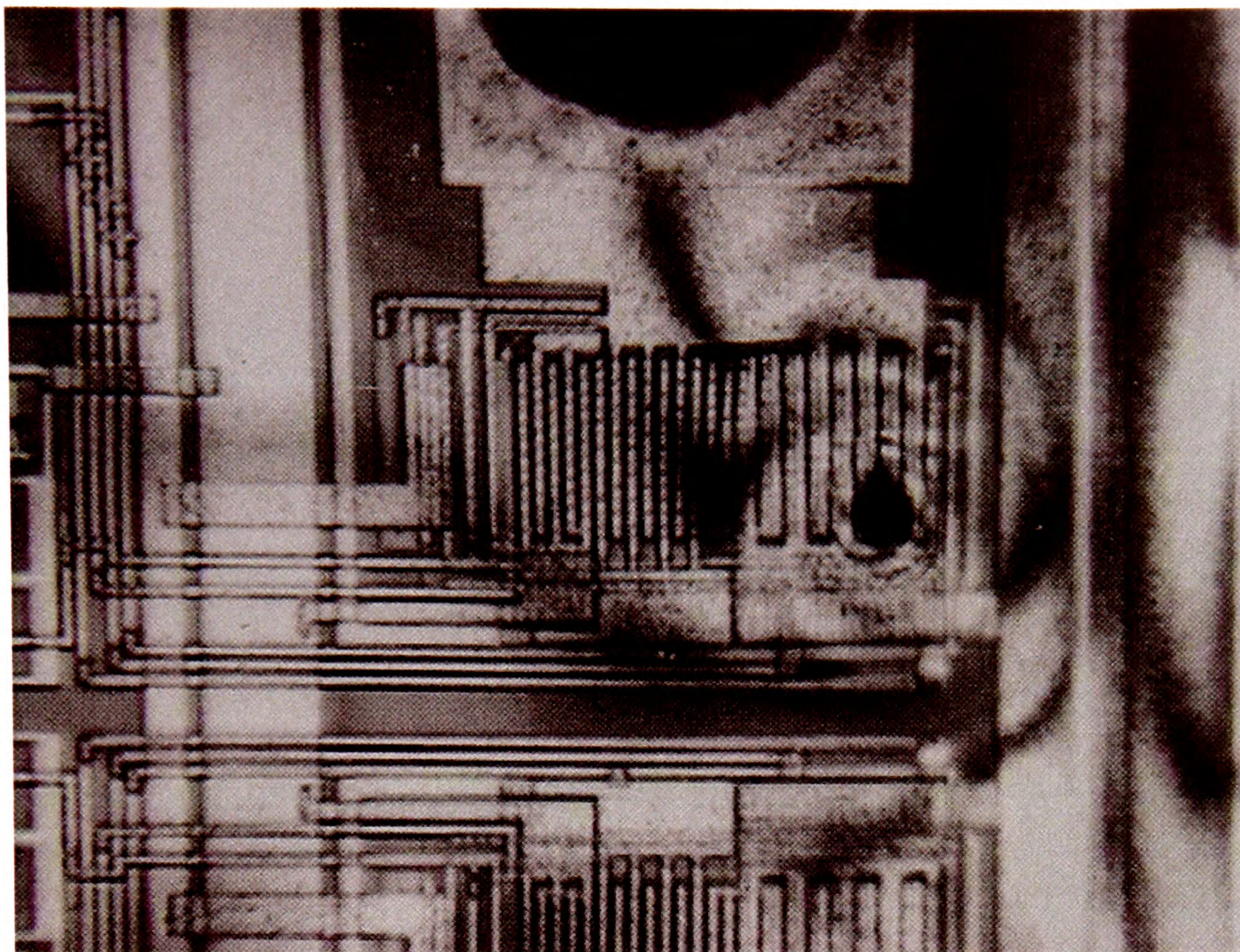


Figure 4 Liquid crystal analysis [Amerasekera97]

Fluorescent Microthermal Imaging (FMI) provides a hot spot detection technique with greater thermal sensitivity and better spatial resolution than liquid crystal analysis. Based on using a film with a temperature dependent fluorescent quantum yield, it generates high resolution thermal maps of the chip surface. Figure 5 shows one FMI detected hot spot.

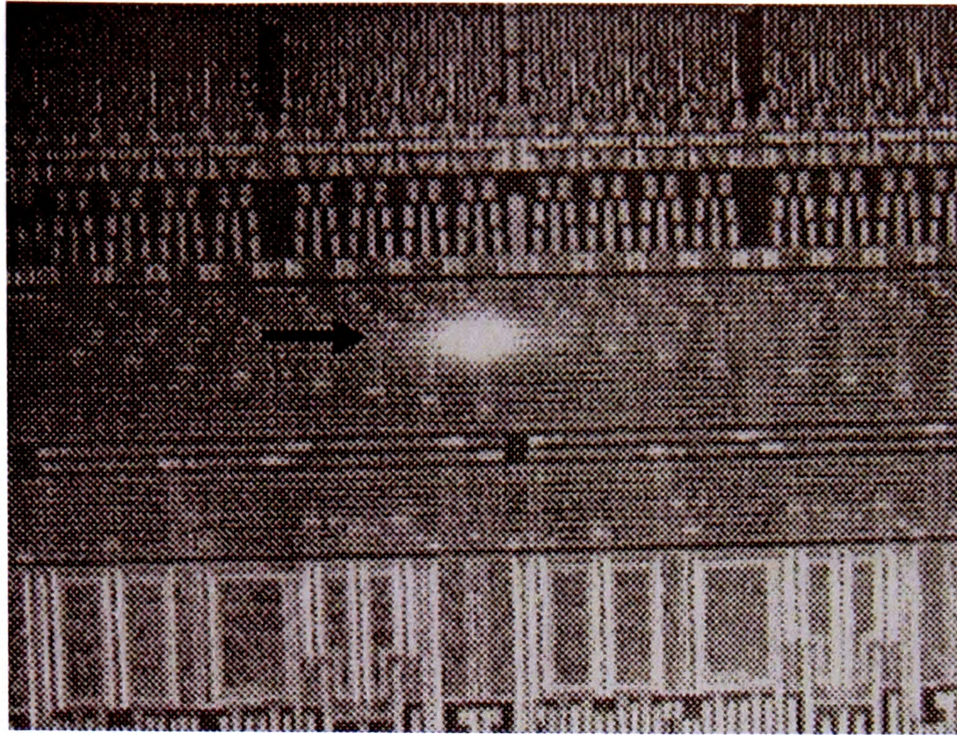


Figure 5 FMI detected hot spot [Wagner99]

Other techniques rely on the excessive light emission generated by the defect during the chip operation. Photon Emission Microscopy (PEM) is a common failure analysis technique for semiconductor devices. PEM collects visible and some near infrared wavelength photons emitted from transistors, *pn* junctions and other photon generating structures. The imaging light can be collected either from the silicon surface or through the silicon substrate and emerging from the back, when it is referred to as backside PEM [Wagner99] [Hora02].

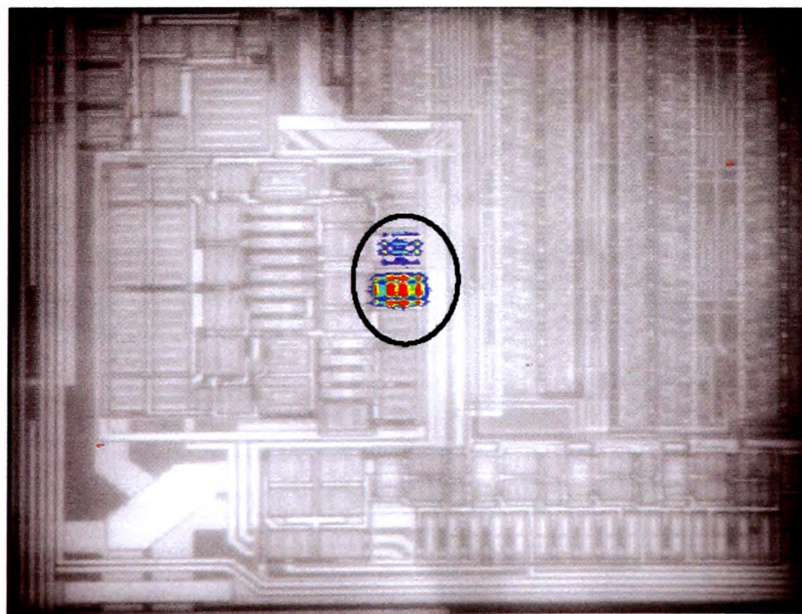


Figure 6 Result from backside PEM [Philips FA Lab]

Figure 6 shows a defect area spotted by backside PEM scrutiny. There are other techniques that take a more active approach by using the electron beams' ability to interact with the IC device. Both Charge-Induced Voltage Alteration (CIVA) and Light-Induced Voltage Alteration (LIVA) use the same electrical approach to generate photocurrent and heating, but with different probes. When the E-beam (in CIVA) or the optical beam (in LIVA) scan across the chip, the secondary carriers are generated at the *pn* junctions, causing current and voltage changes as the beam passes. Images of the photocurrent are then used to locate the defect area [Veendrick98]. These techniques rely heavily on the beam probing the face the problem of multi-level metallization. The shielding effect of these multi metal layers could compromise the techniques' effectiveness on fully processed wafers. Unless etching techniques are used to expose the beneath layers, only the top level metal can be imaged. Therefore during the chip's design phase, extra pads are sometimes deliberately added on the top layer metal to provide the probing points for some of the most critical signal nodes.

Electrical fault diagnosis

Electrical fault diagnosis has normally been categorized into two types: cause-effect diagnosis and effect-cause diagnosis. The first one is the most popular algorithm which has been extensively studied [Waicukauski89] [Abramovici90] [Pomeranz92] [Kunda93] [De95] [Sheppard96] [Hartanto97] [Chakravarty99] [Lavo01]. The cause-effect algorithm basically relies upon computer aided simulation tools to

understand how the circuit fails under the presence of a particular fault. A simulation of any fault instance produces a set of fault signatures, *i.e.* a list of all the test vectors and circuit outputs by which a fault should be detected. The process of cause-effect diagnosis is mainly to compare the fault signatures representing a fault candidate with what really happens on the tester, Figure 7. The general historical trend used to be to use very simple or exact matching, where the defect is assumed to behave in a way very close to the model used. In the last decade, there has been a movement toward more complicated matching and scoring system able to deal with a range of defect types and hard to model defects [Lavo97] [Venkataraman01] [Hora02].

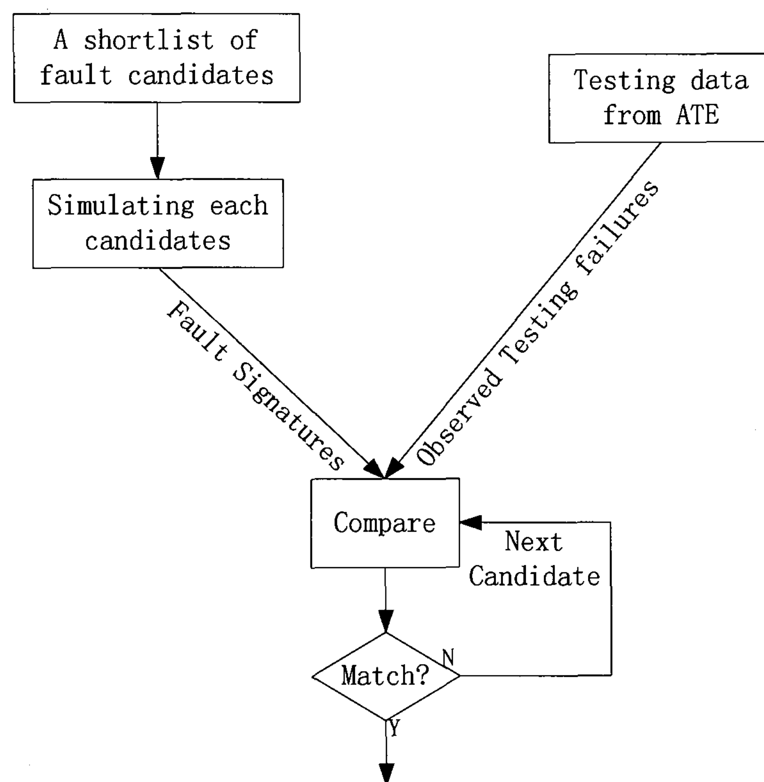


Figure 7 Flow of cause-effect diagnosis

Attempts have also been made to compress the size of the fault signature because the growing modern IC size and complexity would result in colossal simulation data if all possible faults were to be simulated and data restored. Several methods have been developed to reduce the size of the fault dictionary with some compromise in

diagnostic resolution [Lavo01] [Pomeranz92] [Chakravarty99].

The other main electrical fault diagnosis approach is called effect-cause [Abramovici80] [Rajski87] [Venkataraman97] [Hsu98]. As indicated by its name, effect-cause algorithms use a more intuitive way of starting from where the wrong outputs are observed on the circuits, the "effect". Through reasoning to trace back to the possible sources of failures, the "cause", effect-cause diagnosis most commonly suggests a logical location or area of the defect, but not necessarily a failure mechanism. Although a lot of effect-cause based works claim to be model independent, it has to be mentioned that a purely model-free algorithm is hard to achieve. One has to make assumptions and descriptions of the fault behaviours in order to proceed to any fault diagnosis, and it is difficult to make any assumptions and descriptions without any inference to a fault model. Most of the works implicitly use the stuck-at model to perform the path-tracing algorithm where assumption of propagation and sensitization of the faults are made. The path-tracing procedures start from the primary outputs with efforts to justify internal nets' values backwards simultaneously. If there are more possible ways to justify a value, a decision has to be made on one possible justification. If a decision leads to conflicts, the algorithm has to go back to the last decision point and make an alternative decision. The benefits of effect-cause diagnosis lie in the fact that no time consuming simulation and huge response data storing is needed beforehand. It can also be constructed to be general enough to cover the multiple fault diagnosis which would otherwise require special

treatment in cause-effect diagnosis. The disadvantage of effect-cause diagnosis is its inherent imprecision, usually a large area is implicated as faulty. Given its lack of accurate fault modelling, this drawback seems an inevitable result.

Deprocessing

Though electrical fault diagnosis may point to a suspected defect location as precise as a specific net, it is after all a speculation until the only ultimate visual evidence is achieved. Therefore, after the fault has been electrically located, physical measures have to be used to provide more visibility and accessibility to the areas below the surface where the electrical fault diagnosis indicates. Deprocessing can be viewed as the reversed process of IC manufacturing. Deposited layers have to be removed one by one. Deprocessing can be performed locally where a selected area is removed or globally where an entire layer is removed. There are basically three techniques to deprocess an IC: wet etch, plasma etch, and mechanical polish [Wagner99]. In many cases, a combination of these techniques is used to make a successful deprocessing. Wet etching relies on chemicals to etch out the selected layers. The primary concern of using chemicals is its isotropic nature, which leads to an undercutting effect. Undercutting effects cause narrow metal lines to have a tendency to lift off the surface and also unwanted etching of materials through holes such as vias, and subsequently etching away the underneath layer. Plasma etching is known as dry etching and normally consists of the following steps: a glow discharge produces chemically

reactive species from a relatively inert gas which diffuse to and get adsorbed by the surface to be etched. A volatile byproduct is produced when the species reacts with the surface and later desorbed from the surface. Figure 8 shows a finished plasma deprocessed sample with three levels of metal exposed for visual inspection.

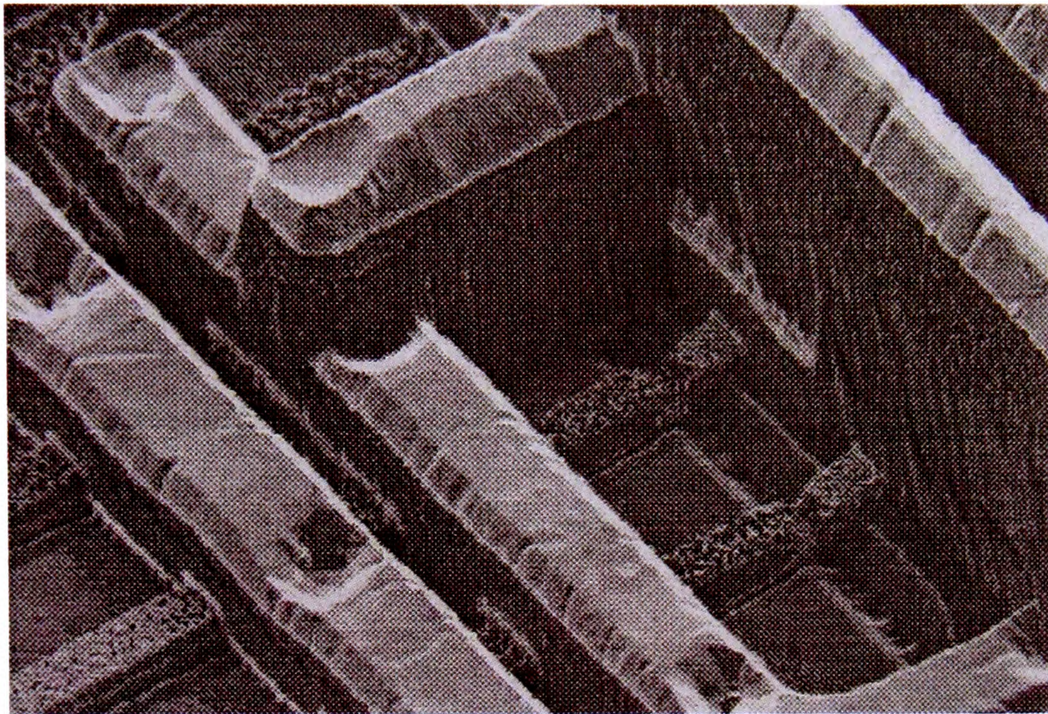


Figure 8 Plasma deprocessed sample [Wagner99]

Mechanical deprocessing is a physical measure using abrasive materials. The process is time consuming by nature. However, with increasing layers used in IC manufacturing, it has become more competitive with other techniques in regard of the overall deprocessing time. One advantage of mechanical deprocessing is the possibility to view features in the area of interest within the same plane. It is not possible to view, for example, metal and oxide structures at the same plane in wet and dry etching techniques, because removal of oxide would leave only the metal structure. Whatever the techniques used, deprocessing is a destructive procedure, therefore has to be conducted carefully. Improper and hasty deprocessing can result in loss of valuable information of the defects.

