On diagnosing faults in digital circuits

Hora, S. C.

DOI:
10.6100/IR559933

Published: 01/01/2002

Citation for published version (APA):
On Diagnosing Faults in Digital Circuits

Camelia Hora
On Diagnosing Faults
in
Digital Circuits

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
Rector Magnificus, prof.dr. R.A. van Santen, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op maandag 25 november 2002 om 16.00 uur

door

Stefania Camelia Hora

geboren te Oradea, Roemenië
Dit proefschrift is goedgekeurd door de promotoren:

prof.ir. M.T.M. Segers
en
prof.ir. M.P.J. Stevens

©Copyright 2002 Camelia Hora
All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written permission from the copyright owner

Druk: Universiteitsdrukkerij Technische Universiteit Eindhoven

CIP-DATA LIBRARY TECHNISCHE UNIVERSITEIT EINDHOVEN

Hora, Stefania Camelia

Proefschrift.-ISBN 90-386-1990-1
NUR 959
Trefw.: foutendiagnose/logische schakelingen; faultdetectie/
productieoptimalisering/CMOS-schakelingen.
Subject headings: fault diagnosis/ failure analysis/integrated circuit yield/
CMOS logic circuits.
To my mother
Contents

1 Introduction .......................... 1
   1.1 Introduction .......................... 1
   1.2 Thesis outline .......................... 2

2 Failure analysis ....................... 5
   2.1 Introduction .......................... 5
   2.2 Electrical characterization .......... 6
   2.3 Fault localization. Fault diagnosis ......... 10
      2.3.1 Physical fault diagnosis .......... 10
      2.3.2 Electrical fault diagnosis .......... 17
   2.4 Deprocessing .......................... 19
   2.5 Defect localization .................... 20
   2.6 Inspection and physical characterization .. 21
   2.7 Conclusions and discussion ............ 24

3 Electrical fault diagnosis ............ 25
   3.1 Introduction .......................... 25
   3.2 Diagnosis and fault models ............. 25
      3.2.1 The stuck-at fault model .......... 25
      3.2.2 Bridging fault models ............ 28
      3.2.3 The open fault model ............. 32
      3.2.4 The delay fault model ............ 34
   3.3 Stuck-at fault diagnosis ............... 37
   3.4 Bridging fault diagnosis ............... 39
   3.5 Open fault diagnosis .................. 43
   3.6 Delay fault diagnosis .................. 44
   3.7 Other approaches ........................ 45
   3.8 Conclusions .......................... 47
4 Fault diagnosis based on the stuck-at fault model 49
4.1 Introduction ....................................................... 49
4.2 Identification of the faulty net .......................... 50
4.3 Mapping with a fault model ................................. 55
   4.3.1 Mapping with the stuck-at fault model .......... 57
   4.3.2 Mapping with the bridging fault model .......... 58
   4.3.3 Mapping with the interconnect open fault model .... 62
4.4 The format of the output list ........................... 65
4.5 Delay faults diagnosis ....................................... 66
4.6 Experimental results ........................................ 69
4.7 Conclusions .................................................... 81

5 Yield loss: causes and improvement methods 83
5.1 Introduction ....................................................... 83
5.2 The structure of a CMOS manufacturing process .... 84
5.3 Yield loss causes ............................................. 89
5.4 Yield loss classifications .................................. 93
5.5 Yield improvement techniques ............................ 94
   5.5.1 Yield improvement using in-line inspection data .. 95
   5.5.2 Yield improvement using test structures .......... 95
   5.5.3 Yield improvement using embedded memories .... 99
5.6 Conclusions and discussion ............................ 100

6 Statistical diagnosis for yield improvement 103
6.1 Introduction ....................................................... 103
6.2 Previous work .................................................. 104
6.3 Diagnosis algorithm description ........................ 105
6.4 Experimental setup .......................................... 107
6.5 Data analysis results ....................................... 108
   6.5.1 Performance analysis .................................. 108
   6.5.2 Fail histogram .......................................... 116
6.6 Case study results .......................................... 118
   6.6.1 Case study 1 ............................................. 118
   6.6.2 Case study 2 ............................................. 119
6.7 Conclusions .................................................... 124

7 Summary .......................................................... 125

Bibliography ....................................................... 129

Samenvatting ...................................................... 139
CONTENTS

Acknowledgment 143
About the author 145
Chapter 1

Introduction

1.1 Introduction

Integrated circuit (IC) technologies have reached the stage where the inclusion of more than one billion transistors on a monolithic piece of silicon is possible. IC’s are becoming in fact Systems-on-Silicon. Such extreme high levels of integration complicate the entire IC development and manufacturing process. Crucial test-related steps that are affected are initial system debug, the ramp to volume production, the yield improvement phase and the volume manufacturing test itself. In all of these manufacturing phases, the higher level of integration complicates the task.

During the manufacturing process an integrated circuit can fall victim to a large variety of failure mechanisms. Ideally, the related problems are detected early in the manufacturing process. However, some only show up during the final tests, or even worse, they might not be identified before the chip is integrated on a customer’s board. All these failure mechanisms have to be detected and characterized, so corrective actions can be taken to avoid them in the future. Increased competition in the IC marketplace demands that the correction of the detected failure mechanism needs to be executed as fast as possible to rapidly achieve high yields. Any delay can cause a loss of millions of dollars.

The first and the most critical step in the process of identification of a defect mechanism is fault diagnosis. Fault diagnosis aims at locating failures in chips that have been identified as defective. There are two main activities in which fault diagnosis plays an important role: failure analysis and low yield analysis. The objective of these two activities is different but the end result is the same, to improve the manufacturing process.

The objective of failure analysis is to find the defect in a given die, which fails a manufacturing test or a customer application. Fault diagnosis is the first and the
most critical step in the failure analysis process. Traditional techniques include Photon Emission Microscopy (PEM), liquid-crystal hot-spot analysis, Fluorescent Microthermal Imaging (FMI) and Electron-Beam Testing (EBT). Not all defects emit light or cause localized heating and the process for localizing them is very long and has a low rate of success. The integration density is increasing and the access to the IC’s transistors and wiring becomes more difficult. The defect identification process is often destructive and entails etching away metal layers for visual inspection. Therefore, software techniques are needed to provide the correct guidance to the fault location before the destructive action starts.

The goal of low yield analysis is different from that of failure analysis; it is to improve yield by identifying new, systematically repeating defect mechanisms and to drive corrective actions in the wafer fabs. For this purpose a large number of devices has to be analyzed, preferably on-line, so the results are available as soon as possible for a quick feedback to the manufacturing process. The complexity and level of integration demanded in the new designs continue to increase as the industry is moving towards System-on-Silicon technology and the importance of logic blocks increases significantly. Consequently, the traditional techniques based on using embedded memories and special test structures as yield improving monitors are no longer able to bring the yield to the desired high levels. Therefore new techniques, based on analyzing the defect mechanisms that occur in the logic part of the design, are needed.

The research described in this thesis tries to address both of the needs mentioned of the fault diagnosis process. Two methods were developed: a full diagnosis method which precisely and correctly pinpoints to the fault location as needed by the failure analysis process, and a statistical diagnosis method which is able to identify systematically repeating failing mechanisms as needed for yield improvement related activities.

1.2 Thesis outline

The outline of this thesis is as follows:

In chapter 2 the main steps of the failure analysis process are briefly presented. It starts with describing the electrical characterization step, than continues with the presentation of several physical fault diagnosis techniques and a short overview on the electrical diagnosis methods. Next, the deprocessing, defect localization, identification and physical characterization steps are described.

Chapter 3 presents a more detailed overview of the electrical diagnosis methods developed. These diagnosis methods are classified according to the fault model they use. The chapter begins with a short description of the most important fault models: stuck-at, bridging, open and delay fault model. The next sections present an overview of each class of the diagnosis methods developed.
1.2. Thesis Outline

In chapter 4 a new fault diagnosis method is described. This method improves the existing stuck-at based diagnosis infrastructure and it is able to locate stuck-at, bridging, interconnect opens and transition delay faults. The results of the experiments performed show the efficiency of the method.