Physical Inspections

Some of the defects are not accessible even the defect surface is expose by deprocessing. For example, failures occur at the bottom of etched holes such as vias and contacts are not observable after deprocessing. A cross-section often gives more information about the defect. The most fundamental cross-section technique is manual cleaving of the wafer. With the help of semi-automated instruments developed specifically for cleaving, this technique can approach the defect area with an accuracy of 1.5 microns [Wagner99]. The other cross-section technique is called Focused Ion Beam (FIB), where a focused gallium ion beam cuts across the defect area, milling a rectangular hole. One side of the milled area is the cross-section plane. Figure 9 shows an example of FIB cross-section.

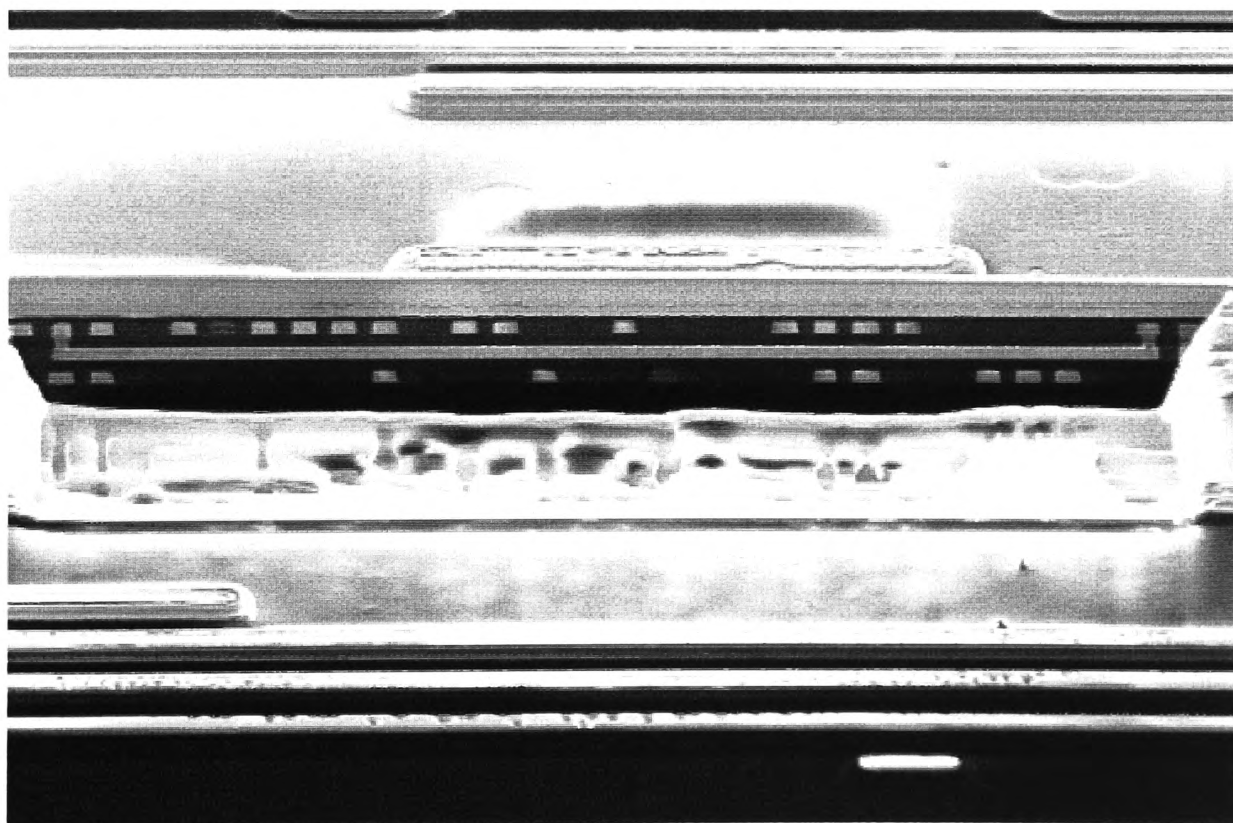


Figure 9 FIB cross-section [Philips FA Lab]

Improvements in ion beam resolution enable the cross-section to achieve submicron features with the precision that can not be achieved by the mechanical cleaving techniques. A disadvantage of FIB sectioning is the beam damage to the sample that may alter the surface of the cross-section. A cross-section operation is often followed by a staining step which applies chemical delineation and decoration to provide contrast in imaging the device features.

After the defect is located, it is important to inspect its material properties. This information is valuable to manufacturing process and is used to determine the source of the defect. Microscopes play a key role in defect inspections and some of the most commonly used ones are: optical microscopy, scanning electron microscopy, transmission electron microscopy. Each microscope has its own limitations. The probe used in the microscope and its energy or wavelength are important in determining the transparency of a material, and therefore the accessibility of the area of interest [Richards92].

Optical microscopy is easy to use. No vacuum is required, sample loading is therefore simple. Optical microscopy provides the ability to view through most of the thin film dielectrics used in IC products, a unique feature of its own. The major disadvantage of optical microscopy is its resolution, which is limited to approximately half the wavelength of the light used.

The decreasing size and increasing complexity of semiconductor devices demand some form of better resolution microscopy inspection. The Scanning Electron Microscopy (SEM) overcomes the deficiencies of its optical predecessor. Using a beam of electrons with energy between 0.2 and 50kV, it is capable of scanning features sizes smaller than 10nm. On top of that, SEM can function in different modes to perform sophisticated investigations on semiconductor materials and devices in which properties such as morphology, composition, carrier concentration, defect density, *etc.*, can be assessed. A picture from SEM is given in Figure 10

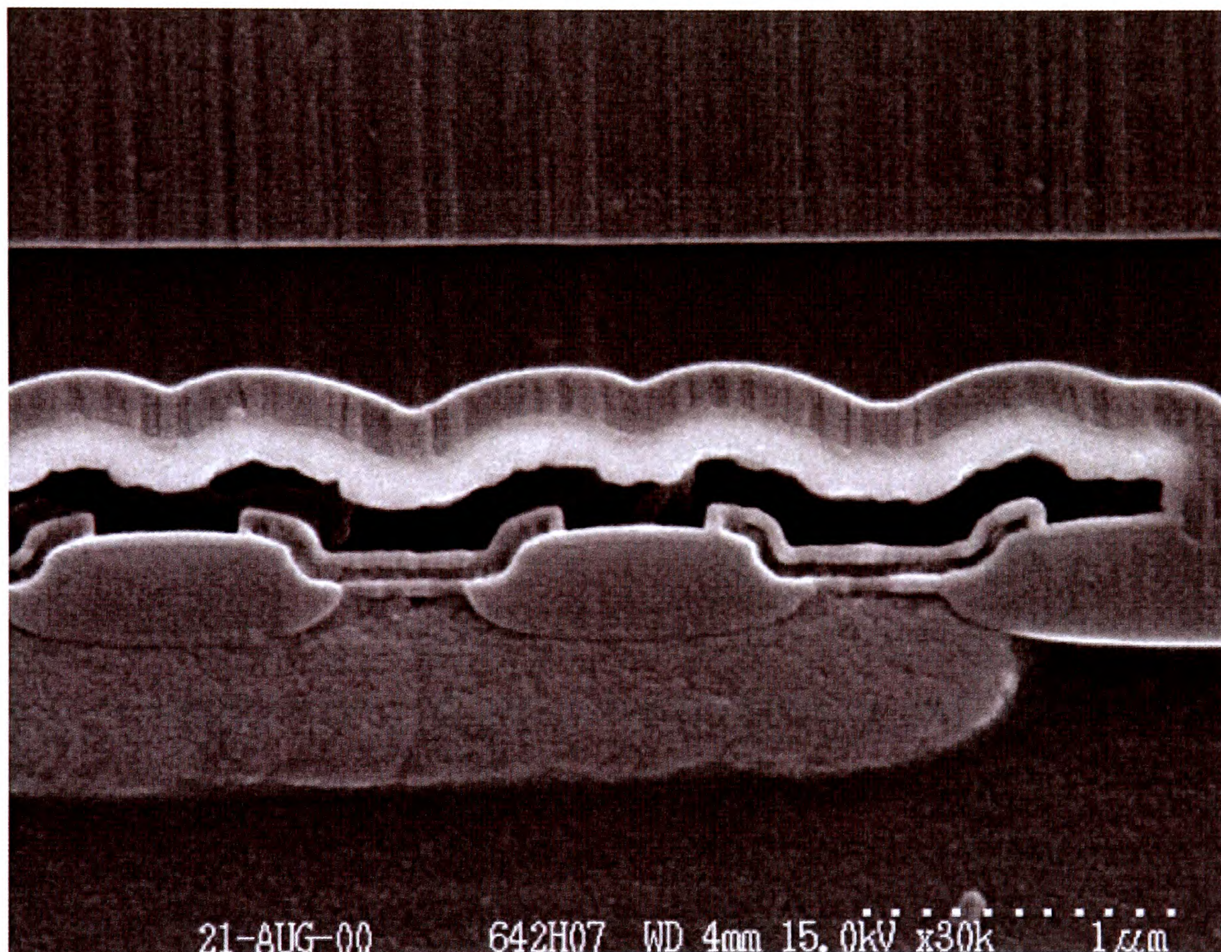


Figure 10 SEM picture of a cross-section [Philips FA lab]

Transmission Electron Microscopy (TEM) has been useful for physical inspection for many years. It has provided even higher resolution than the SEM. TEM requires preparation of extremely thin samples to allow electron transparency. The device structure has to be thinned to less than 3 μm so that it can be imaged in transmission.

Despite the difficult and time consuming sample preparation, TEM is one of the best techniques available for structural assessment with high spatial resolution and good picture contrast. TEM is very often used along with the cross-section inspection. Figure 11 shows a cross-section picture from TEM.

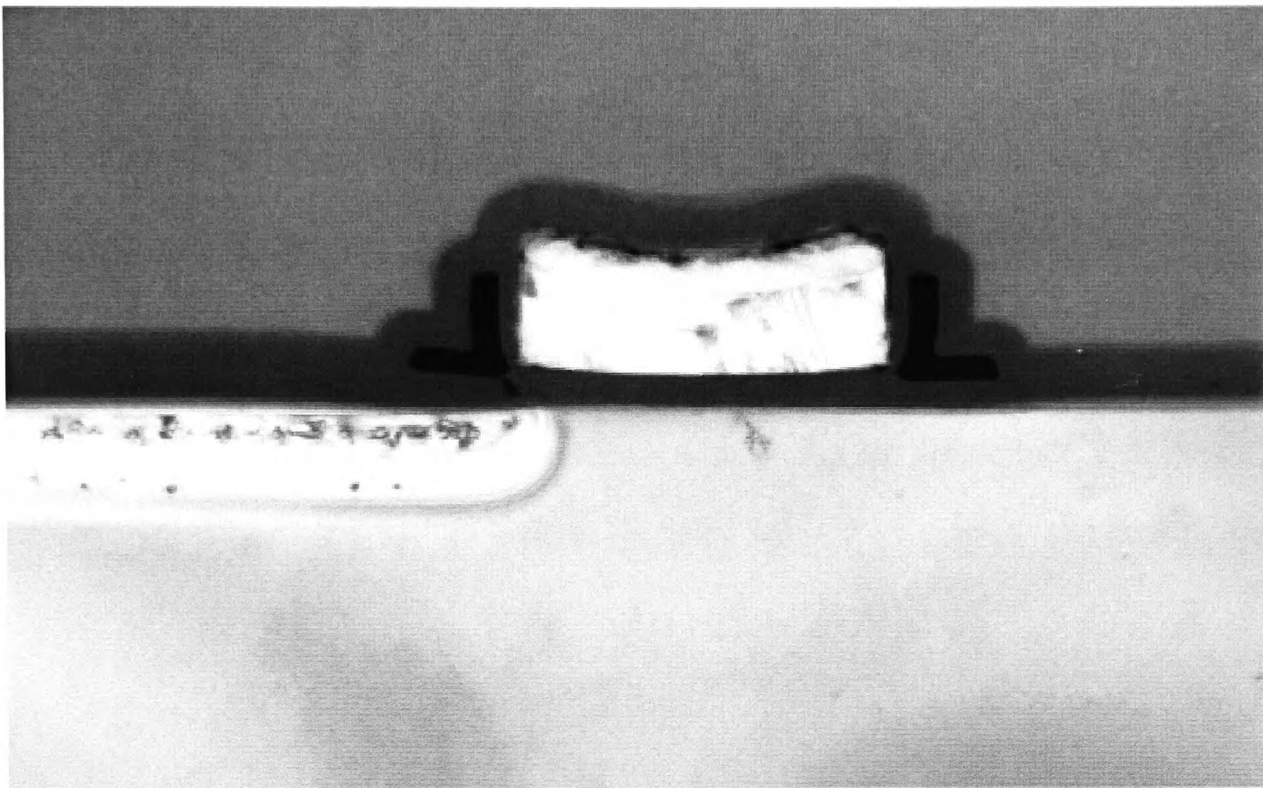


Figure 11 A cross-section TEM picture [Philips FA Lab]

Although pressures are gathering on the resolution of the defect inspections tools, SEM can be expected to be the main physical measures for many years. It has been used to develop many efficient and highly automated tools. Therefore until a replacement tool with higher resolution, such as TEM, can be efficient and automated enough to lower the cost of operation, SEM is expected to remain the most popular FA inspection tool.

Summary

Failure analysis provides the key information about the locations and mechanisms of

IC defects therefore suitable improvement can be made to the manufacturing process to increase the yield and the reliability of the IC products. It is essential to have a disciplined schedule to ensure that no relevant information is lost during the FA process, which often employs destructive physical measures. A basic flow of FA process consists of several steps such as visual inspection, electrical characterization, fault locating, deprocessing, and physical inspection. Each step is instrumental in offering extra information to pin down the defect on the basis of previous steps. The ultimate searching is conducted by various types of microscopes through the deprocessing and cross-section operations which expose the exact defect area to visual inspection.

Appendix B

Published Papers on Intra-gate Fault Diagnosis

1. Xinyue Fan, Will Moore, Camelia Hora, Guido Gronthoud, "*A Novel Stuck-At Based Method for Transistor Stuck-Open Fault Diagnosis*", IEEE International Test Conference, 2005, Austin, pp. 378-386.

2. Xinyue Fan, Will Moore, Camelia Hora, Guido Gronthoud, "*Stuck-Open Diagnosis with Stuck-At Model*", IEEE European Test Symposium, 2005, Estonia, pp. 182-187.

3. Xinyue Fan, Xinyue Fan, Will Moore, Camelia Hora, Mario Konijnenburg, Guido Gronthoud, "*A Gate-Level Method for Transistor-Level Bridging Fault Diagnosis*", IEEE VLSI Test Symposium 2006, Berkeley, pp. 266-271.

A Novel Stuck-At Based Method for Transistor Stuck-Open Fault Diagnosis

Xinyue Fan and Will Moore
Department of Engineering Science
Oxford University, OX1 3PJ
{xinyue.fan, will.moore}@eng.ox.ac.uk

Camelia Hora and Guido Gronthoud
Philips Research Labs
Prof. Holstlaan 4, 5656AA Eindhoven, Netherlands
{camelia.hora, guido.gronthoud}@philips.com

Abstract

While most of the fault diagnosis tools are based on gate level fault models, for instance the stuck-at model, many faults are actually at the transistor level. The stuck-open fault is one example. In this paper we introduce a method which extends the use of available gate level stuck-at fault diagnosis tools to stuck-open fault diagnosis. The method transforms the transistor level circuit description to a gate level description where stuck-open faults are represented by stuck-at faults, so that the stuck-open faults can be diagnosed directly by any of the stuck-at fault diagnosis tools. The transformation is only performed on selected gates and thus has little extra computational cost. This method also applies to the diagnosis of multiple stuck-open faults within a gate. Successful diagnosis results are presented using final wafer test data and an internal diagnosis tool from Philips.

***Keywords**: fault diagnosis, stuck-open fault, digital circuit*

1. Introduction

Fault diagnosis refers to the localizing of failures in chips that have been identified as defective. As the feature sizes get smaller and physical localization becomes more difficult, accurate software fault diagnosis is receiving more and more attention. Many fault diagnosis tools have been developed and have successfully diagnosed various types of faults such as stuck-at faults, bridging faults, open faults, etc [1] [2] [10] [11]. However, the existing tools only address the faults at the interconnection level. None of them is able to achieve a full diagnosis for the faults within a gate. In this paper we propose a diagnosis method for intra-gate stuck-open faults for digital circuits.

The stuck-open model assumes a particular transistor is permanently turned off (see Figure 1, transistor T1 is turned off) as a result, for example, of a failed transistor or connection break [3] [5] [6] [10]. Stuck-open faults can cause sequential behaviour and thus require a certain sequence of patterns in order to be detected.

Much work has been done in respect to stuck-open test pattern generation. Some authors have investigated the stuck-open fault coverage when a set of stuck-at test patterns is applied [4] [5]. Abd-El-Barr [7] has proposed a deductive method to generate robust stuck-open test patterns. The testing of open defects in memory address decoders is addressed in [9]. Most of the work has focused on the detection of stuck-open faults but with little attention to diagnosis.

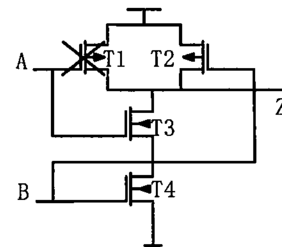


***Figure 1.** Transistor A stuck open*

Until now, the only systematic published work of stuck-open diagnosis is from Li [6] [8] [10]. Li's method [8] has to build up an excitation table for every type of gate manually, which will take a considerable amount of time for a large scale design. Also, Li's method requires the rewriting of a matching procedure which is already contained in most of the conventional diagnosis tools. To improve Li's method in an automatic way, we develop a transformation method which allows a stuck-at fault diagnosis tool to be applied directly on stuck-open fault diagnosis. The method is also applicable to diagnose multiple stuck-open faults within a gate, an issue never addressed before.

In Section 2, we examine the relationship between stuck-open and stuck-at faults and introduce the diagnosis terminology defined by Hora [2] and used in Li's work. In Section 3, we present the overall flow of our method. The transformation method is described in Section 4. The transformation method has been also described in our previous paper [13]. Experiments are conducted on

Philips final wafer test data. Results and analysis are reported in Section 5.

2. The relationship between stuck-open faults and stuck-at faults

To detect a stuck-open fault, a sequence of two patterns needs to be applied. Consider a NAND gate with a stuck-open fault in p-transistor T1 (Figure 1). Table 1 shows all possible patterns that we could apply, including the sequential effects - the previous value of *Z* in column 2. Column 5 is the expected fault free output (*Z*), column 6 is what we observed at the output (Obs) by performing electrical simulation when T1 is stuck open, and column 7 and column 8 are the fault signatures of *Z* stuck-at 0 (sa0) and *A* stuck-at 1 (sa1), respectively.

Pattern#	PreZ	A	B	ExpZ	ObsZ	Z-sa0	A- sa1
1	0	0	0	1	1	0	1
2	0	0	1	1	0	0	0
3	0	1	0	1	1	0	1
4	0	1	1	0	0	0	0
5	1	0	0	1	1	0	1
6	1	0	1	1	1	0	0
7	1	1	0	1	1	0	1
8	1	1	1	0	0	0	0

Table 1. *The behaviour of different faults in Figure 1*

We define two sets – *Obs* and *Sim*. The set *Obs* (**Observed Results**) represents the pairs of pattern/failed pin observed on the tester, and the set of *Sim* (**Simulation Results**) represents the pairs predicted by the fault simulation. In previous work on stuck-at diagnosis by Hora [2], two measurements have been used to judge the quality of the diagnosis. The term "Matching" is used to quantify the extent to which the observed failing results actually match with the simulation of a particular fault.

$$\text{Matching (M)} = \frac{|Obs \cap Sim|}{|Obs|} \times 100\%$$

A second term "Prediction" is used to quantify the extent to which simulations of a particular fault predict only those failing observed results.

$$\text{Prediction (P)} = \frac{|Obs \cap Sim|}{|Sim|} \times 100\%$$

Suppose a stuck-open occurs in T1 in Figure 1, from Table 1 we know by electrical simulation that Obs is {2/Z} (pattern number/failed pin). If Z stuck-at-0 is simulated, the Sim will be {1/Z, 2/Z, 3/Z, 5/Z, 6/Z, 7/Z}, and if A stuck-at-1 is simulated, the Sim will be {2/Z, 6/Z}.

For *Z* stuck-at-0,
Matching (M) = |{2/Z}| / |{2/Z}| = 1/1 = 100%
Prediction (P) = |{2/Z}| / |{1/Z, 2/Z, 3/Z, 5/Z, 6/Z, 7/Z}| = 1/6 =17%

And for *A* stuck-at-1,
Matching (M) = |{2/Z}| / |{2/Z}| = 1/1 = 100%
Prediction (P) = |{2/Z}| / |{2/Z, 6/Z}| = 1/2 =50%

As we can see, when a stuck-open fault is present inside a gate, a stuck-at fault at the output (either stuck-at-0 or stuck-at-1) will always give Matching = 100% and Prediction ≤ 100%, and if the stuck-open transistor has a direct input (in Figure 1, *A* is the direct input of transistor T1), the stuck-at fault at this direct input (either stuck-at-0 or stuck-at-1) will also give Matching = 100% and Prediction ≤ 100%.

With an open in a p-transistor, the gate output cannot be pulled up to a logic one via that transistor. If the output was previously a zero, it will stay at a zero and appear as a stuck-at fault at the output. But if the previous output was a one, or the output can be pulled up to a logic one via some other transistors, the output will be correct. It therefore follows that the *Obs* of a stuck-open will always be a subset of, or same as, the *Sim* of a stuck-at fault at the output. As a result, in the previous example we have Matching = 100% and Prediction = 17% for *Z* stuck-at 0.

When a stuck-open p-transistor has a direct input (as *A* for T1 in Figure 1), the *A* stuck-at-1 fault has the same effect of turning off T1. On top of that, it also helps to turn off other pull up paths and open the pull down paths. So, regardless of the previous output value, when a pattern is due to fail under T1 stuck-open model, it must fail under the *A* stuck-at-1 model. It therefore follows that the *Obs* of a stuck-open will always be a subset of, or same as, the *Sim* of a stuck-at fault at the direct input. As a result, we have Matching = 100% and Prediction = 50% for *A* stuck-at-1 in the previous example.

In general, when a stuck-open fault occurs, and we use stuck-at model to diagnose the circuit, both the output of the faulty logic gate and the direct input (if there is one) of the stuck-open transistor will report Matching = 100%. The relationship of their Predictions is: Prediction (output) ≤ Prediction (direct input, if there is one) ≤ 100%

3. Overall flow of stuck-open diagnosis

Figure 2 shows the overall flow of our method in comparison with the previous work of stuck-open diagnosis from Li [6] [8] [10]. Our method shares the same first two steps with Li's work. We perform an initial single stuck-at fault diagnosis in the first step. From the stuck-at diagnosis, we select those gates with either input or output diagnosed as Matching = 100% and Prediction < 100%. From there on, Li's work needs to compose an

excitation table to judge which of the signatures in stuck-at *Sim* will remain as stuck-open signatures. This excitation table is constructed manually on an individual gate basis, and every transistor will have its own excitation table. This could take a considerable effort for a large scale design. Li's work also has to build up another program to calculate Matching and Prediction for every stuck-open fault in the gate. It is not worthwhile to do this especially when we have so many diagnosis tools for stuck-at faults that can already calculate Matching and Prediction.

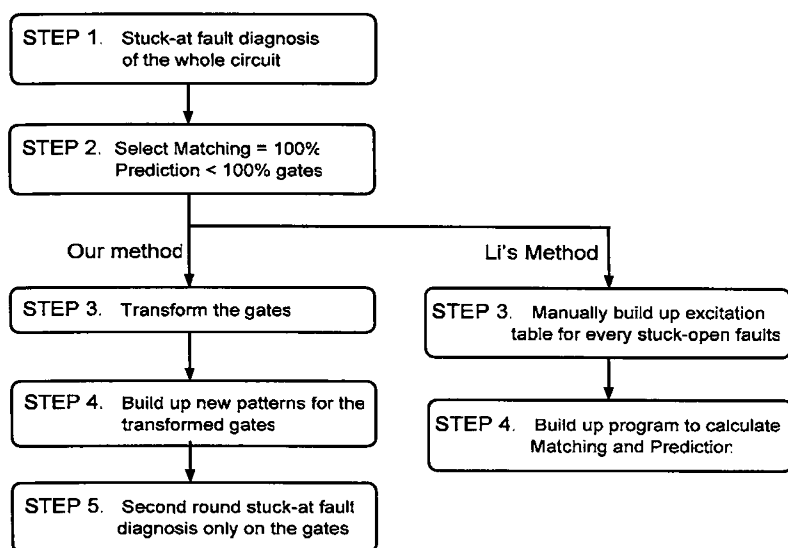


Figure 2. *Overall flow of our method in comparison with Li's work*

Instead of constructing an excitation table for each stuck-open fault, we transform the gates to a new form where all the stuck-open faults are represented by stuck-at faults. This transformation method will be introduced in the next section. After the transformation, new patterns are constructed from both the previous and current input values of the transformed gates. For every stuck-at pattern that propagates the target stuck-at fault to the output pins, we find out its previous pattern. The two consecutive patterns that appear on the transformed gate are concatenated to form a new pattern. Finally, the stuck-at diagnosis is performed once more but this time only on the transformed gates. If successful, the stuck-open faults will be diagnosed in the names of the corresponding stuck-at faults.

In doing so, any stuck-at fault diagnosis tool will be able to diagnose the stuck-open fault directly. This method is very straightforward and can be easily integrated into commercial diagnosis tools.

4. Transformation method

The basic idea of the transformation is to represent a stuck-open fault with a stuck-at fault so the stuck-at diagnosis tool can be used, instead of building a new stuck-open fault diagnosis tool. The following example explains how the transformation works.

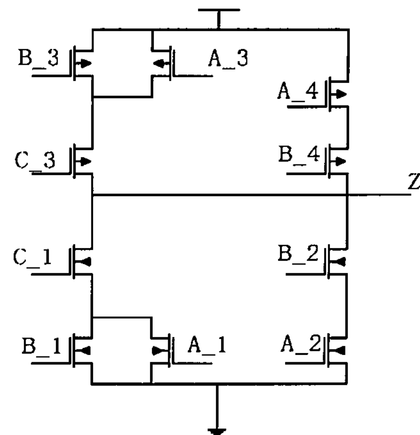


Figure 3. *Schematics of the example circuit*

Figure 3 is the schematic of an example logic gate with three inputs (*A B C*) and one output (*Z*). We split each input signal into a number of branches, one for each transistor the input is connected to. For the gate from our example, input *A* will be split in *A_1, A_2, A_3* and *A_4* (see Figure 3). We transform the transistor level schematic to the gate level by the following rules.

1. For n-transistors

(a) Replace all the n-transistors with the element as shown in Figure 4. We want to guarantee that a zero voltage from the source will be transmitted to the drain when value on the input (*A* in our example) is one (transistor turned-on). When the input *A* is one (transistor turned-on) and the source signal is zero, the drain will be zero. Otherwise, the drain will be one.

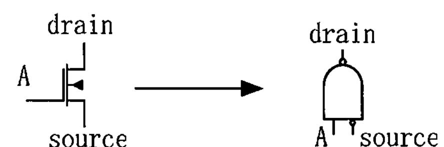


Figure 4. *Replacement of n-transistor*

(b) Place an AND gate where a parallel connection between transistors is present, as Figure 5. Here we also want to guarantee that the zero signals from the source will be transmitted to the drain as long as either one of the transistors is turned-on. Therefore the AND gate makes sure if one of the drains is zero, the output will be zero. If none of the drains is zero, the output will be one.

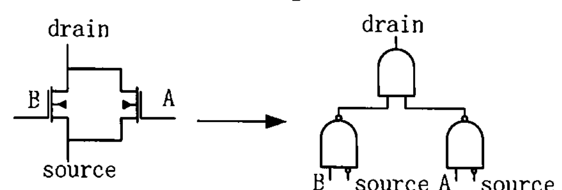


Figure 5. *Replacement of parallel n-transistors*

2. For p-transistors
(a) Replace all the p-transistors with the element as shown in Figure 6. The purpose is similar, to guarantee the propagation of logic one from the source.

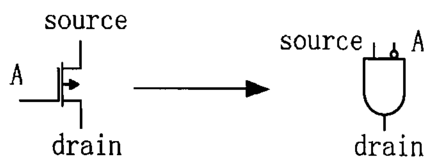


Figure 6. *Replacement of p-transistor*

(b) Place an OR gate where a parallel connection is present, as Figure 7.

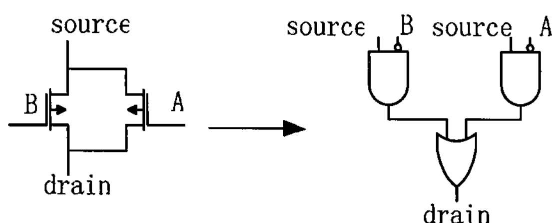


Figure 7. *Replacement of parallel p-transistors*

The view after the whole transformation of the gate shown in Figure 3 will be as Figure 8. As one can see, the whole graph consists of a *P* Part and an *N* Part. Because we are only interested in stuck-open faults, we assume the only possible faults will be stuck-at-0 on branch inputs for the *N* Part and stuck-at-1 on branch inputs for the *P* Part. For instance, for our example, a stuck-open fault at *C_1* in Figure 3 will be represented by a stuck-at-0 fault at *C_1* in Figure 8 and a stuck-open fault at *A_3* in Figure 3 will be represented by a stuck-at-1 fault at *A_3* in Figure 8.

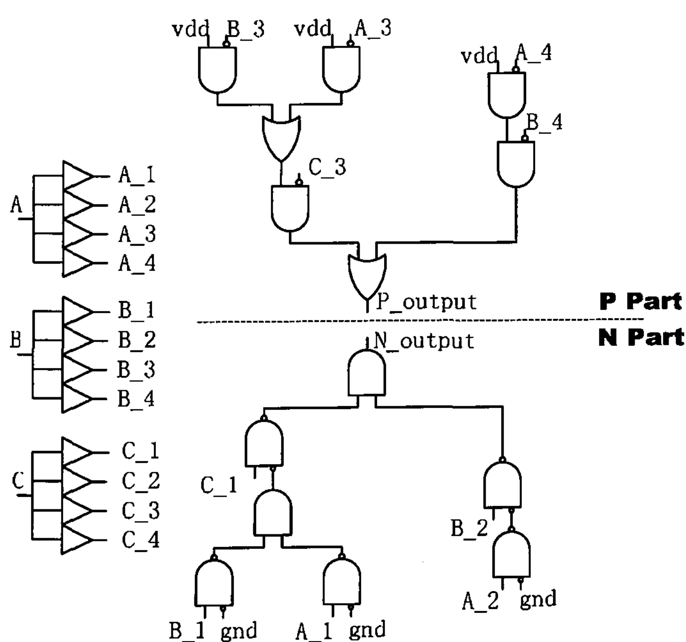


Figure 8. *The first phase of transformation*

Let us see how we represent the stuck-open faults. In the faulty free case, the P_output and N_output give the same responses. They both give the results that we would see at *Z* in Figure 3. Consider now the faulty case where there is a stuck-at fault in one of the input branches. To excite a stuck-open fault, we need two patterns, the first pattern sets the output to a certain value, the second pattern has to make sure that the target transistor plays an indispensable role to drive the output to the opposite value. Thus when the target transistor is stuck open, the output fails to go to the opposite value and keeps the previous one. From the structural point of view, this means the stuck-open transistor has to control the only serially conducting path.

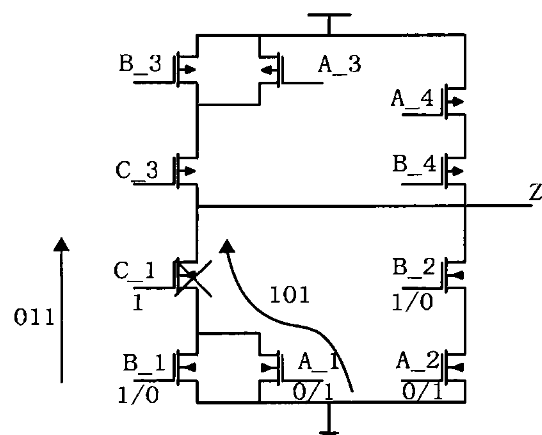


Figure 9. *Stuck-open fault at C_1*

For instance in Figure 9, to excite a stuck-open at *C_1*, the first pattern (for instance *ABC*=000) sets *Z* to one, the second pattern should be either *ABC*=011 or *ABC*=101. The expected pull-down paths for these two patterns are depicted by the arrows on Figure 9. In either case, *C_1* controls the only conducting path.

Let us see how Figure 8 represents the controls that *C* has over this scenario. The placement of AND and OR gates makes sure that the stuck-at fault propagates only when there is no parallel conducting path, in other words, the stuck-at fault controls the only conducting path. For instance in Figure 10, under either of the second patterns (*ABC*=011 and *ABC*=101) that excite the *C_1* stuck-open in Figure 9, the stuck-at-0 fault at *C_1* in Figure 10 is also propagated to N_output.

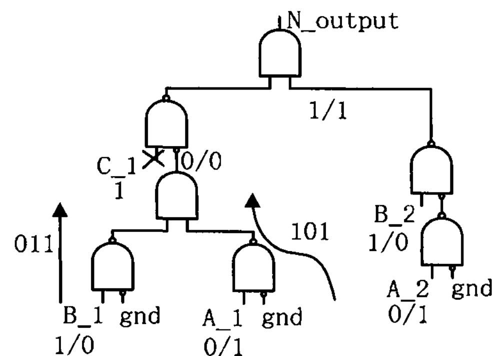


Figure 10. *Stuck-at fault at C_1*

Now, only when the stuck-open second pattern excitation requirement is met, will either P_output or N_output in Figure 8 fail.