Chapter 5 presents an overview of the yield loss causes and traditional improvement methods. A brief description of a manufacturing process together with some yield loss causes and classification criteria are presented in the first three sections. Then the traditional yield improvement techniques based on in-line inspection data, special test structures and (embedded) memories are described.

In chapter 6 a new statistical diagnosis method focusing on yield improvement is presented. This method tries to fill the need for a structured approach to identify systematically repeating failure mechanisms, which are the main cause for yield loss.

Finally, chapter 7 summarizes the research performed.
CHAPTER 1. INTRODUCTION
Chapter 2

Failure analysis

2.1 Introduction

In IC manufacturing, failure analysis is defined as the process of determining the cause of a failure. It combines electrical and physical methods in order to localize and characterize the defect that caused the failure. The results are used to improve and correct the manufacturing process and to avoid similar defects in the future.

A generic failure analysis flow is presented in Figure 2.1.

![Failure analysis flow](image)

Figure 2.1: Failure analysis flow
Due to the wide variety of the processes, defects and failure mechanisms, the failure analysis process is very complex and contains many loops between the steps shown in Figure 2.1. In principle the whole process can be viewed as consisting of two primary parts: the electrical isolation of the failure and the physical root cause analysis.

The isolation process consists of all actions taken to determine where to "look" for the defect. This process begins with electrical characterization of the failure mechanism followed by fault localization methods. The fault localization process is also often referred to as fault diagnosis. Subsequently, the physical analysis is gathering information about the failure mechanism and/or the physical cause of the failure. This information is used to understand how the defect was generated and how to eliminate the root cause. The physical part of the failure analysis process is destructive. It implies etching and polishing procedures during the deprocessing and defect localization steps to be able to get at the defect location and to precisely characterize it. Therefore, it is important that the fault location indicated by the localization step is the correct one, because it can be destroyed when searching at another location.

In this chapter the main steps of the failure analysis process, as presented in Figure 2.1 will be briefly discussed in subsequent sections. Finally, conclusions will be drawn.

2.2 Electrical characterization

Electrical characterization is the starting point of the failure analysis process. It provides information about how the device is failing from an electrical point of view.

There are three main categories of electrical failures [1]: continuity, parametric and functional failures. Continuity failures are caused by defects that affect the input/output connections of the device. If one or more parameters of the device are not between the specified limits than a parametric failure has occurred. When a device does not perform correctly the task for which it has been designed, a functional failure has occurred. A clear distinction between these three types of failures cannot be made, each of them can cause other failure modes. For example a device with an open input will determine many additional failures, functional as well as Iddq. These functional and Iddq failures appear due to incorrect input conditions and floating gates. A device with a parametric failure is functioning correct but for example only at a lower speed or at a different supply voltage than for which is has been designed. Such a device will fail the functional at-speed test resulting also in a functional failure.

The electrical failure modes mentioned before have their specific requirements for electrical characterization. Therefore, it is important for the failure analysts to have the failure mode expressed in simple terms when he starts the analysis. In the case when a device is failing different tests, the order of difficulty of analysis from the
2.2. ELECTRICAL CHARACTERIZATION

failure analysis perspective is: continuity, parametric and functional failures.

Continuity failures are characterized by measuring for opens and shorts on the
external pins of the device. Parametric failures require more complex measurements
for characterization. These include input and output voltage levels, power supply
current, frequency response, offset voltages or other parameters specific for a given
device. Each device has its own distinct parameters that have to fall within a range
of acceptable values. The large spectrum of parameters and their specificity increase
the complexity of the parametric failure characterization. Functional failure charac-
terization is performed by applying known stimuli at the inputs and measuring the
outputs. If the measured output value is different from the expected one, the device
is considered faulty.

One of the tools used for continuity and basic parameter analysis is the curve
tracer. It is used to measure the electrical characteristic of the I/O (Input/Output)
structures on the device by applying a variable voltage to a pin and displaying the
resulting current as a function of the voltage (I-V characteristics, see Figure 2.2). The
curve tracer can be used also to characterize internal device components as resistors,
diodes and transistors.

![Figure 2.2: An I-V characteristic of a PN junction](image)

The tools used for characterization of parametric failures depend mainly on the pa-
rameter analyzed. One of the instruments widely used is the semiconductor parameter
analyzer. It has programmable I-V source/monitor units and can measure very low
levels of currents and voltages.

Also the tools used for characterizing the functional failures depend on the type of
CHAPTER 2  FAILURE ANALYSIS

Figure 2.3: An example of the output from the wave tool

Figure 2.4: An example of a Shimoo plot, Vdd versus period
2.2. ELECTRICAL CHARACTERIZATION

failure encountered and the complexity of the device. For simple devices the analysis can be done with bench test electronics (power supplies, function generators, pattern generators, logic analyzers). For the complex devices the ATE (Automatic Test Equipment) is the main tool used. The debug features incorporated in today’s testers, like wave, vector, pattern, shmoo tools can be easily accessed through a graphical interface. The wave tool is a digital oscilloscope which displays both the input and output waveform of the device (Figure 2.3).

![Diagram of ASCII format of a Shmoo plot]

Figure 2.5: The ASCII format of a Shmoo plot

The vector or pattern tool displays the inputs and outputs for the entire vector set highlighting the failing pins within the vector or pattern. Shmoo plots are widely used as a characterization tool for the performance of the IC in relation with the changes in the environment such as temperature, timing and power supply voltage [2]. Multiple parameters (e.g. temperature, supply voltage) can be simultaneously varied while the IC is tested and the Pass/Fail information obtained is used to create a plot called shmoo plot. Usually the shmoo plots are created with ASCII characters as the one presented in Figure 2.5. Today ATE's have an improved display and GUI capabilities
and the graphical representation of the shmoo plots is also improved (Figure 2.4).

The failure analysis begins with the analysis of the datalog, which is in fact the measured continuity, parametric and functional output values from the production test equipment. This datalog is mostly incomplete as the production test program stops when the first failure is recorded. Therefore a retest is necessary to obtain the full datalog. In case the device has failed more tests, the analysis of the full datalog helps the analyst to determine if one failure mode is dominant and causes the others to occur.

Based on the results of the analysis, more detailed electrical characterization is sometime necessary for a better understanding of the failure mechanism. Shmoo plots generated over a wide range of temperatures and supply voltages combined with the values obtained from the Iddq testing are valuable information for the failure analysts to determine if there is a continuity, a parametric or a functional failure. It is important that the failure mode is correctly identified so the right tools and methods used for localization and identification of the defects can be chosen to target this specific failure type. For example in the case of a continuity failure that causes also functional failures, analyzing it as a functional failure will increase significantly the complexity of the analysis.

2.3 Fault localization. Fault diagnosis

Fault localization is the most critical step in the failure analysis process. Given the increased complexity of the ICs it is imperative to correctly localize faults before the destructive process begins. The localization can be done down to a logic block or circuit net, or in some cases directly to the defect location. There are two main groups of fault localization techniques: physical (hardware based) and electrical (software based). The first group requires significant investment in equipment, tools and qualified personnel in testing, chip architecture and the tool itself. Physical localization techniques are also time consuming but they can for many cases isolate the failure directly to the defect location. Electrical based diagnosis techniques are combining simulation results with the data from the tester to locate the failure. The simulation is using fault models that are not so accurate, having sometimes as result an imprecise or incorrect location.

2.3.1 Physical fault diagnosis

The physical fault diagnosis techniques roughly fall into two broad categories. The first one attempts to identify a secondary effect (e.g. thermal or optical) of the failure. The second are probe techniques where electrical signals within the device are measured.
2.3. **FAULT LOCALIZATION, FAULT DIAGNOSIS**

Many defect types result in a high power supply current, which produce heat (hot-spots) during operation. The infrared (IR), liquid crystal and Fluorescent Microthermal Imaging (FMI) techniques use this effect to locate the defects which cause it. The IR thermal techniques calculate the temperature of an object from its infrared emission. The hot areas on the integrated circuits can be easily detected and located. These techniques have an excellent temperature range and resolution but a poor spatial resolution, the hot-spots caused by thermal conductivity are diffuse, they have difficulties when the features are less than 5μm.