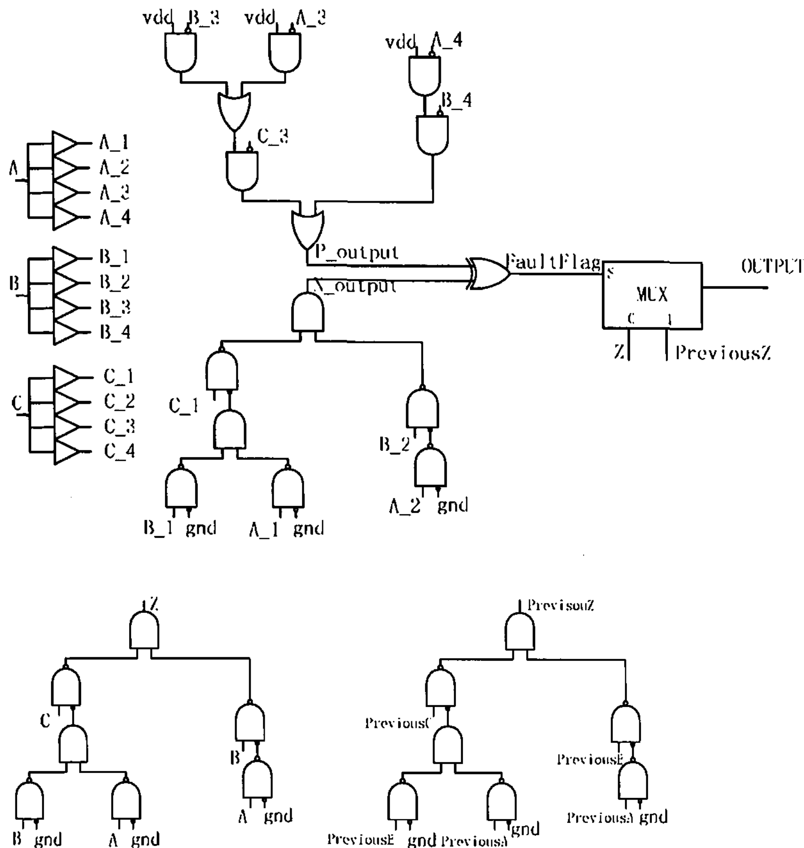


***Figure 11.** Final picture of transformation*

However, not only does the second pattern excitation requirement need to be met, but the first pattern excitation requirement, (*i.e.* to set up the first pattern output value at the opposite of the second), also needs to be met before we can see a fail at the final output. Here comes the final transformation shown in Figure 11. Two sub-circuits are used to generate the fault free value of the current *Z* and the previous *Z*. The P_output and N_output are connected by an XOR gate to generate a FaultFlag signal, which will be zero when no fault happens and one otherwise.

When there is no stuck-at fault, P_output=N_output, and FaultFlag=0, the MUX selects the correct current *Z* value, so the OUTPUT will not fail. When a stuck-at fault propagates (the second pattern requirement of stuck-open excitation is met), P_output $\neq$ N_output, and FaultFlag = 1. The MUX then selects the previous *Z* value, which determines if the OUTPUT fails or not. If the previous *Z* value is different from the correct current *Z* value, the OUTPUT fails. If they are the same, the OUTPUT does not fail. By adding this extra MUX, the first pattern requirement of stuck-open excitation is also met. Now, all the stuck-open faults in the *P* part are represented by the corresponding stuck-at-1 faults and all the stuck-open faults in the *N* part are represented by the corresponding stuck-at-0 faults.

The example above shows how a single block gate is transformed. Now we extend this method to gates with multi blocks like gate nds4ab in Figure 12.

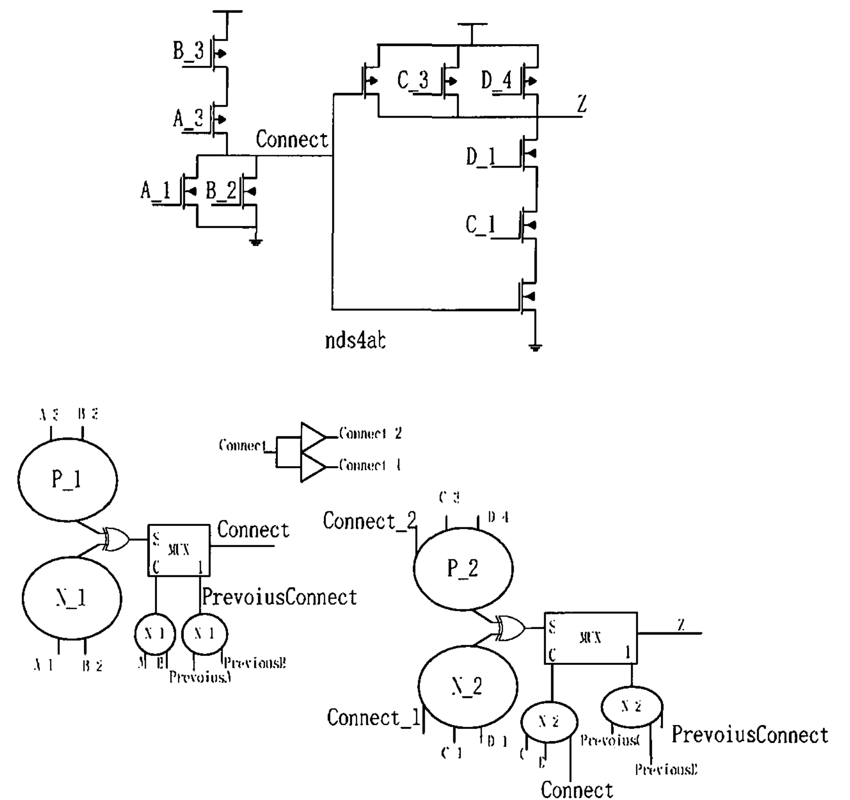


***Figure 12.** Multiple blocks gate transformation*

Each of the two blocks is transformed to *P* part and *N* part like P_1, N_1 for the first block and P_2, N_2 for the second block. The transformation procedure for each block is exactly the same as single block gate, with signal '*Connect*' taken as an input signal of the second block. Notice signal *'PreviousConnect'* is taken to generate the previous output of the second block in Figure 12.

5. Experimental results

In our previous paper, experiments based on electrical simulation are performed and successful stuck-open diagnosis results on the transformed circuits are achieved on three example gates [13]. Here we further examine our method using the final wafer test data from 25 wafers of Philips 0.18μm design, which has ~500k logic gates. A stuck-at diagnosis tool (FALOC, a Philips internal CAT tool) first diagnoses the faulty dies and suggests the likely stuck-at faulty nets. 43 dies are diagnosed as having stuck-at nets with Matching = 100% and Prediction < 100%. These results might be due to stuck-open transistors in one of these gates connected to the faulty nets and these gates are therefore taken as the targets for our transformation.

The design uses the full scan test strategy, so testing patterns are shifted in through the scan chains and applied at the normal cycle. For all the normal cycle (current)

#Die	Gate Type	Open Transistor	M (Matching) and P (Prediction) before transformation	M (Matching) and P (Prediction) after transformation
1	nd3	p-transistor at input C	C stuck-at-1 M = 100% P = 25%	C_4 stuck-at-1 M = 100% P = 100%
2	nd2	p-transistor at input A	A stuck-at-1 M = 100% P = 14%	A_3 stuck-at-1 M = 100% P = 100%
3	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 75%	A_1 stuck-at-0 M = 100% P = 100%
4	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 75%	A_1 stuck-at-0 M = 100% P = 100%
5	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 75%	A_1 stuck-at-0 M = 100% P = 100%
6	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 75%	A_1 stuck-at-0 M = 100% P = 100%
7	ao36	n-transistor at input A	A stuck-at-0 M = 100% P = 79%	A_1 stuck-at-0 M = 100% P = 100%

Table 2. *Fully diagnosed single stuck-open*

patterns that propagate the specific stuck-at fault to the primary output pins, we find out the logic values applied on the inputs of the suspected stuck-open gates, These values will be applied at the corresponding (current) inputs of the transformed circuit, for instance, inputs A, B and C in Figure 11. To get the previous input values on the suspected gates, we trace backward, starting from the last shift in cycle, until we find the first instance of the input values on the suspected gates being different from the input values of the normal cycle. (The speed of pattern being shifted into the scan chain is 100ns per cycle. From electrical simulation we have observed that the effect of a stuck-open fault can last for several microseconds. This also means the effect of stuck-open fault can last for several tens of cycles if the input values of the suspected gates do not change. However, in our experiment, it turns out that we have never needed to trace back more than 10 cycles before we see a change in the input values.) These input values are taken as previous values (like PreviousA, PreviousB, PreviousC in Figure 11), concatenated with current values from the normal cycle to form the new patterns for the transformed gate. Having built the new patterns for every suspected gate, the transformed circuit and the new patterns are then compiled to be diagnosed by the stuck-at diagnosis tool again.

The 43 suspected dies are subjected to this procedure. Seven dies are fully diagnosed as having single stuck-open transistors and three of them are diagnosed by further analysis of timing skew effect or multiple stuck-open faults in one gate. The rest of the dies can not be identified as having stuck-open faults by our method. Following is the detail of the results.

Results A. Fully diagnosed stuck-open

Seven dies have been diagnosed as having single stuck-open fault with Matching = 100% and Prediction = 100%. Table 2 shows the type of the gate and the location of the open transistor. Column 4 and column 5 compare the Matching and Prediction before and after the transformation.

Figure 13 shows the transistor schematics of the gates and their open transistors. Note we have separated the fan-out branches to each transistor to produce A_1, A_2, *etc.* The stuck-at fault in the last column of Table 2 indicates the corresponding transistor as stuck-open.

In fact, the stuck-at fault in the last column of Table 2 is diagnosed together with other logically equivalent faults in the transformed circuit. For gate ao36, the report contains following faults diagnosed with Matching = 100% and Prediction = 100%: net=*A_2* SA0, net=*A_1* SA0, net=*E_1* SA0, net=*E_2* SA0 and some extra internal nets created during the transformation. As mentioned earlier, the only stuck-at faults considered as possible are stuck-at-0 (SA0) faults in the *N* Part and stuck-at-1 (SA1)

#die	Gate type	Open Transistor	M and P before transformation	M and P after transformation	M and P after timing skew pattern modification
8	nr2	n-transistor at input A	A stuck-at-1 M = 100% P = 53%	A_1 stuck-at-1 M = 60% P = 100%	A_1 stuck-at-1 M = 100% P = 100%
9	nr2	n-transistor at input A	A stuck-at-1 M = 100% P = 52%	A_1 stuck-at-1 M = 80% P = 100%	A_1 stuck-at-1 M = 100% P = 100%

Table 3 *stuck-open affected by timing skew*

#die	Gate type	Open Transistor	M and P before transformation	M and P after transformation	M and P after multiple stuck-open modification
10	ao52	n-transistor at input C and D	Z stuck-at-0 M = 100% P = 22%	C_3 stuck-at-1 M = 29% P = 100% D_4 stuck-at-1 M = 57% P = 100%	MULTI stuck-at-1 M = 100% P = 100%

Table 4 *Multiple stuck-open faults in one gate*

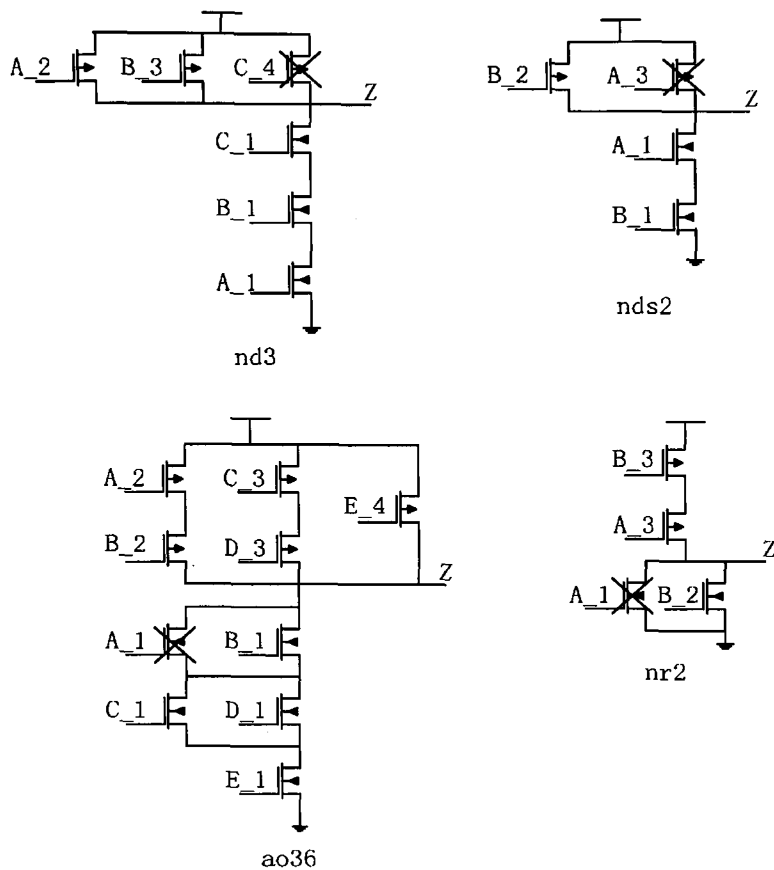


***Figure 13.** Location of stuck-open transistors*

faults in the *P* Part. In this particular case, that includes *A_1* and *E_1* for SA0, the rest are the redundant faults which can be removed. *E_1* is also not a realistic candidate, because if *E_1* was stuck open, *Z* would be completely detached from the ground and would thus give out Z stuck-at-1 with Matching = 100% and Prediction = 100% in the first round stuck-at fault diagnosis, rather than the result of Matching = 100% and Prediction < 100% that we actually see. The reason *E_1* is also reported is that under the assumption *E_1* is stuck-open, the previous values we draw out from the shift-in cycles are not valid in the sense that whatever the input values are, Z is always 1. Thus, in our experiment, those transistors that can singly decide the conducting of *P* part or *N* part are not considered as valid stuck-open candidates. As a result, *A_1* SA0 remains to be the only valid diagnosis result, indicating *A_1* is the open transistor.

Results B. Stuck-open affected by timing skew

The timing skew problem [10] [12] [13] can arise when two inputs change their values between the previous pattern and the current pattern. Different intermediate patterns may be formed depending on the timing of the transition. (See Figure 14)

When the timing skew problem takes control, the real previous values at the inputs and outputs of the suspected gates are in fact the intermediate values instead of the values we draw from the shift in cycles.

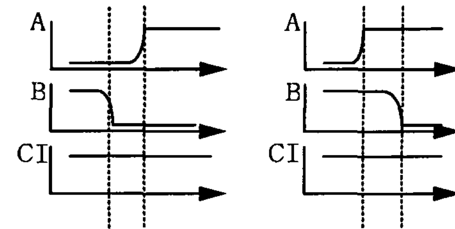


***Figure 14.** Timing skew*

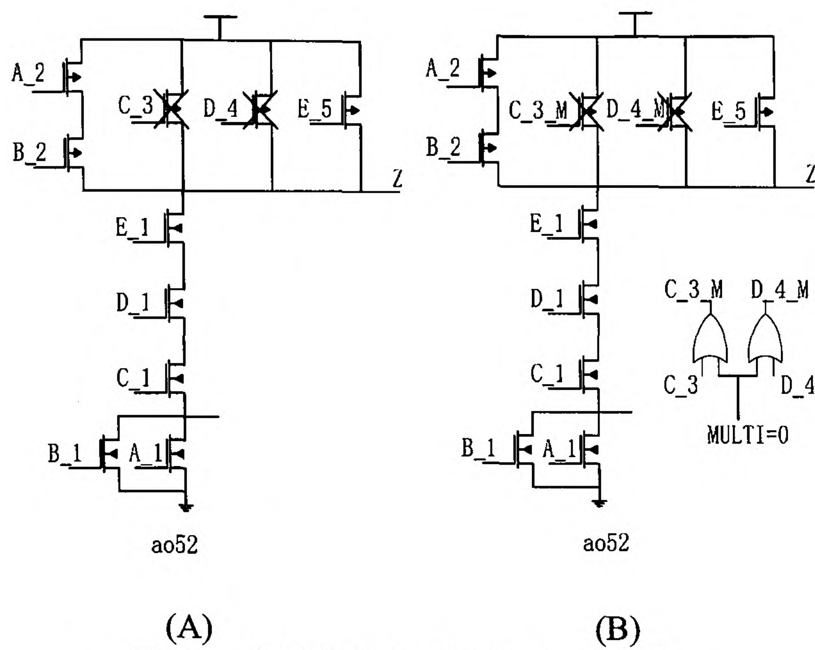
In some cases, we do not have perfect diagnosis but Matching < 100%, Prediction = 100% after the transformation. This means some pattern/pin pairs observed failed on the tester are not predicted by the fault model. We try to explain this with the timing skew problem and succeed in two gates (both are nr2 in Figure 13). For these two gates, we look at the values at the inputs of the gate and discover that the loss of Matching is due to a certain sequence of patterns, {previous AB=01; current AB=10}. We assume B changes value earlier than A, *i.e.* the intermediate value is AB=00, and we replace the sequence with {previous AB=00, current AB=10} before applying the patterns to the transformed circuit again. Under this assumption, we achieve both Matching = 100% and Prediction = 100% as shown in the last column of Table 3. The similar phenomenon is reported in Li's work [10]. However the timing skew effect is far more complicated than the two samples we explained here. More pattern sequences can be affected when the number of inputs increases, influencing both the Matching and Prediction. To explain it, we need to investigate the exact timing information of the whole circuit, which is beyond the scope of this paper. Given the complexity of the timing simulation, the value of doing this is also questionable.

Results C. Multiple stuck-open faults in one gate

Unlike the samples in Result A and Result B, there is one sample with a gate for which only the output is diagnosed as stuck-at-0 Matching = 100%, Prediction < 100% in the initial stuck-at fault diagnosis (Figure 15 (A)). The gate is transformed and diagnosed again. Two input nets, *C_3* and *D_4*, report Matching < 100%, Prediction = 100% after the transformation.

Multiple stuck-open faults can lead to a phenomenon like this. With only one fault modeled, some of the failed pattern/pin pairs caused by the other faults and those only caused by the simultaneous presences of multiple faults are not predicted, leading to a low Matching for both of the single fault diagnosis candidates.

To model the multiple stuck-open effect, we modify the gate and inject an additional input signal 'MULTI' as shown in Figure 15 (B), either before or after the transformation. 'MULTI' is set to zero and therefore



(A) (B)

***Figure 15.** Multiple stuck-open faults*

does not influence the normal behaviour of the gate. If 'MULTI' is stuck-at-1, both *C_3_M* and *D_4_M* are also stuck-at-1, representing the effect of multiple stuck-open of p-transistor C_3 and D_4. (The same strategy also applies for the multiple n-transistors opens, with the OR gate replaced by AND gate and 'MULTI' given the input value of one instead.)

After this modification, we find out that 'MULTI' is diagnosed as stuck-at-1 with Matching = 100% and Prediction = 100%, indicating p-transistor C_3 and D_4 are both stuck-open. Results before and after the transformation are given in Table 4.

Analysis of the layout gate ao52 reveals a possible explanation of the multiple open. An open contact illustrated in Figure 16 can disconnect both p-transistor C_3 and D_4 from the output Z. To the best of our knowledge, this is the first time experimental diagnosis results have pinpointed multiple stuck-open transistors within a gate.

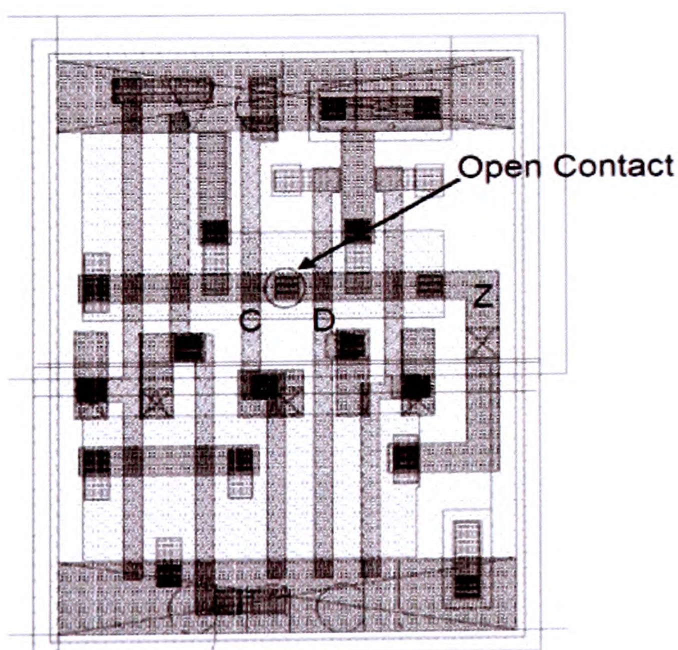


***Figure 16.** Layout analysis of ao52*

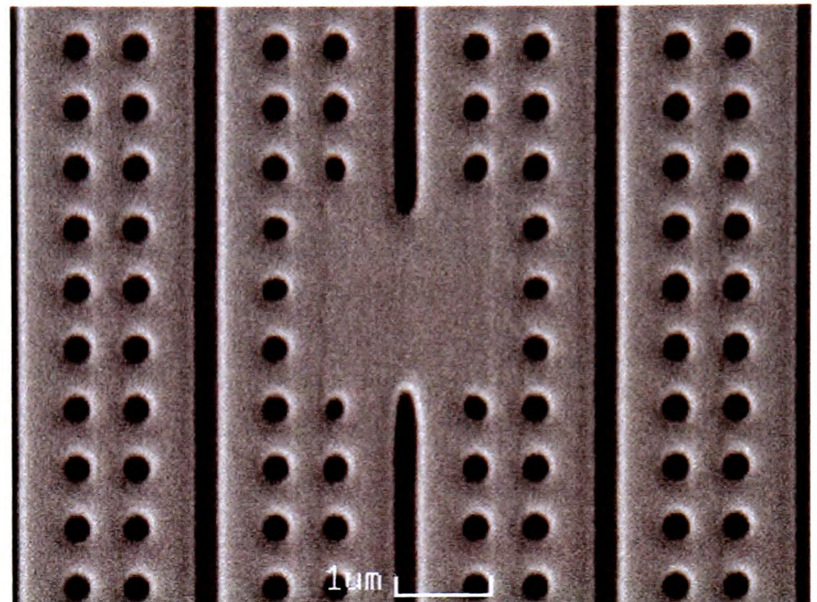


***Figure 17.** Missing Patterning*

The ten samples diagnosed above are not available anymore to perform FA to identify the defect that causes the stuck-open faults. We analyzed the process data still available and all the samples belonged to wafers with known open contact problems. These open defects are caused by missing patterning as in the example shown in Figure 17, which is very likely to be the cause of the stuck-open faults we have diagnosed.

6. Conclusions

To diagnose stuck-open faults, a method of transforming from transistor level schematic to a logic gate level description has been proposed. By this method, a stuck-open fault in the transistor level can be represented by a stuck-at fault at the logic gate level. A commercial stuck-at diagnosis tool can be brought in to pick out the stuck-open fault directly without the need to build up excitation tables and rewrite a part of the diagnosis tool, which has been proposed previously [10]. The method is also adaptable to diagnosis of opens affected by timing skew and to the diagnosis of multiple stuck-open faults. Also, because we transform only those cells identified by a first-pass stuck-at fault diagnosis, our method is independent of the size of the circuit. The required transformations can be automated and can readily be precomputed for all cells in our library.

Experiments have been performed on the final wafer test data of 25 wafers of a Philips 0.18μm design. Seven single stuck-open faults and one multiple stuck-open fault are completely diagnosed. Another two single stuck-open faults are also diagnosed based on the assumption of

timing skew effects. Locations of stuck-open transistors are given. We hope to provide with more diagnosis results and failure analysis information in the near future and will report progress in the final version of this paper.

7. Acknowledgement

The authors would like to thank Paul Volf from Philips Semiconductors for helping with the experimental data and inline information, Ananta Majhi, Mohamed Azimane, Maurice Lousberg of Philips Research and Stefan Eichenberger of Philips Semiconductors for their contributions to this work.

[1] S. Venkataraman and S. B. Drummonds, "POIROT: A Logic Fault Diagnosis Tool and Its Applications", *International Test Conference,* 2000, pp.253-262.

[2] C. Hora *et al*, "On Electrical Fault Diagnosis in Full-Scan Circuits", *International Workshop on Defect Based Testing,* 2001, pp. 17-22.

[3] S.M. Menon *et al*, "Testable Design of BiCMOS Circuits for Stuck-Open Fault Detection using Single Patterns", *VLSI Test Symposium*, April 1993, pp. 296-302.

[4] B. W. Woodhall *et al*, "Empirical Results on Undetected CMOS Stuck-Open Failures", *International Test Conference,* 1987, pp. 166-170.

[5] S. D. Millman and E. J. McCluskey "Detecting Stuck-Open Faults with Stuck-At Test Sets", *IEEE Custom Integrated Circuits Conference*, 1989, pp. 22.3.1-22.3.4.

[6] J. C. M. Li *et al*, "Testing for Resistive and Stuck Opens," *International Test Conference*, 2001, pp. 1049-1058.

[7] M. H. Abd-El-Barr *et al*, "Transistor Stuck-Open Fault Detection in Multilevel CMOS Circuits", *Great Lakes Symposium on VLSI*, 1999, pp. 388-392.

[8] J. C. M. Li, "Test and Diagnosis of Open Defects in Digital CMOS IC," Ph.D. Dissertation, Stanford University, 2002.

[9] M. Sachdev, "Open Defects in CMOS RAM Address Decoders," *IEEE Design and Test of Computers*, vol. 14, no. 2, 1997, pp. 26-33

[10] J. C. M. Li *et al*, "Diagnosis for Sequence Dependent Chips", *VLSI Test Symposium*, 2002, pp.187-192.

[11] Y. Sato *et al*, "A Persistent Diagnostic Technique for Unstable Defects", *International Test Conference,* 2002, pp.242-249

[12] S.M. Reddy *et al*, "On Testable Design for CMOS Logic Circuits", *International Test Conference,* 1983, pp. 435-444

[13] X. Fan *et al*, "Stuck-Open Diagnosis with Stuck-At Model", to be published in *IEEE European Test Symposium,* 2005

Stuck-Open Fault Diagnosis with Stuck-At Model

Xinyue Fan and Will Moore
Department of Engineering Science
Oxford University, OX1 3PJ
{xinyue.fan, will.moore} @eng.ox.ac.uk

Camelia Hora and Guido Gronthoud
Philips Research Labs
Prof. Holstlaan 4, 5656AA Eindhoven, Netherlands
{camelia.hora, guido.gronthoud} @philips.com

Abstract

While most of the fault diagnosis tools are based on gate level fault models, many faults are actually at the transistor level. The stuck-open fault is one of them. In this paper we introduce a stuck-open fault diagnosis method based on the stuck-at fault model. First we investigate how stuck-open faults show in the stuck-at diagnosis. Based on the stuck-at diagnosis result, a transformation method is used to represent stuck-open faults. This method transforms the transistor level circuit description to a gate level description so that the stuck-open faults can be diagnosed directly by any of the stuck-at fault diagnosis tools. After the transformation, all the stuck-open faults are fully diagnosed by FALOC, a gate level fault diagnosis tool from Philips.

1. Introduction

Fault diagnosis refers at localizing failures in chips that have been identified as defective. As the feature size goes smaller and physical localization becomes more difficult, accurate software fault diagnosis is receiving more and more attention. Many fault diagnosis tools have been developed and successfully diagnosed various types of faults like stuck-at fault, bridging fault, open fault, etc [1] [2] [10] [11]. However, they only address the fault at the interconnection level, none of them is able to achieve a full diagnosis for the faults inside the gate. In this paper we propose a diagnosis method for intra-gate stuck-open faults for digital circuits.

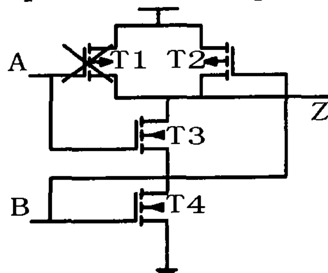


Figure 1. Transistor A stuck open

The stuck-open model assumes a particular transistor is permanently turned off (see Figure 1, transistor T1 is turned off) as a result, for example, of a failed transistor or connection break [3] [5] [6] [10]. Stuck-open faults can cause sequential behaviour and thus require a certain sequence of patterns in order to be detected. Much work has been done in respect to stuck-open test pattern generation. Some authors have investigated the stuck-open fault coverage when a set of stuck-at test patterns is applied [4] [5]. Abd-El-Barr [7] has proposed a deductive method to generate robust stuck-open test patterns. Testing of open defects in memory address decoders is addressed in [9]. Most of the work has focused on the detection of stuck-open faults but with little attention to diagnosis.

So far, the only systematic published work of stuck-open diagnosis is from Li [6] [8] [10]. As an essential pre-processing, Li's work uses a stuck-at fault diagnosis tool to pick out the likely candidates of stuck-open fault before he carries out his own stuck-open diagnosis. In Section 2, we examine the relationship between stuck-open and stuck-at faults and introduce the diagnosis terminology defined by Hora [2] and used in Li's work. In Section 3, we present the overall flow of our method. Li's method [8] has to build up an excitation table for every type of gate manually, which we think is not efficient enough. To improve Li's method in an automatic way, we develop a transformation method in Section 4. Experimental results are shown in Section 5. We concentrate on CMOS technology only.

2. The relationship between stuck-open faults and stuck-at faults

To detect a stuck-open fault, a sequence of two patterns needs to be applied. Consider a NAND gate with a stuck-open fault in p-transistor T1 (Figure 1). Table 1 shows all possible patterns that we could apply (including the sequential effects - the previous value of Z in column 2). Column 5 is the expected fault free output (Z), column 6 is what we observed output (Obs) by performing electrical simulation when T1 is stuck open,

and column 7 and column 8 are the fault signatures of Z stuck-at 0 and A stuck-at 1, respectively.

Table 1. The behaviour of different faults in Figure 1

Pattern#	PreZ	A	B	ExpZ	ObsZ	Z-sa0	A- sa1
1	0	0	0	1	1	0	1
2	0	0	1	1	0	0	0
3	0	1	0	1	1	0	1
4	0	1	1	0	0	0	0
5	1	0	0	1	1	0	1
6	1	0	1	1	1	0	0
7	1	1	0	1	1	0	1
8	1	1	1	0	0	0	0

We define two sets – Obs and Sim. The set Obs (Observed Results) represents the pairs of pattern/failed pin observed on the tester, and the set of Sim (Simulation Results) represents the pairs predicted by the fault simulation. In previous work on stuck-at diagnosis by Hora [2], two measurements have been used to judge the quality of the diagnosis. The term "Matching" is used to quantify the extent to which the observed failing results actually match with the simulation of a particular fault.

$$\text{Matching (M)} = \frac{|Obs \cap Sim|}{|Obs|} \times 100\,\%$$

A second term "Prediction" is used to quantify the extent to which simulations of a particular fault predict only those failing observed results.

$$\text{Prediction (P)} = \frac{|Obs \cap Sim|}{|Sim|} \times 100\,\%$$

Suppose a stuck-open occurs in T1 in Figure 1, from Table 1 we know by simulation that *Obs* is {2/Z} (pattern number/failed pin). If Z stuck-at-0 is simulated, the *Sim* will be {1/Z, 2/Z, 3/Z, 5/Z, 6/Z, 7/Z}, and if A stuck-at-1 is simulated, the *Sim* will be {2/Z, 6/Z}.

For Z stuck-at-0,
Matching (M) = |{2/Z}| / |{2/Z}| = 1/1 = 100%
Prediction (P) = |{2/Z}| / |{1/Z, 2/Z, 3/Z, 5/Z, 6/Z, 7/Z}| = 1/6 =17%

And for A stuck-at-1,
Matching (M) = |{2/Z}| / |{2/Z}| = 1/1 = 100%
Prediction (P) = |{2/Z}| / |{2/Z, 6/Z}| = 1/2 =50%

As we can see, when a stuck-open fault is present inside a gate, a stuck-at fault in the output (either stuck-at-0 or stuck-at-1) will always give Matching = 100% and Prediction ≤ 100%, and when the stuck-open transistor has a direct input (in Figure 1, A is the direct input of transistor T1), the stuck-at fault at this direct input (either stuck-at-0 or stuck-at-1) will give Matching = 100% and Prediction ≤ 100%.

With an open in a p-transistor, the gate output cannot be pulled up to a logic one via that transistor. If the output was previously a zero, it will stay at a zero and appear as a stuck-at fault at the output. But if the previous output was a one, or the output can be pulled up to a logic one via some other transistors, the output will be correct. It therefore follows that the *Obs* of a stuck-open will always be a subset of, or same as, the *Sim* of a stuck-at fault at the output. As a result, in the previous example we have Matching = 100% and Prediction = 17% for Z stuck-at 0.

When a stuck-open p-transistor has a direct input (as A for T1 in Figure 1), the A stuck-at-1 fault has the same effect of turning off T1. On top of that, it also helps to turn off other pull up paths and open the pull down paths. So, regardless of the previous output value, when a pattern is due to fail under T1 stuck-open model, it must fail under the A stuck-at-1 model. It therefore follows that the *Obs* of a stuck-open will always be a subset of, or same as, the *Sim* of a stuck-at fault at the direct input. As a result, we have Matching = 100% and Prediction = 50% for A stuck-at-1 in the previous example.

In general, when a stuck-open fault occurs, and we use stuck-at model to diagnose the circuit, both the output of the faulty logic gate and the direct input of the stuck-open transistor will report Matching = 100%. The relationship of their Predictions is: Prediction (output) ≤ Prediction (direct input) ≤ 100%

3. Overall flow of stuck-open diagnosis

The only systematic published work of stuck-open diagnosis is from Li [6] [8] [10]. Li's work includes an excitation table to judge which of the signatures in stuck-at *Sim* will remain as stuck-open signatures. This excitation table is constructed manually on an individual gate basis, and every transistor will have its own excitation table. This could take a considerable effort for a large scale design. It is not worthwhile to do this especially when we have so many diagnosis tools for stuck-at faults. Figure 2 shows the overall flow of our method.

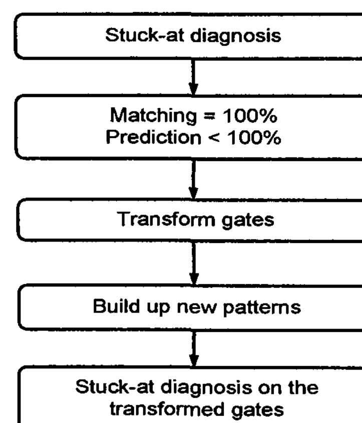


Figure 2. Overall flow of our method

We performed an initial single stuck-at fault diagnosis in the first step. From the stuck-at diagnosis, we select those gates with either input or output diagnosed as Matching = 100% and Prediction < 100%.

Instead of constructing an excitation table for each stuck-open fault, we transform the gates to a new form where all the stuck-open faults are represented by stuck-at faults. This transformation method will be introduced in the next section. After the transformation, new patterns are constructed from both the previous and current input values of the transformed gates. For every stuck-at pattern that propagates the target stuck-at fault to the output pins, we find out its previous pattern. The two consecutive patterns that appear on the transformed gate are concatenated to form a new pattern. Finally, the stuck-at diagnosis is performed once more but this time only on the transformed gates. If successful, the stuck-open faults will be diagnosed in the names of the corresponding stuck-at faults.

In doing so, any stuck-at fault diagnosis tool will be able to diagnose the stuck-open fault directly. This method is very straightforward and can be easily integrated into commercial diagnosis tools.

4. Transformation method

The basic idea of the transformation is to represent a stuck-open fault with a stuck-at fault so the stuck-at diagnosis tool can be used, instead of building a new stuck-open fault diagnosis tool. The following example explains how the transformation works.

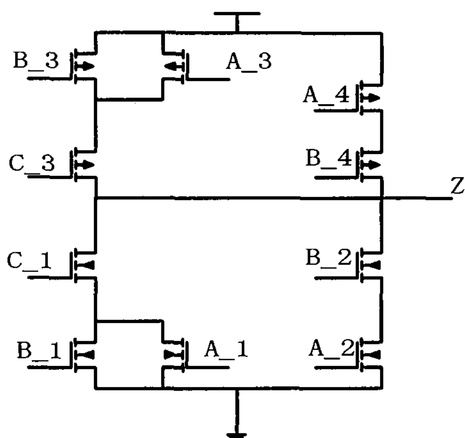


Figure 3. Schematics of the example circuit

Figure 3 is the schematic of an example logic gate with three inputs (*A B C*) and one output (*Z*). We split each input signal into a number of branches, one for each transistor the input is connected to. For the gate from our example, input *A* will be split in *A_1, A_2, A_3* and *A_4* (see Figure 3). We transform the transistor level schematic to the gate level by the following rules.