![Image of hot-spot detection using liquid crystals](image)

**Figure 2.6:** An example of hot-spot detection using liquid crystals

Liquid crystal based methods are commonly used for hot-spot detection. 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. The device to be analyzed is powered and placed on a precisely controlled heat chunk. A thin and uniform layer of liquid crystals is applied on the surface of the device. The chunk is heated until the temperature is above the liquid crystal’s transition temperature and a black image is obtained. The temperature is lowered until the average surface temperature is 0.1 degree under the liquid crystal’s transition temperature. The surface of the device is visible, however, black spots can be observed on the regions where the temperature is higher due to local dissipations. When alternating the power supply
voltage the black spots (the hot-spots) will change synchronously. This change can be observed by using polarizers together with a common optical equipment. For example in Figure 2.6, the black spot marked on the picture is the "hot-spot" of the device. Smaller feature sizes influence the spatial resolution of this technique. This technique needs relatively large dissipation powers. Therefore, the reduction in power supply voltages and hence power dissipation make the thermal sensitivity of liquid crystals an issue [3]. Also, when applying this technique an accurate and uniform surface temperature and good thermal contact are required, which are not easy to assure for packaged devices.

Fluorescent microthermal imaging (FMI) is another physical diagnosis technique which detects the hot-spot of a faulty device. It uses a film with a temperature dependent fluorescent quantum yield to generate a high-resolution thermal map [4]. This technique shows an improved combination of thermal and spatial resolution over other thermal techniques.

![Figure 2.7: PEM image taken from the front side](image)

The other important secondary effect, light emission, is widely explored and many techniques for physical fault localization based on this effect were developed. An IC during electrical operation emits light. Visible and near infrared wavelength photons are emitted from transistor’s pn junctions and other photon generating structures.
2.3. FAULT LOCALIZATION. FAULT DIAGNOSIS

![Image]

Figure 2.8: An example of the results obtained using backside PEM

These photons are transmitted through the relatively transparent dielectric layers and are passing between or are scattered around the metal interconnections. Detection of these photons is called front side PEM (Photon Emission Microscopy) (see Figure 2.7).

The imaging light passing through the silicon substrate and emerging from the back is called backside PEM. For both PEM techniques a reflected light image of the same field of view as the PEM image is acquired for registration of the light emitting area. The reflected light images are normally combined in an overlay with the emission image into a single processed image in order to facilitate localization. The analysis of the wavelengths of light emitted by the device pinpoints to a defect class. This technique is effective in identifying gate-oxide shorts, degraded pn junctions, MOS transistors in saturation due to interconnect shorts and open circuit defects [5]. In addition, spectroscopic PEM offers the capability of analyzing the behaviour of (hot) electrons in semiconductor devices [6].

Given the increased use of flip-chip packaging and the increased number of opaque metal layers, the interest for backside PEM is growing. The silicon becomes less transparent when more dopants are added, therefore many milling and thinning techniques of the silicon substrate were developed to improve backside access and signal integrity. An example of a backside PEM image is shown in Figure 2.8. The marked spot points to the region were the defect is located.

Other techniques take advantage of a light or an electron beam’s ability to interact with an IC and the images obtained as result of this interaction are used for fault localization. The results of the interaction of a light beam with the IC are a photocurrent (generated by electron-hole pair production) and heating. These both
effects are used by different defect localization techniques. Photocurrents are used in the Optical Beam Induced Current (OBIC) and Light-Induced Voltage Alteration (LIVA) imaging techniques. In case of the OBIC the electron-hole pairs are generated by an energetic laser beam and this technique is used to image depletion layers and to detect, among others, junction leakage’s.

Local heating is analyzed by Optical Beam Induced Resistance Change (OBIRCH), Thermally Induced Voltage Alteration (TIVA) and Seebeck Effect Imaging (SEI). OBIRCH and TIVA are used for detecting mainly shorts and resistive conductors and SEI for open conductors. Later developments in the OBIRCH technique have also shown the applicability of this method for localization of resistive opens [9]. All of these techniques can be applied from the front and from the backside of an IC by use of a proper optical wavelength.

![Image](image.png)

**Figure 2.9:** The principle of the OBIRCH method

The principle of the OBIRCH technique (Figure 2.9) is as follows. The device to be analyzed is connected to a constant voltage source and an IR-laser beam scans across the device. The current change caused by the laser beam heating is proportional to the resistance change, which is also proportional to the temperature increase. However, heat conduction is impeded when the beam encounters defects such as opens and Si precipitates. This creates a temperature gradient between the irradiation points that are near the defect and the points that are not. An image can be created point-by-point in the form of a brightness change. Figure 2.10 shows an example of an image obtained after applying the OBIRCH technique. The marked spots on the picture are indicating the presents of a defect.
2.3. FAULT LOCALIZATION. FAULT DIAGNOSIS

The TIVA technique is similar to OBIRCH. In this case the device is connected to a constant current source and again an IR-laser beam scans across the device. When a defect is encountered, the voltage may be changed and this modifications can be visualized.

Similar to the methods mentioned above, several techniques were developed which use an electron beam to generate electron-hole pairs. Three of the most important techniques are: Electron Beam Induced Current (EBIC), Resistive Contrast Imaging (RCI) and Charge-Induced Voltage Alteration (CIVA). In the EBIC technique the electron beam generates electron-hole pairs in the semiconductor. The pairs generated in the depletion layer are separated by the electrical field of the depletion layer and are detected as an EBIC signal. Figure 2.11 presents an EBIC picture of a J-FET transistor.

In the last years, PICA (Picosecond Imaging Circuit Analysis) became a powerful diagnosis tool for detecting timing related defects [7]. Hot electron light emission is generated as a subnanosecond pulse coincident with the normal logic state switching of CMOS circuits. The emission arises from intraband transition of hot carriers and can be observed and used directly to measure the propagation of high-speed signals through individual gates in fully-functional CMOS circuits. The hot carrier luminescence can be observed not only from the front side, but also from the backside of a
CHAPTER 2  FAILURE ANALYSIS

Figure 2.11: EBIC picture of a J-FET transistor

die after it is thinned [8].

The fault localization techniques mentioned until now don’t address all the problems that can be encountered during the fault localization process. When debugging a new design and no physical defect seems to be present or when the defect doesn’t produce light or heat that can be detected by the above mentioned techniques, probing from a failing output back into the circuit to isolate the origin of the failure is the only practical physical localization option. This is a quite time consuming technique and often simulations and pre-localization software methods are applied before the probing starts. With the increasing complexity of today’s designs, software techniques that by processing the failing information and simulation data are able to reduce the area that is needed to be probed are becoming prerequisite.

Probing is the collection of electrical measurements from IC’s internal nodes. These measurements can be: the voltage on a single node, timing, waveform or parametric (I-V curves of a diode junction). The most common techniques are the mechanical and the e-beam probing.

Mechanical probing is the oldest and simplest probing technology. It is used to measure voltage, current and impedance by direct electrical contact between the probe and the signal line on the die. Simultaneous measurements at several locations in parallel are also possible when using multiple mechanical probes. This technique has
2.3. FAULT LOCALIZATION. FAULT DIAGNOSIS

also some limitations and disadvantages. It became more difficult to probe deep sub-
micron lines with normal mechanical probes. Mechanical damage from time to time
is inevitable when probing the microscopic features on the active surface of an IC.
To avoid this, probe-points are specially included in the design phase of the device or
created later for failure analysis purposes using a FIB (Focused Ion Beam). The metal
lines, when exposed to air and moisture, will be covered with a thin layer of oxide so
this has to be scraped away or broken using the probe before a reliable contact can
be made. For this reason, often the probe is pushed inside the conductors to minimize
the chance of damage due to probe sliding. The contact resistance formed and the
electrical loading will influence the accuracy of high precision measurements.