1. For n-transistors
(a) Replace all the n-transistors with the element as shown in Figure 4. We want to guarantee that a zero voltage from the source will be transmitted to the drain when value on the input (*A* in our example) is one (transistor turned-on). When the input *A* is one (transistor turned-on) and the source signal is zero, the drain will be zero. Otherwise, the drain will be one.

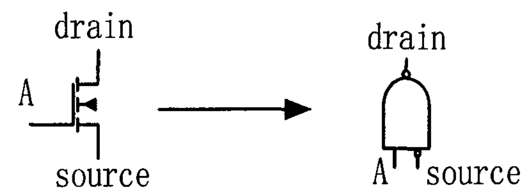


Figure 4. Replacement of n-transistor

(b) Place an AND gate where a parallel connection between transistors is present, as Figure 5. Here we also want to guarantee that the zero signals from the source will be transmitted to the drain as long as either one of the transistors is turned-on. Therefore the AND gate makes sure if one of the drains is zero, the output will be zero. If none of the drains is zero, the output will be one.

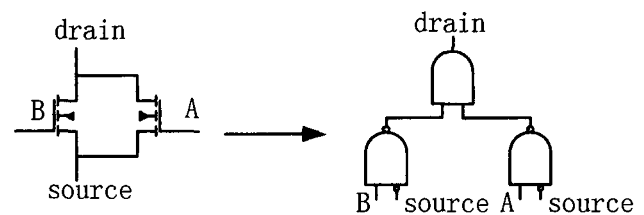


Figure 5. Replacement of parallel n-transistors

2. For p-transistors
(a) Replace all the p-transistors with the element as shown in Figure 6. The purpose is similar, to guarantee the propagation of logic one from the source.

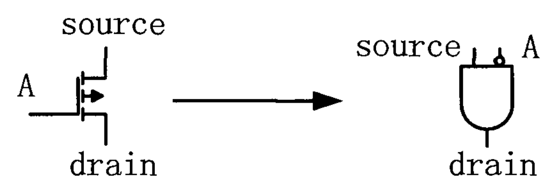


Figure 6. Replacement of p-transistor

(b) Place an OR gate where a parallel connection is present, as Figure 7.

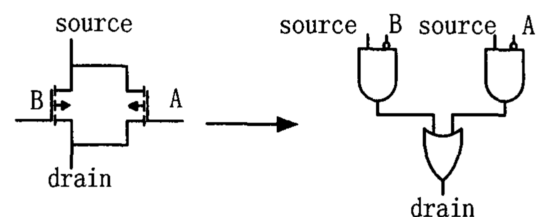


Figure 7. Replacement of parallel p-transistors

The view after the whole transformation of the gate shown in Figure 3 will be as Figure 8. As one can see, the whole graph consists of a *P* Part and an *N* Part. Because we are only interested in stuck-open faults, we assume the only possible faults will be stuck-at-0 on branch inputs for the *N* Part and stuck-at-1 on branch inputs for the *P* Part. For instance, for our example, a stuck-open fault at *C_1* in Figure 3 will be represented by a stuck-at-0 fault at *C_1* in Figure 8 and a stuck-open fault at *A_3* in Figure 3 will be represented by a stuck-at-1 fault at *A_3* in Figure 8.

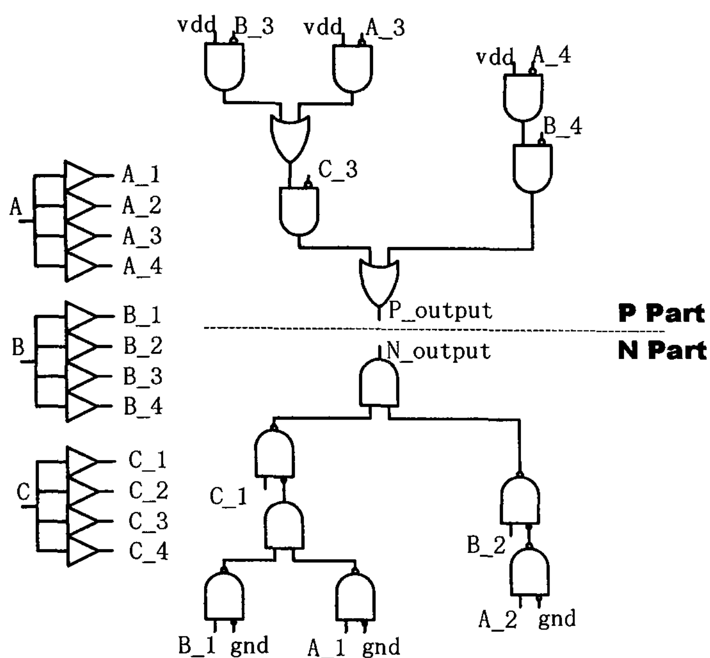


Figure 8. The first phase of transformation

Let us see how we represent the stuck-open faults. In the faulty free case, the P_output and N_output give the same responses. They both give the results that we would see at *Z* in Figure 3. Consider now the faulty case where there is a stuck-at fault in one of the input branches. To excite a stuck-open fault, we need two patterns, the first pattern sets the output to a certain value, the second pattern has to make sure that the target transistor plays an indispensable role to drive the output to the opposite value. Thus when the target transistor is stuck open, the output fails to go to the opposite value and keeps the previous one. From the structural point of view, this means the stuck-open transistor has to control the only serially conducting path.

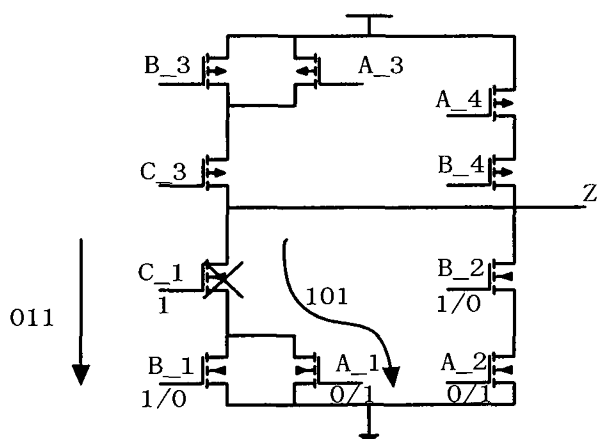


Figure 9. Stuck-open fault at C_1

For instance in Figure 9, to excite a stuck-open at *C_1*, the first pattern (for instance *ABC*=000) sets *Z* to one, the second pattern should be either *ABC*=011 or *ABC*=101. The expected pull-down paths for these two patterns are depicted by the arrows on Figure 9. In either case, *C_1* controls the only conducting path.

Let us see how Figure 8 represents the controls that *C* has over this scenario. The placement of AND and OR gates makes sure that the stuck-at fault propagates only when there is no parallel conducting path, in other words, the stuck-at fault controls the only conducting path. For instance in Figure 10, under either of the second patterns (*ABC*=011 and *ABC*=101) that excite the *C_1* stuck-open in Figure 9, the stuck-at-0 fault at *C_1* in Figure 10 is also propagated to N_output, as depicted by the arrows.

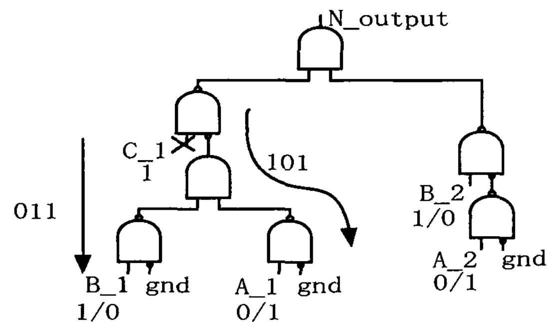


Figure 10. Stuck-at fault at C_1

Now, only when the stuck-open second pattern excitation requirement is met, will either P_output or N_output in Figure 8 fail.

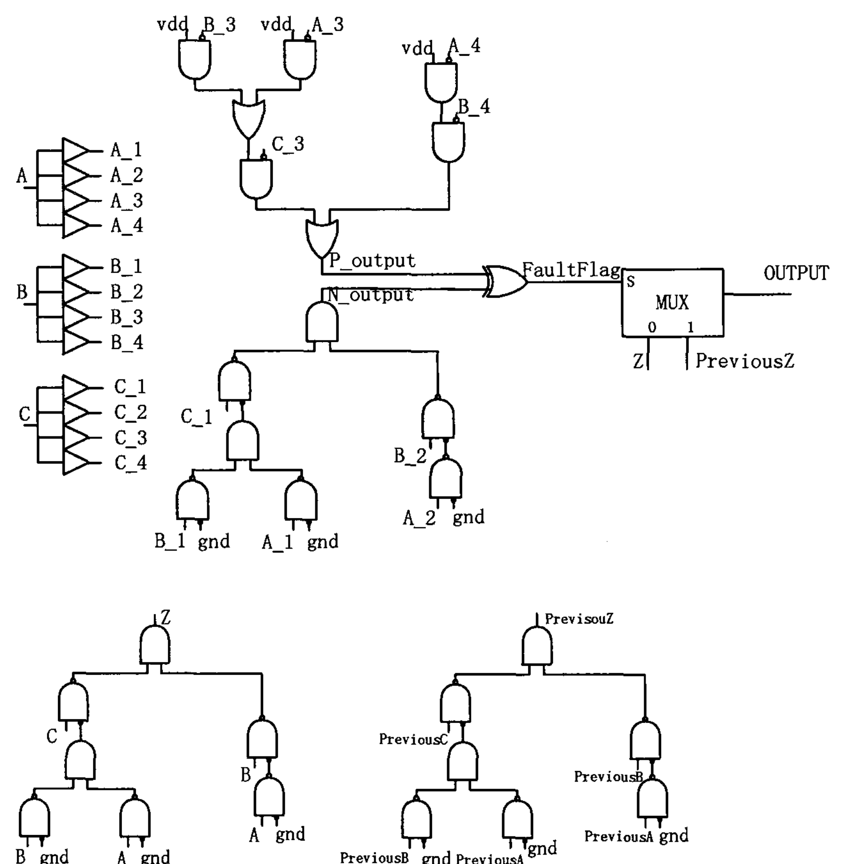


Figure 11. Final picture of transformation

However, not only does the second pattern excitation requirement need to be met, but the first pattern excitation requirement, (i.e. to set up the first pattern output value at the opposite of the second), also needs to be met before we can see a fail at the final output. Here comes the final transformation shown in Figure 11. Two sub-circuits are used to generate the fault free value of the current *Z* and the previous *Z*. The P_output and N_output are connected by an XOR gate to generate a FaultFlag signal, which will be zero when no fault happens and one otherwise.

When there is no stuck-at fault, P_output=N_output, and FaultFlag=0, the MUX selects the correct current *Z*

value, so the OUTPUT will not fail. When a stuck-at fault propagates (the second pattern requirement of stuck-open excitation is met), P_output≠N_output, and FaultFlag=1. The MUX then selects the previous Z value, which determines if the OUTPUT fails or not. If the previous Z value is different from the correct current Z value, the OUTPUT fails. If they are the same, the OUTPUT does not fail. By adding this extra MUX, the first pattern requirement of stuck-open excitation is also met. Now, all the stuck-open faults in the P part are represented by the corresponding stuck-at-1 faults and all the stuck-open faults in the N part are represented by the corresponding stuck-at-0 faults.

5. Experimental results

Experiments have been performed using a stuck-at diagnosis tool (FALOC, a Philips internal CAT tool). Based on our method, stuck-at faults with Matching = 100% and Prediction < 100% are first selected. For all the patterns that propagate the specific stuck-at fault to the output pins, we find out its previous pattern. These two patterns are then combined to form a new pattern. The transformed gate and the new patterns are then diagnosed by stuck-at model again to have the final stuck-open diagnosis. Here we assume that several gates are selected by the first round stuck-at diagnosis and their patterns are given. We have experimented with several gates, from simple one-block gates to gates with multiple blocks. The design and the parameters of the gates for electrical simulation are from Philips CMOS 0.18μm technology. The results show that all stuck-open faults are completely diagnosed unless they are affected by timing skew problem (we will explain this problem in section 5). We present the experimental procedure and the result of three of them. The gate in Figure 3 is one of the three and is named as Gate No.1. Figure 12 shows the schematics of the other two, Gate No.2 and a two block gate, Gate No.3.

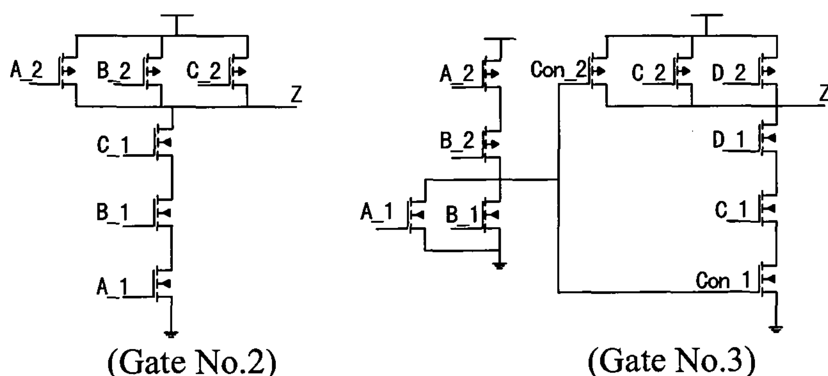


Figure 12. Experimental gates

Note we have separated the fan-out branches to each transistor to produce *A_1, A_2*, etc.

Table 2 shows the patterns applied. Gate No.1 and Gate No.2 are tested with 16 pairs of patterns, each includes a previous pattern (Pre) and a current pattern (Cur). Gate No.3, which has four input nets, is tested with 32 pairs of patterns. The current patterns we used include all the possible input combinations for every gate.

Table 2. Experimental patterns

Pattern#	Gate No.1 (ABC)		Gate No.2 (ABC)		Gate No.3 (ABCD)				
	Pre	Cur	Pre	Cur	Pre	Cur	Pattern#	Pre	Cur
1	111	000	100	000	1100	0000	17	0011	0000
2	111	001	100	001	1100	0001	18	0011	0001
3	111	010	100	010	1100	0010	19	0011	0010
4	111	011	000	011	1100	0011	20	0110	0011
5	111	100	000	100	1100	0100	21	0011	0100
6	111	101	100	101	1100	0101	22	0011	0101
7	111	110	100	110	1100	0110	23	0011	0110
8	110	111	100	111	1100	0111	24	0011	0111
9	110	000	111	000	1100	1000	25	0011	1000
10	000	001	111	001	1100	1001	26	0011	1001
11	000	010	111	010	1100	1010	27	0011	1010
12	000	011	111	011	1100	1011	28	0011	1011
13	000	100	111	100	0100	1100	29	0011	1100
14	000	101	111	101	1100	1101	30	0011	1101
15	000	110	111	110	1100	1110	31	0011	1110
16	000	111	001	111	1100	1111	32	0011	1111

The three gates are then simulated by a SPICE level simulation tool, subjected to the patterns in Table 2. Each stuck-open defect is injected into the transistor schematics and simulated. We assume the failed transistor has a hard open defect so we modeled it with a resistor of 1GΩ connected to a net of the failed transistor. The failed patterns are reported in Table 3.The failed patterns plus the transformed circuits of the three gates are fed into FALOC to be diagnosed under stuck-at model. Table 3 shows the result. For each gate, the first column indicates which transistors are supposed to be stuck open. The second column shows the failed patterns. The third column indicates whether the corresponding stuck-at faults are diagnosed as Matching = 100% and Prediction = 100%.

In fact, the corresponding stuck-at fault will be diagnosed together with other logically equivalent faults. For Gate a, when pattern 12 and 14 are set as the failed patterns feeding into FALOC, the report contains following faults diagnosed as Matching = 100% and Prediction = 100%: net=*C* SA0, net=*C_3* SA0, net=*C_1* SA0. As mentioned earlier, the only stuck-at faults considered as possible are stuck-at-0 (SA0) faults in the *N* Part and stuck-at-1 (SA1) faults in the *P* Part. In this particular case, that includes *A_1, B_1, C_1, A_2, B_2* for SA0 and *A_3, B_3, C_3, A_4, B_4* for SA1. As to this diagnosis report, the only qualified candidate is *C_1* for SA0, the rest are the redundant faults which can be removed. *C_1* stuck-open fault is exactly the right answer. From Table 3 we can see that all the stuck-open faults are diagnosed by the corresponding stuck-at faults, thus proving the effectiveness of our method.

6. Conlusions and future work

A method of transforming from transistor level schematic to logic gate level description has been proposed. By this method, a stuck-open fault in the

Table 3. Diagnosis results

Gate No.1			Gate No.2			Gate No.3		
Stuck-Open Transistor	Failed Patterns	Diagnosed Matching = 100% Prediction = 100%	Stuck-Open Transistor	Failed Patterns	Diagnosed Matching = 100% Prediction = 100%	Stuck-Open Transistor	Failed Patterns	Diagnosed Matching = 100% Prediction = 100%
A_1	14	A_1 sa-0	A_1	8,16	A_1 sa-0	A_1	28	A_1 sa-0
A_2	15	A_2 sa-0	A_2	12	A_2 sa-1	A_2	4,20	A_2 sa-1
A_3	3	A_3 sa-1	B_1	8,16	B_1 sa-0	B_1	24	B_1 sa-0
A_4	2	A_4 sa-1	B_2	14	B_2 sa-1	B_2	4,20	B_2 sa-1
B_1	12	B_1 sa-0	C_1	8,16	C_1 sa-0	C_1	4,20	C_1 sa-0
B_2	15	B_2 sa-0	C_2	16	C_2 sa-1	C_2	18	C_2 sa-1
B_3	5	B_3 sa-1				D_1	4,20	D_1 sa-0
B_4	2	B_4 sa-1				D_2	19	D_2 sa-1
C_1	12,14	C_1 sa-0				Con_1	4,20	Con_1 sa-0
C_3	3,5	C_3 sa-1				Con_2	24,28,32	Con_2 sa-1

transistor level can be represented by a stuck-at fault at the logic gate level. A commercial stuck-at diagnosis tool can be brought in to pick out the stuck-open fault directly without the need to build up excitation tables and rewrite a part of the diagnosis tool, which has been proposed previously [10]. Also, our method has little to do with the size of the circuit, it will remain the same as long as the cell designs are not changed. Experimental results show the correctness of our method.