The disadvantages related to mechanical probing are eliminated when using the
e-beam technique. It uses the voltage contrast phenomena in a Scanning Electron
Microscope (SEM). A pulsed e-beam is directed to the node of interest which will
provide a mode of operation like a sampling oscilloscope enabling the acquisition of
high-speed quantitative voltage waveforms. The e-beam technique has a high spatial
resolution, non-contact, non-loading, high bandwidth and can probe buried conduc-
tors. Nevertheless there are also some limitations. It can not measure intermittent
signals, has limited accuracy for voltage measurements and the field effects on the
device can deflect the primary e-beam resulting in an incorrect captured image.

The increase in flip-chip utilization has limited and complicated the access for
probing from the front-side of the device. The traditional techniques, mechanical and
e-beam probing, can be used from the backside if first a hole is drilled to expose the
node that has to be measured. In order to eliminate the drilling procedure, a new
optical probing technique called Laser Voltage Probe (LVP) [10] has been developed.
This technique is based on an infrared laser and allows signal waveform acquisition
and high frequency timing measurement directly from active P-N junctions through
the silicon substrate’s backside.

2.3.2 Electrical fault diagnosis

In this section an overview of the electrical diagnosis techniques will be given. A more
detailed description is done in chapters 3 and 4.

Electrical diagnosis techniques can be classified in two groups [11]: cause-effect
and effect-cause analysis. Cause-effect diagnosis techniques (Figure 2.12) are using
fault simulation to determine the possible response of a circuit in the presence of
faults when a test set is applied. This information is then matched with the response
obtained from the tester in order to obtain the fault location.

One strategy for cause-effect analysis is to use a precomputed fault dictionary. This
approach is often referred to as static cause-effect diagnosis. A fault dictionary is a
database that contains the faulty responses of each modeled fault. Various diagnosis
algorithms are used to determine the fault which best matches the behavior observed on the tester [12, 13]. In some cases, there are more faults that are determined as the best match for the tester output when a particular test set is applied. One solution is to generate special test patterns to distinguish them [14, 15]. A disadvantage of this approach is that the device must be retested applying this new set of test patterns. Another solution is to use more complex fault models when creating the fault dictionary.

Building a fault dictionary for all possible faults is not practical. Given the increasing complexity and dimension of today’s ICs, the files obtained will be very large (GB) and it will be difficult to store and handle them. Several methods have been developed for compressing the size of the fault dictionaries [16, 17, 18, 19] with some loss in the precision and accuracy of the diagnosis results. Other research was focused on developing heuristics for reducing the initial fault list by using a limited fault dictionary. Then dynamic diagnosis algorithms are used to locate the fault [20, 21].

The other group of diagnosis techniques, effect-cause analysis [22, 23, 24], processes the response obtained from the tester to deduce the internal values of the IC under test. The deduction procedure starts at the primary outputs (PO) and tries to justify simultaneously all the values obtained at be PO’s. The situation where there exists only one way to justify a value represents an implication. If there are more
alternatives, a decision is made and one possible situation is considered. The decision process requires the use of a backtracking strategy, such that if a decision leads to an inconsistent state (i.e. a state in which some consistency property is violated), the algorithm backtracks to the last decision point and tries an alternative decision. Based on the deduced values the method can identify the fault location within an equivalence class. This approach does not require fault enumeration and a comparison mechanism between an expected response (mainly obtained with fault simulation) and the tester results.

One of the important aspects of the fault diagnosis process is the fault model used. Cause-effect and effect-cause diagnosis techniques were developed based on the stuck-at [12, 13, 16, 22], bridging [23, 25, 26], open [27] and delay fault models [24, 28]. Moreover, model independent fault diagnosis methods have been proposed [35, 36]. These methods identify the fault location without pinpointing its nature.

The single stuck-at fault model, which is the most common model used, has been proven to be effective in providing high coverage for other types of fault when used for test pattern generation [29], but is not as efficient for diagnosis [30]. Nevertheless, methods for diagnosing bridging faults [25, 31, 32], opens [27] or mixed fault [33, 34] based on the stuck-at fault model were developed. The use of a complicated matching algorithms tries to overcome the fact that the method is based on a simple fault model.

2.4 Deprocessing

Once the fault has been localized as precisely as possible, the die must be prepared for further characterization and inspection. Depending on the nature of the failure and the accuracy of the localization process we may have to remove one or more layers of the die. The purpose is to provide the analyst more accessibility to signals for defect site isolation and also visibility for physical and chemical characterization of the defect. Deprocessing is in fact the reversal of the manufacturing process, the thin film layers are removed sequentially in the reverse order of application in the wafer fab. Therefore a good knowledge of the manufacturing process is necessary before the removing process starts. Deprocessing can be performed on a die while still in the package or on a die removed from the package. The methods used are the same. When performed on a die in a package, the extra steps for removing the die from the package are avoided. Deprocessing can be performed locally, only on a small surface of the die, or globally, when an entire layer from the entire die surface is removed. Moreover, deprocessing can be applied selectively and only one type of material, for example the oxide layer, is etched. The type of deprocessing method applied depends on the nature of the analysis required.
There are three basic methods for removal of layers [37]: wet etch, plasma etch (Figure 2.13), and mechanical polish. A deprocessing scheme consists mostly of a combination of these techniques, for example plasma etch for dielectrics and wet etch for metals. Deprocessing is a destructive process. If implemented improperly or hastily one can lose valuable information which is needed to understand the physical cause of the failure. In practice, typical deprocessing of the die is accomplished in discrete steps and inspection of the affected area where the defect has been isolated is performed between each deprocessing step.

2.5 Defect localization

Not always the fault localization techniques are very precise in pinpointing the defect location. Therefore, after deprocessing, a second step for defect localization may be necessary. Some of the hardware diagnosis techniques mentioned previously (e.g. e-beam or mechanical probing) can be used to isolate the actual location of the defect when a long conductor is affected.

To locate defects which are difficult to observe after deprocessing, for example defects that occur at the bottom of etched holes like vias and contacts, performing a cross-section is the only way to directly localize the defect. Making a cross-section means actually cutting the device through one ore more layers in order to gain better
observability of its internal structure. This technique also gives more information about the point in the manufacturing process where the defect was introduced than the one obtained if only deprocessing is performed.

![Cross-section realized with a FIB]

Figure 2.14: Cross-section realized with a FIB

Depending on the precision needed several sectioning techniques were developed: sawing and grinding for localizing defects related with packaging and assembly in packaged devices; wafer cleaving for layer and localized structure problems; die polishing for large defects (> 0.5 microns) and FIB for smaller defects (< 0.5 microns) (Figure 2.14). It is also possible to simultaneously observe the side-walls of the cross-section with a FESEM (Field Emission Scanning Electron Microscope) while milling with a FIB.

There is an increasing challenge to improve these sectioning techniques in order to achieve higher precision while continuously shrinking of the features of interest. FIB techniques [38] have the best precision but the ion beam produces some damages to the sample. New materials introduced with the emerging technologies will impose the development of other sectioning techniques.

2.6 Inspection and physical characterization

The finding and identifying of a defect is an interactive process with sample preparation (deprocessing, cross-section) followed by inspection which leads to additional sample preparation and inspection. After the defect is identified it must be charac-
terized to determine its material properties. This information is important for the manufacturing process and is used to determine the source of the defect. The three most common inspection microscopy techniques used are: optical, scanning electron and scanning probe microscopy.

Optical microscopy is easy to use, doesn’t require vacuum, it allows viewing through numerous vertical transparent dielectric layers. The disadvantages are its limited resolution and depth of field at high magnifications because the new feature sizes are below the diffraction limit of the visible light.

Scanning Electron Microscopy (SEM) [39] provides an improved level of resolution and depth of field compared with optical microscopy (see Figure 2.15). It can also evaluate material properties of the defect such as atomic weight and chemical content.