There are two situations where the patterns may have influence on the final diagnosis result. One is the timing skew problem [8] [10] when there are two bits changing between the previous pattern and the current pattern. Different intermediate patterns are formed depending on the timing of the transition. (See Figure 13)

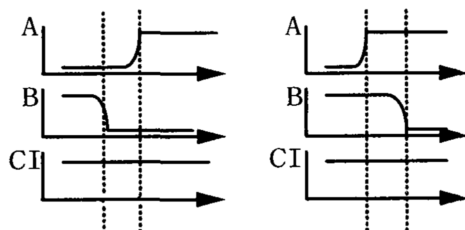


Figure 13. Timing skew

Another problem is that when the two patterns both cause high impedance Z. Whether or not the current pattern fails depends on the patterns even earlier. We are investigating the impact of these two situations and will report on them later.

7. Acknowledgement

We would like to thank Ananta Majhi, Mohamed Azimane, Maurice Lousberg of Philips Research and Stefan Eichenberger of Philips Semiconductors for their contributions to this work.

[1] S. Venkataraman and S. B. Drummonds, "POIROT: A Logic Fault Diagnosis Tool and Its Applications", *International Test Conference,*2000. pp.253-262.

[2] C. Hora, et al, "On Electrical Fault Diagnosis in Full-Scan Circuits", *International Workshop on Defect Based Testing,* 2001. pp. 17-22.

[3] S.M. Menon, et al, "Testable Design of BiCMOS Circuits for Stuck-Open Fault Detection using Single Patterns", *VLSI Test Symposium*, April 1993. pp. 296-302.

[4] B. W. Woodhall, et al, "Empirical Results on Undetected CMOS Stuck-Open Failures", *International Test Conference,* 1987. pp. 166-170.

[5] S. D. Millman and E. J. McCluskey "Detecting Stuck-Open Faults with Stuck-At Test Sets", *IEEE Custom Integrated Circuits Conference*, 1989. pp. 22.3.1-22.3.4.

[6] J. Li et al, "Testing for Resistive and Stuck Opens," *International Test Conference*, 2001. pp. 1049-1058.

[7] M. H. Abd-El-Barr, et al, "Transistor Stuck-Open Fault Detection in Multilevel CMOS Circuits", *Great Lakes Symposium on VLSI*, 1999. pp. 388-392.

[8] J. Li, "Test and Diagnosis of Open Defects in Digital CMOS IC," Ph.D. Dissertation, Stanford University, 2002.

[9] M. Sachdev, "Open Defects in CMOS RAM Address Decoders," *IEEE Design and Test of Computers*, vol. 14, no. 2, 1997. pp. 26-33

[10] J. Li et al, "Diagnosis for Sequence Dependent Chips", *VLSI Test Symposium*, 2002, pp.187-192.

[11] Y. Sato, et al, "A Persistent Diagnostic Technique for Unstable Defects", *International Test Conference,*2002. pp.242-249

A Gate-Level Method for Transistor-Level Bridging Fault Diagnosis

Xinyue Fan, Will Moore
Department of Engineering Science
Oxford University, OX1 3PJ
{xinyue.fan, will.moore}@eng.ox.ac.uk

Camelia Hora, Mario Konijnenburg, Guido Gronthoud
Philips Research Labs
Prof. Holstlaan 4, 5656AA Eindhoven, Netherlands
{camelia.hora, mario.konijnenburg, guido.gronthoud}@philips.com

Abstract

The paper addresses the issue of transistor-level bridging fault diagnosis. While most of the previous bridging fault diagnosis work focuses on the gate-level bridging faults, this method provides a solution to intra-gate bridging faults diagnosis for the first time. Instead of using any transistor level simulation tools, we develop a transformation technique that allows transistor-level bridging faults to be diagnosed by the commonly used gate-level bridging faults diagnosis tools. Real diagnosis results from Philips designs are presented.

1. Introduction

As the new IC manufacturing technology continues to offer higher integration density, manual search and identification of the defects becomes impractical because of the exceedingly high cost and the time it could take. Therefore automatic fault diagnosis is needed in order to provide a short-list of candidate defects for precise failure analysis.

Bridging faults are among the most commonly seen defect types in IC manufacturing [1]. Inter-gate bridging faults have been extensively studied and methods for their diagnosis have been developed [2] [3] [4] [7] [8]. The basic approach used is to build up composite bridging fault signatures from stuck-at fault signatures by assuming a certain bridging fault model. The effect of the electrical shorts has been assumed equal to an AND gate or an OR gate, known as the wired-AND and wired-OR bridging fault model. Another straightforward bridging fault model is the dominant model, in which one of the shorted lines always drives the value of the other. More accurate models like the voting model and the bias voting model are proposed in [12] [13]. These bridging fault models focus on the logical behaviour of the fault and thus are easy to implement in simulation, test pattern generation and diagnosis tools. The other main approach for bridging fault diagnosis is the application of Iddq testing, which makes use of the increased quiescent current generated when the component nodes of the bridge are driven to opposite values [5] [6].

All these methods are implemented at the inter-gate level. However, since complex gates are often used in modern CMOS design, the chances of a bridging fault occuring as an intra-gate fault are high. The gate-level diagnosis tools are unable to handle this type of fault as the bridged intra-gate nodes are not represented in the netlist. Up until now, no systematic diagnosis method has been given to tackle intra-gate bridging faults. In this paper, we introduce a method that can migrate the commonly used inter-gate level bridging fault model to the intra-gate level, thus enabling a commercial bridging fault diagnosis tool to diagnose the bridging faults inside gates.

The paper is organized as follows. Section 2 discusses the way that suspected intra-gate bridging gates are short-listed. In section 3, a new transformation method is proposed and the rules to represent the intra-gate bridging faults in the inter-gate level are discussed. Section 4 is the overall flow of our diagnosis method. Experimental results and analysis are given in Section 5.

2. Shortlist the possible gates with intra-gate bridging faults

Before we introduce the new method, some terminologies and the basic diagnosis flow should be made clear. The diagnosis flow is literally a process of comparing what we see on the tester and what we have predicted by simulation of the assumed fault.

Here we define two sets – *Obs* and *Sim*. The set *Obs* (Observed Results) represents the pairs of pattern/failed pin observed on the tester, and the set of *Sim* (Simulation Results) represents the pairs predicted by the fault simulation.

There are two different versions of the diagnostic measures used to determine the quality of the diagnosis [3] [9], but both involve ranking candidate faults by their values of *Obs* and *Sim*. In this paper, we use the system defined by Hora [3] who uses two measures to judge the quality of the diagnosis. The term "Matching" quantifies

the extent to which the observed failing results actually match with the simulation of a particular fault.

$$\text{Matching (M)} = \frac{|\,Obs \cap Sim\,|}{|\,Obs\,|} \times 100\,\%$$

A second term "Prediction" quantifies the extent to which simulations of a particular fault predict *only* those failing observed results.

$$\text{Prediction (P)} = \frac{|\,Obs \cap Sim\,|}{|\,Sim\,|} \times 100\,\%$$

Because we cannot realistically make an intra-gate bridging diagnosis for every gate in a large circuit, a shortlist of possible faulty gates, in which an intra-gate bridging fault may be present, is generated by using a net diagnosis model.

The net diagnosis model, initially developed for interconnect open defects [10], combines both stuck-at-0 and stuck-at-1 signatures of a net, thus guaranteeing that the output of the faulty gate will have Matching=100%. Consider the circuit in Figure 1 and the patterns shown in Table 1. We assume a net diagnosis model on the output line Z.

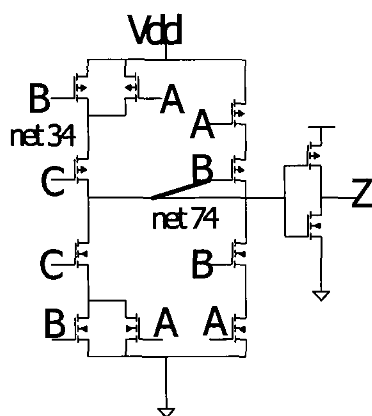


Figure 1. Sample gate

Pattern	ABC	Z
#1	000	0
#2	001	0
#3	010	0
#4	011	1
#5	100	0
#6	101	1
#7	110	1
#8	111	1

Table 1. Sample patterns

The Z stuck-at-1 signature is {#1 (Z fails at pattern #1), #2, #3, #5} and the Z stuck-at-0 signature is {#4, #6, #7, #8}, so the net signatures of Z is *Sim* = {#1, #2, #3, #4, #5, #6, #7, #8}. Suppose the resistance of the bridging fault is 1 ohm between B and net74, electrical simulation is performed to generate the observed result, *Obs* = {#1, #2, #4, #5, #7}. According to the definition of Matching and Prediction:

$$\text{Matching(M)} = \frac{|\,Obs \cap Sim\,|}{|\,Obs\,|} \times 100\% =$$

$$\frac{|\,\{\#1,\#2,\#4,\#5,\#7\} \cap \{\#1,\#2,\#3,\#4,\#5,\#6,\#7,\#8\}\,|}{|\,\{\#1,\#2,\#4,\#5,\#7\}\,|} \times 100\% = 100\%$$

$$\text{Prediction(P)} = \frac{|\,Obs \cap Sim\,|}{|\,Sim\,|} \times 100\% =$$

$$\frac{|\,\{\#1,\#2,\#4,\#5,\#7\} \cap \{\#1,\#2,\#3,\#4,\#5,\#6,\#7,\#8\}\,|}{|\,\{\#1,\#2,\#3,\#4,\#5,\#6,\#7,\#8\}\,|} \times 100\% = 63\%$$

Clearly the output net will always have Matching=100% under the net diagnosis model even though the fault is actually inside the gate. However we expect a low Prediction since no real defect will cause the output net both stuck-at 0 and stuck-at 1.

The diagnosis tools working under net diagnosis model will pick out those nets with Matching = 100% but Prediction < 100%, and our method will further investigate these gates whose output nets have been picked out. If there is real intra-gate bridging fault which matches with one of the models we used, the result of our method will be both Matching=100% and Prediction=100%, the most convincing diagnosis result.

Some of the intra-gate bridging faults will only see fault signatures either from output stuck-at-1 or from output stuck-at-0 appearing in its *Obs*. For instance in Figure 1, when net34 is bridged with Vdd (the fault can be also modeled as net34 stuck-at-1), we would only see sometimes net74 stuck-at-1 and, consequently, Z stuck-at-0. So, for those faults, the faulty gates will be diagnosed under stuck-at model as Matching = 100%, Prediction < 100%.

Therefore, to cover all the intra-gate bridging faults, those gates with output diagnosed as Matching = 100%, Prediction < 100% under either net diagnosis model or stuck-at model should be taken as our primary intra-gate bridging fault candidates.

3. Transform the transistor schematics

Having short-listed the primary candidate gates, we need to consider how to confirm if the gates contain intra-gate bridging faults as well as which bridging faults. The most direct solution would be to use electrical level simulation to predict the fault signatures (*Sim*) of every possible intra-gate bridging pairs and then compare them with the *Obs* we extract on the gate output. However, the number of all the possible intra-gate bridging pairs and the numerous patterns we have to simulate means time may become an issue. Also, for every simulation, we have to assume a certain resistance value which does not guarantee that the simulation result will be the same as for the real defect. In addition, the whole diagnosis software has to be rewritten.

We believe the traditional bridging fault diagnosis tools should not be restricted in the gate-level and we can extend its capability to the intra-gate domain. To use the gate-level bridging fault model for a transistor-level circuit, we have to transform the transistor-level

description to the gate-level while having all the potential bridging nets represented. (Some of the fault simulation tools have the transistor-level description, however the transistor primaries are normally unidirectional, thus they are not able to represent the values of the internal nodes.) The transformed gate can then be directly diagnosed with the traditional bridging fault diagnosis tools.

The gate-level bridging diagnosis tools analyze the values of possible bridging nets under each pattern. If the values between two possible bridging nets are different, then certain stuck-at fault signatures, depending on the bridging model used, are taken as the bridging fault signatures [2]. To fit in with this process, during the course of transformation, we follow two principles:

1) Every net (for instance in Figure 2: A, B, C, D, E, net1, net2, net3, net4, net5, Z) in the transistor-level schematics will be represented in the transformed circuits, and their values at every test pattern will remain the same as those before the transformation.
2) If a stuck-at effect propagates to the output i.e. it is not blocked by any off transistors, the corresponding stuck-at fault in the transformed circuit will also propagate.

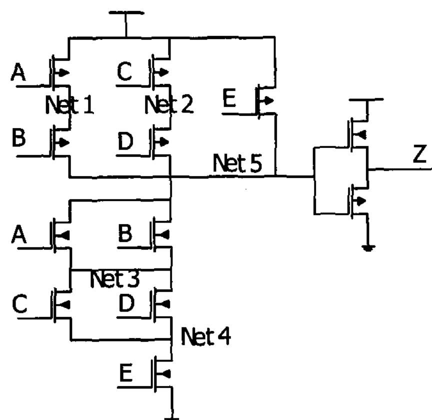


Figure 2. *The gate to be transformed*

Now we demonstrate the transformation method through the example gate in Figure 2. There are five internal nets in this particular gate (Net1-5). According to our first principle, all the five internal nets must be represented in a gate-level description where their values are kept. The following rules are set to justify the values of the internal nets.

1. For n-transistors part (Net3 and Net4), because the transistors are bi-directional, we need to do two rounds of justification, one from the GND and one from the net connecting the n-transistors and p-transistors, in this case Net5. In each round of justification, we have the following steps.

(a) Replace all the n-transistors with the element as shown in Figure 3. The purpose is to guarantee that the zero value from the source will be transmitted to the drain when value on the gate is one (transistor turned-on).

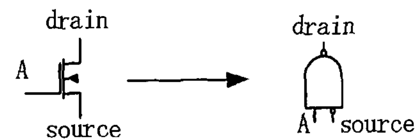


Figure 3. *Replacement of n-transistor*

(b) Replace with an AND gate where a parallel connection between transistors is present, as Figure 4. The AND gate makes sure if one of the drains is zero, the output will be zero, thus guaranteeing the propagation of zero value from the source.

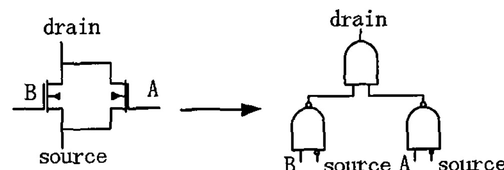


Figure 4. *Replacement of parallel n-transistors*

Figure 5 shows the view after two rounds of justifications. We also need to generate Net5_c, the correct value of net5, by logic combinations, as it will be later used to justify the internal nets. The rules to generate Net5_c are same as the rules we used in the first round of justification (the one from the GND). We use a similar approach to generate the correct values for the junction nets of p-transistors and n-transistors in [11], where a detailed explanation can be found.

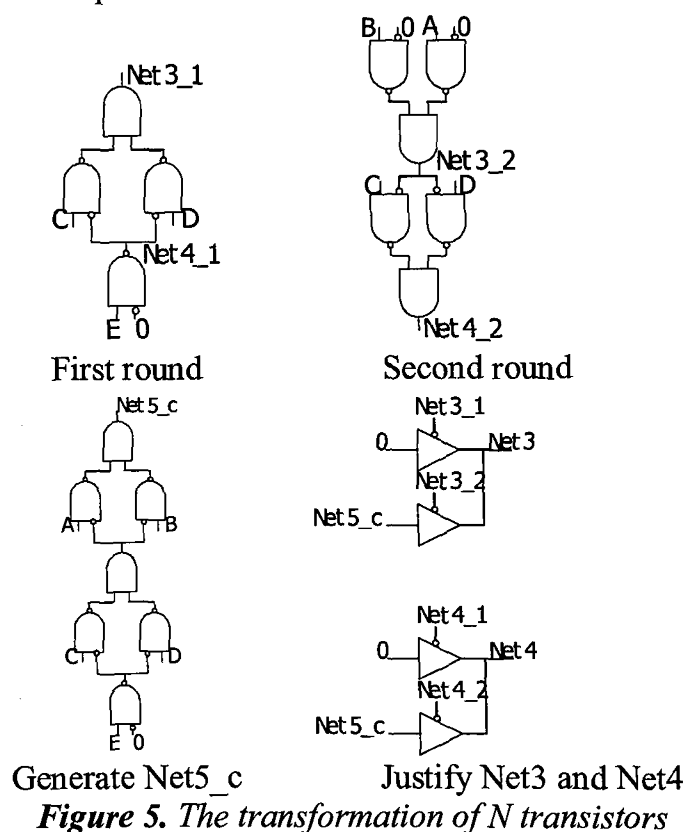


Figure 5. *The transformation of N transistors*

Net3_1 and Net4_1 in the first round justification respectively indicate whether Net3 and Net4 are electrically connected to the GND. A zero value shows it is connected, a one value shows it is disconnected.

Likewise, to judge if Net3 and Net4 are electrically connected to Net5, we have to look at the value of Net3_2 and Net4_2. These four nets with underscore function as the connection indicators.

The final justification for Net3 and Net4 is done by the four tri-state bus drivers, each controlled by a connection indicator and each individually decides if the value of GND or Net5 should be loaded to the Net3 and Net4. Up to this stage, Net3 and Net4 are fully justified.

2. For the p-transistor part (Net1 and Net2), we also have to perform two rounds of justification, one from the Vdd and one from Net5. The rules of transformation are similar to those for the n-transistor part, except when replacing the p-transistor, we use the element in Figure 6 instead of the one in Figure 3.