Another inspection technique, which is suitable for defects localized in very small areas, is Scanning Probe Microscopy (SPM). This technique has an atomic scale resolution and can characterize electrostatic potential, capacitance, atomic force and topography across small areas.

A technique with very high resolution used particularly for viewing cross-sections is Transmission Electron Microscopy (TEM). It also supports complex chemical analysis and the images obtained have a good contrast. An example of an TEM image of a cross section is shown in Figure 2.16. TEM requires preparation of extremely thin samples which is a very difficult and tedious process.

Figure 2.15: SEM picture of a defect called Vulcan/pancake
Figure 2.16: Example of a TEM image of a cross-section

Figure 2.17: Example of the analysis results obtained by using AES
CHAPTER 2  FAILURE ANALYSIS

When these techniques cannot determine the material composition of the defect, more sophisticated chemical and material analysis tools, for example Auger Electron Microscopy (AES) and Electron Spectroscopy for Chemical Analysis (ESCA), can be used. Figure 2.17 presents the images obtained after using AES for material analysis of a device.

2.7 Conclusions and discussion

Failure mechanisms are unfortunately an inherent part of the semiconductor business. Therefore, the electrical and physical identification of the root cause of the failures as a result of a complex failure analysis process is very important. The ability to apply confidently corrective actions has a direct impact on reducing the yield learning period, time-to-market and increasing the customer satisfaction.

Integrated circuit technology continues to progress. New materials and new manufacturing process are introduced, the physical dimensions are shrinking to values close to atom sizes. All these are limiting the applicability of some of today's physical techniques making some of them obsolete. There is a continuous demand for improving the actual techniques and developing new ones when the old ones reached their limits.

Fault localization is a critical step for failure analysis. The difficulty degree for fault localization is increasing with an order of magnitude every six years [40]. Today, the hardware and software diagnosis techniques exist as independent solutions. With the disappearance of the front-side access to the die, the physical diagnosis techniques will become significantly limited. One of the possible solutions is to use more intensively software tools and electrical testing in combination with the actual physical techniques to achieve high localization precision and accuracy. New backside techniques are greatly needed together with improvements in defect modeling and simulation, test pattern generation for high defect coverage, and localization algorithms.
Chapter 3

Electrical fault diagnosis

3.1 Introduction

As mentioned in the previous chapter, due to the development of smaller and denser manufacturing processes most of the hardware localization techniques cannot keep up satisfactory with the technology trend. There is an increasing need in precise and accurate electrical diagnosis tools to help identify as fast and correct as possible the fault location.

Traditional diagnosis techniques involve two primary elements: a fault model and a matching (comparison) algorithm. In this chapter, after briefly describing the most important fault models, the fault model they used organizes the diagnosis approaches presented. Also techniques which can not be easily categorized by a fault model or matching algorithm are presented in a subsequent section.

3.2 Diagnosis and fault models

It is common to speak of locating or identifying faults in a device when the target is actually a physical defect. The physical defects do not allow a direct mathematical treatment for testing and diagnosis. Therefore, logical representations of the effect of the physical defects on the operation of the system, called fault models, were developed. The faults models commonly used in the diagnosis process are: stuck-at, bridging, open and delay fault model.

3.2.1 The stuck-at fault model

The most common and the first fault model developed is the single stuck-at (SSA). In this model it is assumed that a single signal line or a node somewhere in the circuit
has taken a fixed logic value and is "stuck-at" 0 or 1. The physical defect with which
the stuck-at fault model is mainly associated is a short with a power line for a stuck-at
1 fault (Figure 3.1. a) and with the ground for a stuck-at 0 fault (Figure 3.1. b). If
the value of the signal at the gate output from the Figure 3.1. remains always at the
same value (1 for Figure 3.1. a and 0 for Figure 3.1. b) independently of the values
at the gate inputs than a stuck-at fault has occurred.

![Diagram](image)

**Figure 3.1:** The stuck-at fault model; a) stuck-at 1, b) stuck-at 0

However, these are not the only physical defects that can be modeled using the
stuck-at fault model. For example, an open connection at the gate of a NMOS tran-
sistor in the presence of hot-holes in the gate oxide, or other positive charges, may
turn the transistor permanently on, hence creating a stuck-at fault [41]. Also
punch-through or avalanche breakdown induced shorts between source and drain, shorts
of the gate to the channel can be modeled as stuck-at faults at the transistor's gate [41].
Nevertheless, the stuck-at fault model is not a perfect representation of the behavior
of a wide range of physical defects like shorts between two signal lines or interconnect
opens. For these defects special models were developed which will be discussed in the
following sections.

In spite of this, there are many advantages when using the stuck-at fault model:

- It is independent of technology, the concept of a line being stuck at a logic value
can be applied to any structural model.

- The effect of a stuck-at fault can be easily modeled using Boolean equations,
which simplifies the automation of the test pattern generation procedure.

- Compared with other fault models (as for example the bridging faults) the
number of stuck-at faults is relatively small and fixed. The SSA model is applied
usually at gate level and every circuit node has exactly two faults. In the case of
bridging faults, the signal net connecting two nodes in a circuit can have more
bridging faults, depending on the length of the net.
3.2. Diagnosis and Fault Models

- The number of stuck-at faults is directly proportional with the size of the circuit and can be reduced by fault-collapsing techniques. For example, in Figure 3.2, the stuck-at 0 (SA0) at the output of the AND gate is equivalent with the SA0 at any of the inputs. Also a SA0 at the output of an inverter is equivalent with the SA1 at its input. All these stuck-at faults form an equivalence class and only one fault can be retained as representative.

![Figure 3.2: Example of fault equivalencies classes](image)

- The SSA model can be used to model other types of faults [11]. This is shown in Figure 3.3. A multiplexer (selector) can be inserted on the signal line where we want to model the defect as a stuck-at fault. For \( S = 0 \) the new circuit operates identical to the initial one. If \( S \) is stuck-at 1 (\( S = 1 \)), the new circuit can realize any faulty function \( Z_f \). For example if we connect input \( A \) to \( Z_f \) via the multiplexer, the functionality of the gate will be modified (\( Z_f = \hat{A} \) and not \( Z_f = \hat{A&B} \)). This approach for fault modeling is very flexible, but increases significantly when more complex defect behavior needs to be modeled.

- The test patterns generated using the SSA model detect also other fault types, for example they give a high coverage for resistive bridge faults [29]. Moreover, in [42] it is shown that tests generated for stuck-at faults will detect other types of faults by activating logic in an particular area and make it observable at the primary outputs. The tests generated to force repeated observation of all circuit nodes are more effective than standard tests in detecting faulty devices.

Although its validity is not universal, the SSA model remains the most commonly used even with the new nanometer technologies. R. Aitken [43] considers that its
Figure 3.3: Example of fault modeling using the stuck-at fault model

dominance will probably not end even if other fault types will become more common than they are today and new defect types will emerge.

3.2.2 Bridging fault models

Bridging fault models describe the device behavior when physical defects cause signal nets to be shorted together (Figure 3.4 a). Each signal net is the output of a different gate which attempts to drive the shorted net to a value equal to the gate output in a fault-free circuit.