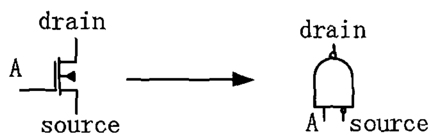


Figure 6. *Replacement of p-transistor*

Figure 7 shows the transformation of P transistors to justify the internal nets, Net1 and Net2. Note that there is no need to transform the transistor that directly connects either Vdd or GND to Net5, like the p-transistor control by E, for there is no internal net.

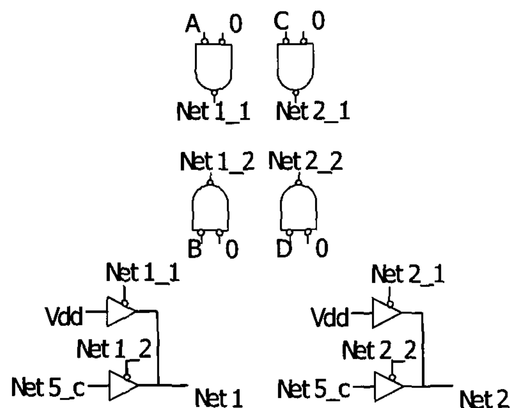


Figure 7. *The transformation of P transistors*

Now the values of internal nets have been fully justified. Next, as required by our second principle, we have to connect these internal nets in a way that all the stuck-at signatures are propagated when they are not completely blocked by the off transistors. Because we already have the connection indicators of every internal net (Net1_2, Net2_2, Net3_2, Net4_2) to Net5, the only thing we need to do is to add tri-state bus drivers controlled by the connection indicators, whose values decide if a stuck-at signature would be propagated to Net5 and subsequently to the output Z. Figure 8 is the whole view after the final transformation. Transistors like the p-transistor controlled by E in Figure 2, which directly connect Vdd or GND to the p-n junction net, have to be replaced by tri-state bus drivers to help Net1-4 to justify the value of Net5. Note that we have added a tri-state bus driver controlled by E.

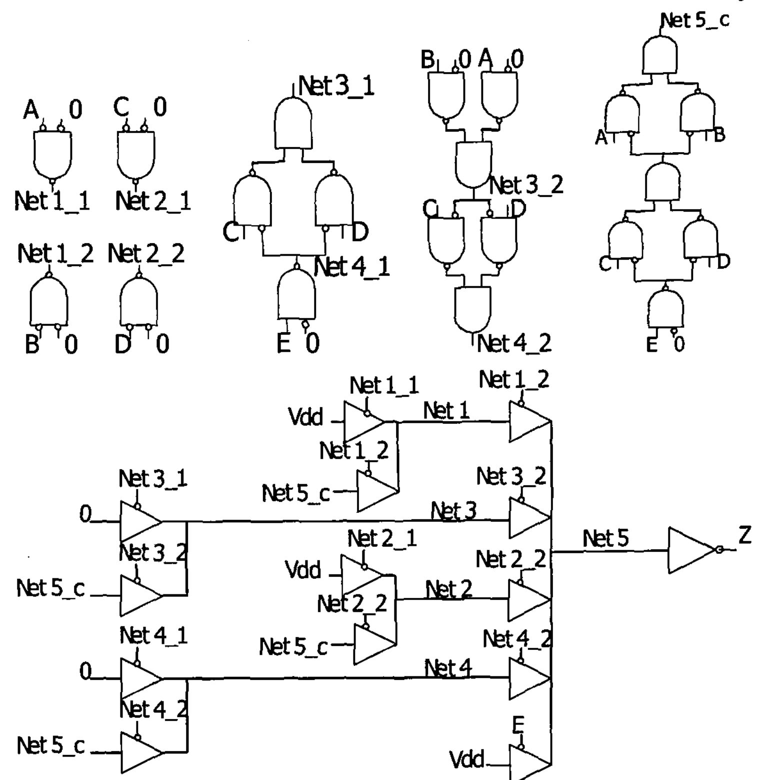


Figure 8. *Final transformation*

Now the two principles are met after the final transformation, the values of Net1-5 are kept and the stuck-at signatures of each internal net are duly propagated when condition allows.

4. Overall flow of diagnosis

The diagnosis flow can be summarized as four steps illustrated in Figure 9. First, a preliminary stuck-at fault diagnosis and failing net diagnosis is performed to shortlist the possible faulty gates with intra-gate bridging. Those gates whose outputs are diagnosed as Matching = 100% and Prediction < 100% under either stuck-at model or net diagnosis model are the primary suspects. The second step is to transform these gates according to the rules set in Section 3. (If desired, transformations can be precomputed for every cell in the library.) The purpose is to use the gate-level bridging diagnosis tools to diagnose possible bridging pairs in the transformed gates. But to achieve that, we do not have to perform the bridging diagnosis on the whole circuit. Instead, in the third step, for those patterns that propagate the failure signatures of the gate's output, we extract the input values of this particular gate and compose them into new patterns for the transformed gate. In the final step, the gate-level bridging diagnosis is applied on this transformed gate only, which takes very little time.

We do not yet have a tool to extract realistic intra-gate bridging pairs from the gate layout, though it would be useful in order to enhance the diagnostic resolution. Currently all pairs of two nets are set as possible intra-

gate bridging faults when we performed the final gate-level bridging diagnosis.

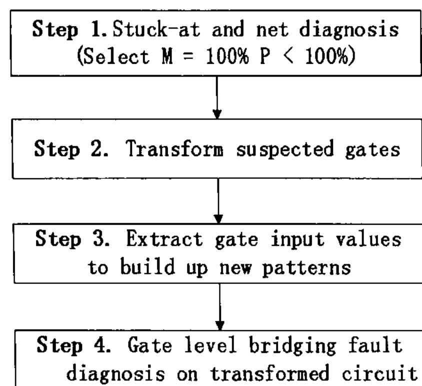


Figure 9. *Overall flow*

5. Experimental results

Experiments are performed with a Philips internal diagnosis tool - FALOC, which has the capacity of stuck-at fault, net fault and bridging fault diagnosis. The wafer testing data are from three different Philips' designs. We have shown seven successfully diagnosed intra-gate bridging faults in Table 2. The second and the third column are the type of the faulty gate and the intra-gate bridging fault it has. The fourth column shows the Matching and Prediction results of this particular gate when the first round diagnosis is performed (Step 1). After the transformation, the second round diagnosis is performed and the results are given in column five. The last column shows the number of intra-gate bridging faults that have been diagnosed as Matching = 100%, Prediction = 100%. Sometimes the number is more than one, therefore we have to refer to the gate layout to judge if one of them is a realistic bridging fault.

For all the seven faulty dies, we are able to enhance the Prediction to 100% after the gate is transformed. Layout information showed that just one of these is a likely site for a bridge (Die #4, Die #5 and Die #7). For those singly diagnosed bridging fault (Die #1, Die #2, Die #3, Die #6), a likely layout explanation was also found. (This step would be unnecessary should a layout extraction tool for intra-gate bridging faults be available before the second round diagnosis.)

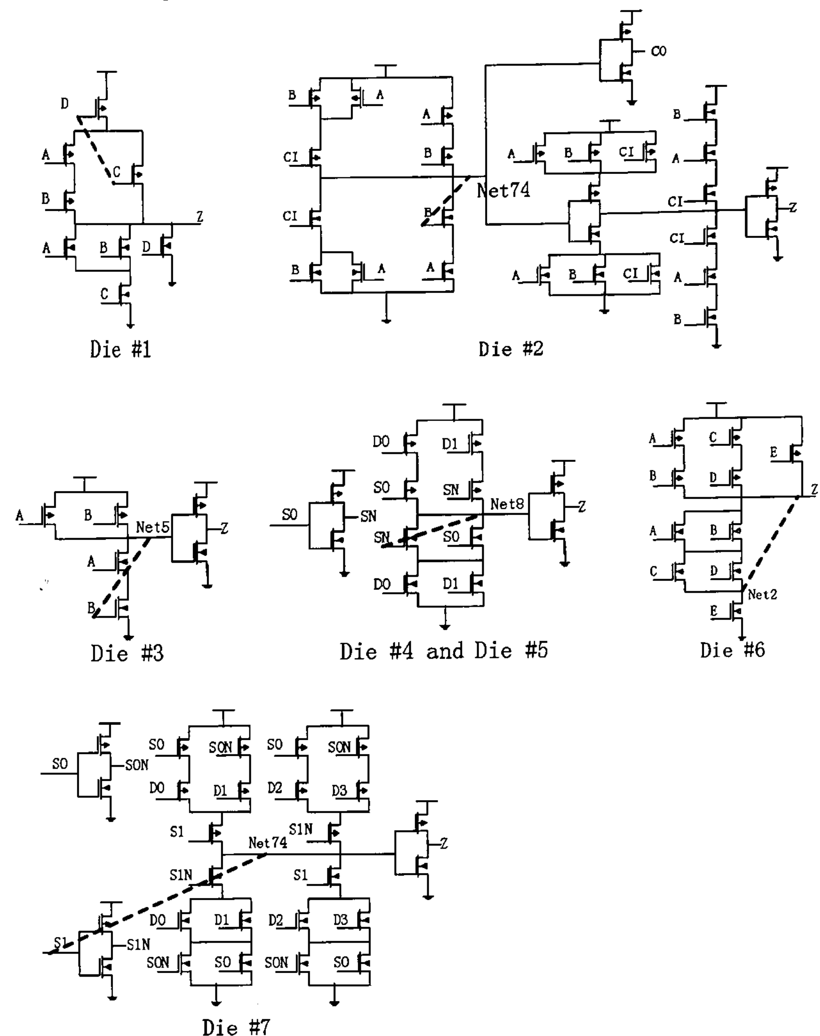


Figure 10. *Seven intra-gate bridging faults*

To further prove our diagnosis, PSPICE-like transistor-level simulations have been performed for all the seven intra-gate bridging faults, and the simulation results confirm the modeled behaviours of all the bridging faults. For Die #1, Die #2 and Die #3, we have been able to get the inline inspection information. Our diagnosis points to bridging faults between C and D (Figure 11, Die1#.a), B and Net74 (Figure 11, Die#2.a), B and Net5 (Figure 11, Die#3.a). The inline inspection pictures in Figure 11.b reveal bridging faults that match exactly with our diagnosis results.

Die	Gate Type	Bridged Nets	M (Matching) and P (Prediction) before transformation	M (Matching) and P (Prediction) after transformation	Number of MP = 100% faults
#1	ao32	C and D	Z stuck-at-1 M = 100% P = 63%	C D wired-AND M = 100% P = 100%	1
#2	fa1	B and Net74	CO net model M = 100% P = 61%	B dominates Net74 M = 100% P = 100%	1
#3	an2	B and Net5	Z net model M = 100% P = 42%	B Net5 wired -AND M = 100% P = 100%	1
#4	mx21	SN and Net8	Z net model M = 100% P = 48%	SN dominates Net8 M = 100% P = 100%	2
#5	mx21	SN and Net8	Z net model M = 100% P = 54%	SN dominates Net8 M = 100% P = 100%	2
#6	ao36	Net2 and Z	Z stuck-at-0 M = 100% P = 3%	Net2 dominates Z M = 100% P = 100%	1
#7	mx41x4	S1 and Net74	Z net model M = 100% P = 33%	S1 dominates Net74 M = 100% P = 100%	2
#7	mx41x4	S1 and Net74	Z net model M = 100% P = 33%	S1 dominates Net74 M = 100% P = 100%	2

Table 2. *Successful diagnosis*

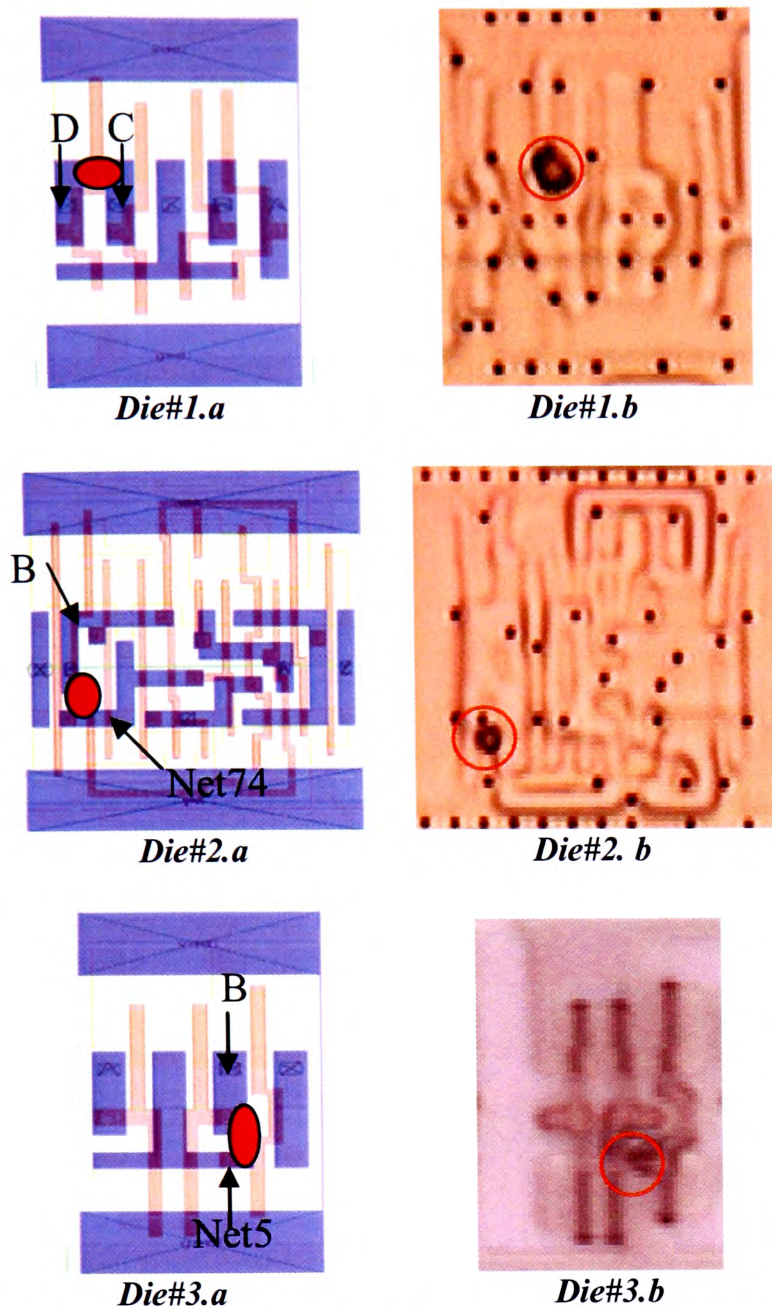


Figure 11. *Failure analysis of Die #1, Die #2, Die #3*

Unfortunately, the other four dies are no longer available for the physical failure analysis. Nonetheless, the diagnosis results have been confirmed by transistor-level simulations.

6. Conclusion

A transformation method is introduced, which allows gate-level bridging diagnosis tools to diagnose intra-gate bridging fault. The method costs little extra time on top of the initial stuck-at and net model diagnosis because the extra steps are performed on the suspected gates only. Gate layout information is used to prune out the unrealistic intra-gate bridging faults. The seven successful diagnosis results, supported by electrical simulation and strengthened by three inline inspection results, prove the effectiveness of this method. This new work, when put together with our previous work on diagnosis of intra-gate stuck-opens [11], provides a powerful extension of our ability to diagnose real defects.

7. Acknowledgement

The authors would like to thank Ananta Majhi, Mohamed Azimane, Maurice Lousberg of Philips Research and Stefan Eichenberger of Philips Semiconductors for their much appreciated help to this work.

[1] F.J. Ferguson and J.P. Shen, "A CMOS fault extractor for Inductive Fault Analysis". *IEEE Transactions on Computer-Aided Design*, Vol 7 (11), 1988, pp. 1181-1194.

[2] D. Lavo et al, "Bridging Fault Diagnosis in the Absence of Physical Information", *International Test Conference,* 1997, pp. 887-893.

[3] C. Hora "On Diagnosing Faults in Digital Circuits", PhD Thesis, Technique University Eindhoven, 2002.

[4] S.D Millman et al, "Diagnosing CMOS bridging faults with stuck-at fault dictionaries", *International Test Conference,* 1990, pp. 860-870.

[5] S. Chakravarty, s. Suresh, "IDDQ measurement based diagnosis of bridging faults in full scan circuits", *Proceedings of the Seventh International Conference on VLSI Design,* 1994 pp.179 – 182.

[6] C. Thibeault, "On the adaptation of Viterbi algorithm for diagnosis of multiple bridging faults", *IEEE Transactions on Computers*, Vol 49, Issue 6, June 2000 pp.575 – 587.

[7] B. Chess et al, "Diagnosis of realistic bridging faults with single stuck-at information", *IEEE/ACM International Conference on Computer-Aided Design*, 1995, pp.185 – 192.

[8] J. Wu, E.M. Rudnick, "Bridge fault diagnosis using stuck-at fault simulation", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol 19, Issue 4, April 2000, pp.489 – 495.

[9] S. Venkataraman and S. B. Drummonds, "POIROT: A Logic Fault Diagnosis Tool and Its Applications", *International Test Conference*, 2000. pp. 253-262.

[10] S. Venkataraman and S. B. Drummonds, "A technique for logic fault diagnosis of interconnect open defects", *IEEE VLSI Test Symposium*, 2000.pp. 313 – 318.

[11] X. Fan et al, "A Novel Stuck-At Based Method for Transistor Stuck-Open Fault Diagnosis", *International Test Conference*, 2005, Accepted.

[12] J. M. Acken and S. D. Millman, "Accurate Modelling and Simulation of Bridging Faults", *Custom Integrated Circuit Conference,* 1991. pp. 63-72

[13] P. C. Maxwell and R. C. Aitken, "Biased Voting: A Method for Simulating CMOS Bridging Faults in the Presence of Variable Gate Logic Thresholds", *International Test Conference*, 1993, pp.63-72