The bridging fault model was first introduced by Mei [67]. In this model it is assumed that the value on both shorted signal nets will be equal with the result of an AND (Figure 3.4 b) or an OR (Figure 3.4 c) operation between the values on the nets. These fault models are also known as the wired-AND and wired-OR bridging fault models and are the easiest to implement for simulation, test pattern generation and diagnosis. In the dominant technologies of the time when these models were developed such as DTL (diode-transistor logic), RTL (resistor-transistor logic) and ECL (emitter coupled logic) the bridges did behave as wired-logic fault models. However, the wired-logic models do not accurately reflect the behavior of the bridges in CMOS technologies [41, 44]. The values on the shorted signal nets depend on several factors like the characteristics of NMOS and PMOS networks of the gates driving the shorted nets, the values on the inputs of these gates and the resistance of the defect which causes the short. Sometimes the resulting value can not be interpreted as a logic value, some of the gates may interpret it as a logic 1 and others as a logic 0, depending on their threshold voltage. This situation is known as the Byzantine General’s problem [68]. When the number of the gates driven by the shorted nets is relatively high (the shorted nets have a large fanout) this uncertainty is complicating the test pattern generation and diagnosis procedures; if a net has k fanout branches, there are $2^k$ logical possible situations as seen from the stem.
A more complex fault model, which also takes the Byzantine General problem into account, is the two-component fault model. This model assumes that the value on the shorted nets is described by a Boolean function of the inputs of the gates driving the shorted nodes. In [69] the authors proposed the voting model as a method for deriving the two-component model. In the voting model, SPICE simulations are done to generate tables with the relative strength of networks consisting from series connected and parallel connected transistors of the same type (N or P). When a bridging fault between two gate outputs is considered, the strength of the driving transistor networks is computed using the tables. The stronger network will "win" the vote and its output will determine the resulting logic value on the shorted nets. The relative strengths contained in the tables are only used to compare the strength of the transistor networks that are shorted, they are not used to calculate voltages. Even when the resulting voltage is between the logic values, this model assumes that the bridge value is always digitally resolvable, the gates driven by the shorted signal nodes will interpret it as a 0 or a 1. This model is not completely general, the situation when the strength of the two transistor networks have close values are not accurately modeled. Moreover, if a new gate that has a threshold voltage outside the ranges used by the simulator to compute the tables, the table generating procedure has to be rerun. To overcome these shortcomings, a more complex version of the voting model, called biased voting that can also determine the actual bridged voltages, is proposed in [70]. The voting procedure is more flexible, the threshold is not considered fixed as in [69]. In order to determine if a bridged voltage is above or below a given threshold, the relative strengths of the N and P networks are compared on a weighted basis.

A more realistic model assumes that in some cases the voltage value on the shorted nets can not be considered as a logic 1 or 0. The model proposed in [72] uses also the thresholds of the inputs driven by the shorted nodes to determine the voltage of the bridge. This model is determined using analog simulation and assumes that the effect of the bridging fault propagates one level downstream from the fault site. This assumption is not always valid, sometimes detailed analog simulations of a larger...
NMOS transistor model

\[
\begin{cases}
    \text{Ids}=K_n \frac{W}{L} (V_{gs}-V_{tn}-\frac{1}{2} V_{ds}) V_{ds} \text{ when } V_{ds}<V_{gs}-V_{tn} \\
    \text{Ids}=0 \text{ otherwise}
\end{cases}
\]

\[\beta = \frac{(K_p/(W_{pe}/L_{pe}))/((K_n/(W_{ne}/L_{ne}))}{(\beta-1)V_o+2(V_{dd}-V_{tn}-\beta V_{tp})V_o-\beta (V_{dd})-2V_{tp})V_{dd}=0} \quad (1)\]

Figure 3.5: Bridge fault modeling using Boolean Expression: the simplified transistor model (a); the transformed circuit (b)

region of a circuit is necessary for a correct modeling of a bridge behavior. The EPROOFS simulator described in [91] uses a SPICE-like analog simulation of the region around the fault sides, including the gates located at a certain distance around the fault location, combined with a digital simulation for the rest of the circuit. Extensive simulation that takes a lot of computational time and power is needed to be able to cover all the possible combinations of gates located before and after the bridging fault.

In order to reduce the simulation time, in [71] the authors proposed a method based on Faulty Boolean Expressions (FBE) to calculate the bridge value. By using a simplified transistor model (see Figure 3.5 a), each conducting circuit is transformed into the one shown in Figure 3.5 b, and then the output value \( V_o \) is computed by solving the equation (1).

The fault models presented until now did not take into consideration the resistance of the defect that causes the signal nodes to be shorted (Figure 3.6). In [73] a fault model which does this when determining the logic value of the bridge is presented. A general electrical model based on the equations that describe the operation of a MOS transistor is proposed. These equations are used to determine if the bridge voltage is above or under a certain threshold value. Several approximations are used when computing the transistor equations, which makes that the model is not very accurate when the resulting value is not around VDD/2. A more complete analysis that takes also in consideration the input values on the gates driven by the shorted nodes is
3.2. Diagnosis and Fault Models

![Resistive bridging fault diagram]

Figure 3.6: Resistive bridging fault

Presented in [29]. Five different cases of bridging faults (BF) are considered: BF between two primary inputs, BF between a primary input and a gate output, BF between two gate outputs (bridged nodes feeding into different gates), BF between two gate outputs (bridged nodes feeding into same gate) and BF between primary outputs. For each situation look-up tables are built which contain the sensitizing vector, the propagation path, the logic threshold of the propagating gate and the maximum value of Rb for which the bridging fault is detected.

![Feedback bridging faults diagram]

Figure 3.7: Feedback bridging faults

However, the models presented here cannot predict the behavior of all the possible bridging faults. The transistor parameters and bridge resistance are susceptible to
process variations which are difficult to predict accurately. Therefore, the models that use these values are not always accurate. To further complicate matters, feedback bridging faults (when the value of one shorted node A depends on the value of the other node B as shown in Figure 3.7 a) can occur. The feedback bridging faults transform a combinational circuit into a sequential one. Even more, feedback bridging faults can cause the circuit to oscillate if there is an odd number of inversions in the loop (Figure 3.7 b). Another situation that was not modeled is when there are more than two lines shorted by a defect. All the reasons mentioned before show that more research and study on fault models for bridging defects is needed.

### 3.2.3 The open fault model

Open circuit defects mean missing material which cause electrical discontinuity. An open can occur inside a CMOS cell, at the gate input of a transistor or on the interconnect wiring between the cells. Their properties depend on defect size, defect location, local electrical structure and process variables [44].

![Figure 3.8: Open defects at a transistor gate; a) small open; b) large open](image)

For example, (Figure 3.8 a) a small open defect to a logic gate where a tunneling current (J tunnel) occurs will increase the fall and the rise time of the gate input node [45]. Consequently, the rise and the fall time at the output of the gate will also increase. A large open (Figure 3.8 b) at the same location which decouples the gate input from the signal will create a floating gate node. The behavior of the floating node is a function of the local electrical structures. There are two expected responses from a large open at a gate input node. The first is that both pair of transistors are
3.2. Diagnosis and Fault Models

On, creating an elevate Iddq current and the second response is that only one of the pairs is on, the other is off. The second situation is very difficult to model, it depends on the local electrical structures like the length of the interconnect net which has the open defect, the characteristics of the gate transistors, the values on the signal nets situated in the neighborhood, etc.

The best known open model is the stuck-open fault model [46]. In this model (Figure 3.9) the gate of a transistor is fixed at the open or "off" value and it cannot pull the cell output to its voltage. For example if the value at the input A is 1, the logic value of Q remains unchanged. This behavior is also referred to as the "memory" fault.

![Stuck-open model diagram](image)

Figure 3.9: The stuck-open model

Maly et al. show in [47] that the stuck-open fault model covers only a little fraction of faults caused by actual opens. They classify three different single open fault models, namely the opens that cause the gate, the drain and the source node to be disconnected. In Figure 3.10 a model for the NMOS transistor with an open at the gate, drain and source is presented. This model is used for the analysis of the behavior of a transistor in the presence of an open defect.

More accurate modeling of open defects inside the cell by taking also in consideration the hazard and the charge-sharing effect is presented in [48]. The authors are using Reduced Ordered Binary Decision Diagram (ROBDD) data structures to represent the detection conditions.

The number of metal layers and vias is increasing. In new designs the number of vias is higher than the number of transistors. Therefore, the probability of a break on the interconnect wires is increasing with the denser technologies. It is difficult to model precisely the effects of interconnect opens. Konuk et al. [49] described 4 factors that
influence the voltage of an open interconnect wire: total wiring capacitance, transistor capacitance, trapped charge and die surface. They studied also the condition for which an interconnect signal line with an open defect can oscillate.

Predicting the behavior of an open defect is very difficult. The stuck-open fault model is considered not representative for many actual CMOS defects. All the other models mentioned here are targeting a limited class of open defects (inside cell opens or interconnect wiring opens) and recommend specific test methods to detect them. More research is needed to create an open fault model that can explicitly cover all the effects of an open defect. Until now, test methods (Iddq [50] and delay test [51]) that target open defects implicitly have been proven to be successful in detecting opens.

### 3.2.4 The delay fault model

Some of the manufacturing defects cause logic circuits to malfunction at the desired frequency, though they function well at a lower speed. These timing related failures are modeled as delay faults. There are three classical delay fault models: transition fault, gate delay fault and path delay fault. Two other models (i.e. line delay fault and segment delay fault) were developed in the last years and are derived from the classical ones.

A defect that delays a rising or a falling transition at the inputs or at the output of a logic gate can be modeled by the transition fault model [52]. There are two types of transition fault: slow-to-rise and slow-to-fall. The slow-to-rise transition fault
3.2. DIAGNOSIS AND FAULT MODELS

temporary behaves as a stuck-at 0 fault, respectively, the slow-to-fall like a stuck-at 1. For example, if in Figure 3.11 a defect which cause a slow-to-rise transition fault is present at the output of the AND gate, the signal will have the good value much later than expected.

![Diagram of a slow-rise transition fault](image)

**Figure 3.11: Example of a slow-to-rise transition fault**

The gate delay model has been introduced by Carter et al. [53]. It assumes that a gate delay occurs when a gate operates more slowly than expected. A fault is an added delay of a certain size in the propagation of a falling or rising transition from the gate input to the gate output.

Both the transition and the gate delay fault models assume that all the extra delay is concentrated in one location, at one gate inputs or output or, respectively inside the gate. They don’t take in consideration the situation when the delay is distributed over many circuit elements. Also, in order to detect them, the delay has to be propagated to a circuit output (the propagation path is not important) and has to be large enough to cause a logical failure at that output. The path delay fault model takes in consideration this situation too and it models distributed defects which affect an entire path. It assumes that any path with a total delay exceeding the system clock interval has a path delay fault.

The path delay fault model was first proposed by Smith [54] and has been extensively studied. For each physical path (Figure 3.12) which connects a primary input to a primary output of a circuit, there are two corresponding path delay faults: the rising path is the path traversed by a transition that is initiated as rising transition at the primary input and respectively, the falling path (P10 from Figure 3.12) is the path traversed by a transition that is initiated as a falling transition at the primary input.

The number of paths in a circuit is very large and it is impossible to generate tests
for all of them. Moreover, not all the paths are testable, the condition to propagate the transition along the desired path cannot be determined. The line and segment delay fault models were developed in order to overcome this disadvantage.

The line delay fault model [55] combines the relevant features of the transition and path delay fault models. A line delay test will test the longest sensitizable path passing through the target line. With this model the number of faults is limited to twice the number of lines in the circuit and can detect also some of the distributed defects by testing the longest path propagation path through a line. However, some of the distributed delay defects can be missed if they are situated along shorter paths through the target line.

The segment delay fault model proposed by Heragu et al [56] considers slow-to-rise and slow-to-fall defects on segments of length \( L \). The length \( L \) can be chosen from available statistics about the manufacturing defects. \( L \) can be 1 (transition faults) or as large as the maximum logic depth (path faults). The number of the faults to be tested is directly proportional with the value chosen for \( L \). For small values of \( L \) some of the distributed delays can be missed, for large values the number of faults to be tested can explode as for the path delay model.

With the new circuits which are working at GHz frequencies, generating effective delay tests is a major challenge. The effectiveness of the line and segment delay fault models has not been proven yet even though theoretically they should give good fault coverage. The gate delay model is strongly dependent on nominal delays which are not always accurately known. The propagation paths preferred by the test pattern generators are the short ones instead of the long paths and small delays which are distributed along long paths can be missed. The best choice available to detect these
3.3. STUCK-AT FAULT DIAGNOSIS

small distributed delays seems to be the path delay fault model. To overcome the
disadvantage of the high number of paths in a circuit when using the path delay
model for test generation, research has been done in finding a subset of all the paths
which must be tested [57, 58].

Not all the circuit delays that are present in the new nanometer technologies (i.e
delay caused by crosstalk, ground bounce) are modeled by the existing delay fault
models. To detect these emerging delay defects, new delay fault models have to be
developed and validated in the near future.

3.3 Stuck-at fault diagnosis

The single stuck-at fault model has been used by researchers to develop fault diagno-
sis methods from the sixties and early seventies [77]. These methods were based on
the effect-cause analysis; a fault dictionary is created up-front by simulating stuck-at
faults and a matching algorithm is used to select the best candidate. The first matching
algorithms were looking for candidates whose fault signature matches exactly the ob-
served faulty behavior or include it. Also, closest-match analysis was implemented
to determine which candidate to select when no exact match could be found. The
 closeness between two signatures can be calculated by using the expression:

\[ C_{12} = \frac{(S_1 \cap S_2)}{(S_1 \cup S_2)} \]

where \( S_1 \) and \( S_2 \) are the fault signatures of fault \( f_1 \) and \( f_2 \).

A more complex system for fault diagnosis called DORA (Diagnostic Organization
and Retrieval Algorithms) was using a matching algorithm called by the author "fuzzy
match" [79]. The fuzzy match technique is placing tolerance on the fault signatures
obtained by simulation allowing that some real fault behavior do not match exactly
onto the simulator predicted failure symptoms. In [80] the authors introduce the
concept of prediction penalties. The comparison algorithm is made at the level of
primary output/pattern (POPAT) pair. A candidate is penalized for each POPAT
pair found in the stuck-at fault signature and not found on the tester output, and for
each POPAT pair found on the tester output and not found in their fault signature.

Kundra [12] proposed a matching algorithm based on partial-intersection with
threshold. For each candidate a fault count is computed as being the intersection
between their fault signature and the observed behavior. The candidates are sorted
in descending order by their fault count. A threshold for partial-intersection is defined
as the set of faults which have the highest fault count. When applying the matching
algorithm with the threshold set to 1, the output list will contain the faults with the
highest fault count. If for example the threshold is set to 2, than the output list will
contain the faults with the top 2 highest fault counts, for threshold 3 the faults with
the top 3 highest fault counts, etc. This algorithm tries to overcome the situation when
no fault candidate can explain all the failures observed but suffers from imprecision, the output list contains a high number of fault candidates.

A more sophisticated algorithm of parameterized matching is presented in [13]. The faults are ranked according to their merit values. For each fault \( f \) the erroneous (Detect\((f)\)) and error-free (NoDetect\((f)\)) behavior are computed by comparing the information obtained on the tester with the simulation results of the stuck-at fault. The merit value of \( f \) is then:

\[
\text{Merit}(f) = C_1 \ast (\text{NFO} - \text{Detect}(f)) + C_2 \ast \text{NoDetect}(f)
\]

where NFO is the number of erroneous outputs.

The user can specify the relative importance of matching the erroneous behavior and the error-free behavior of the faulty part. This method can explicitly target defects that cause faults that behave similar to the stuck-at model and multiple distinguishable stuck-at faults.

An even more complex matching algorithm is proposed by Sheppard and Simpson [16]. They adapt the information flow model developed earlier for system-level diagnostics [81], to fault dictionary-based diagnosis of integrated circuits. A probabilistic evaluation based on the Dempster-Shafer statistical inference [82] is then used to evaluate the list of candidates.

One of the major bottlenecks for the application of fault dictionaries for stuck-at fault localization is size. For today's microprocessors, the storage and processing time are prohibitive. To address this problem, a two-step diagnosis method is proposed in [21]. A fault dictionary is created only for a subset of the possible stuck-at faults. Before the diagnosis process begins, the test set is partitioned and the faults detected by each group of test patterns are stored in so called "coverage lists". The first stage of the diagnosis algorithm reduces the fault list by intersecting the coverage lists of the failing tests. For these reduced lists a fault dictionary will be created and a matching algorithm identifies the suspect faults. The matching algorithm assigns a score for each fault in the list. When looking in the fault dictionary, if a fault is detected by the same vector as the one that failed on the tester its score is incremented by 1, the score of the other faults is reduced by 1. The faults with the highest score are considered the most probable to be the cause of the faulty behavior. A similar approach for stuck-at fault diagnosis is presented in [83]. The reduced fault dictionary is generated using a diagnostic fault simulator. This simulator creates a distinguishability matrix for the fault list simulated and calculates a mismatch score for each fault. The mismatch score of a fault is incremented with 2 when the faulty circuit has a known 0 or 1 value at the primary output which differs from the value obtained by simulating the fault. The faults with the lowest mismatch scores are identified as candidate faults.

All the methods for stuck-at fault diagnosis presented until now are using the cause-effect analysis, thus are using a fault dictionary. The second technique, effect
3.4. BRIDGING FAULT DIAGNOSIS

-cause, which was mentioned in section 2.3.2, is also used for localizing stuck-at faults [22, 59]. The approach proposed by Abramovici [22] locates multiple stuck-at faults in combinational circuits. It uses a Deduction Algorithm to deduce the internal values of the circuit lines. Based on these values can be determined if a line is faulty or not. A similar approach is proposed in [59]. It analyses pairs of input vectors and uses forward propagation and backward implication to deduce the values of the circuit lines. If one line takes both values 0 and 1 it is considered fault-free, this line is considered not to be stuck-at 0 or 1. Because this method analyses pairs of vectors it can be also applied for diagnosis of delay faults.

The methods presented here characterize the general trend of stuck-at fault diagnosis from simple to complex matching algorithms. It is well known that the most failures in the CMOS technologies do not behave exactly like stuck-at faults. Therefore, the inclusion of increasingly more complicated matching algorithms is necessary to achieve more accurate and precise diagnosing results in spite of the fact that the fault model used is a very simple one.

3.4 Bridging fault diagnosis

Due to the increasing occurrence of defects that cause shorts in modern CMOS technologies and the low success of the traditional stuck-at fault diagnosis methods for locating bridging faults, starting in the late 80’s much of the attention in the diagnosis field has been spent in developing methods for locating bridging faults. One of the first diagnosis methods for localizing bridging faults was proposed in [84] and it only considered shorts between primary input and output lines. First approaches were using the existing stuck-at fault diagnosis infrastructure for locating bridging faults. A novel approach was proposed by Millman et al [31]. It uses the stuck-at fault signatures to create bridging fault signatures, called composite signatures. If the node A is bridged to node B then the composite signature of the bridge A@B (C_{AB}) is the union of the four stuck-at fault signatures associated with the bridged nodes (see Figure 3.13).

The matching algorithm is simple: the behavior observed when testing the faulty device has to be included in one of the bridging fault signatures. The methods suffers from imprecision, the average size of the suspect list is very large. Improvements are proposed by Lavo et al [85]: the composite signatures are created only for realistic bridges (the ones that are extracted from the layout) and a more complicated matching is applied. First a match restriction condition is applied when the composite signature is built. Any vector in a composite signature that detects the same valued stuck-at fault on both bridged nodes places the same value on both nodes. Such a vector cannot detect a bridging fault and will be eliminated from the composite sig-
nature. When using the notations introduced before, the match restriction condition can then be expressed as follows:

\[ U_{AB} = (S_{A0} \cap S_{B0}) \cup (S_{A1} \cap S_{B1}) \]

\[ C_{AB} = S_{A0} \cup S_{B0} \cup S_{A1} \cup S_{B1} - U_{AB}. \]

The second parameter introduced is the match requirement. When a vector from the composite signature detects both A stuck-at 0 and B stuck-at 1 or A stuck-at 1 and B stuck-at 0 it is marked as required vector and will be used later in the matching algorithm. Again if the previous notations are used the match requirement condition can now be expressed as follows:

\[ R_{AB} = (S_{A0} \cap S_{B1}) \cup (S_{A1} \cap S_{B0}) \]

The matching algorithm introduced in [85] and described below is not as strict as in the original technique proposed in [31].

Figure 3.14 shows the comparison of the observed behavior (B) and the candidate composite signature (C). The observed behavior which is also included in the candidate composite signature is called intersection (I), the vectors which are predicted by the candidates signature and are not observed are called mispredictions (M), and the output vectors which are observed but not predicted by the candidates composite signature are called nonpredictions (N). The ranking criterion is the size of intersection (I), a candidate that contains more observed vectors in his composite signature is considered superior to other that contains a smaller number. If two candidates have the same value for I then the number of required vectors contained in the intersection is used to decide the best candidate. If still there is a tie then the value of the misprediction (M) is used to break ties of the two metrics, the candidate with the smallest M is considered the best.

Another method that uses the stuck-at fault dictionary, a bridging fault model and
a complex matching algorithm is described in [32]. The diagnosis method analyses only the faulty vectors (tests that detect the fault) and with the help of a stuck-at fault simulator builds the list of stuck-at faults that can be detected by these vectors. Also the list of circuit lines from which there is a path to the failing outputs is built. All the feedback and non-feedback bridging faults, which can be created by the circuit lines previously determined, are considered. Then the inconsistent and inactive bridging faults are dropped. A bridging fault is considered inactive when the test sets both lines at the same logic value. The bridging faults are considered inconsistent when the computed possible faulty response of the device containing the bridging fault is different than the one obtained from the tester. For computing the bridging fault behavior the voting model (described in the section 3.2.2 [69]) is used. This approach also suffers from imprecision because the average size of the suspect list is very large. A similar approach is proposed in [25]. It uses single stuck-at fault and logic simulation of the failing vectors to identify candidate fault locations. The bridges considered are between circuit lines which have opposite values for the failing vectors (as determined from the logic simulation). If none of the circuit nodes suspected to have a bridging fault fails as a stuck-at fault for the failing vectors, the bridging fault is eliminated from the list. No bridging fault model is used and also no ranking mechanism for ordering the candidates. The average size of the candidate list is smaller than for the method proposed in [32] but still large.

The advantages of using the existing stuck-at fault diagnosis infrastructure when
locating bridging faults are exploited in the method proposed by Stanojevic [26].
First, stuck-at fault diagnosis is performed to obtain a list of suspect nets. These
nets are then matched against a bridging list obtained from the layout in order to
identify the potential bridging faults associated with a given net. The bridging faults
are then simulated using the wired-AND, wired-OR and Wired model. The Wired
model propagates an X on the bridged nodes when they are driven to opposite logic
values. Calculating the Hamming distance between the simulation results and the
tester output then ranks the bridging fault candidates. For the multiple bridging
fault models, the smallest Hamming distance and bridging fault model is recorded.
A good diagnosis precision is obtained at the cost of extra simulation of the three
bridging fault models.

Diagnosing bridging faults with stuck-at fault information can often lead to a large
list of candidates and to a high number of misleading results, when none of the shorted
nodes is present in the candidate list. To address these deficiencies, the diagnosis
methods proposed in [86, 30] are using the fault dictionaries built by simulating
bridging faults. The average size of the candidate list is relatively small (under ten
as reported in [30]) and contains very few misleading diagnoses. However, the costs
of obtaining the bridging fault dictionary are much higher than for the stuck-at fault.
In a circuit there are more bridging faults that stuck-at faults and the bridging fault
model requires validation and refinement to accurately describe the actual behavior
of such a bridging fault.

A method for diagnosing bridging faults that does not require fault dictionaries
and/or complex bridging fault models is proposed by Venkataraman and Fuchs in
[23]. The method is based on the effect-cause analysis and consists of two deduc-
tive procedures. The first procedure identifies the lines potentially associated with a
bridging fault. This is done by means of a path-tracing from the failing outputs which
uses the values obtained by logic simulation. This information is then combined using
an intersection graph that is processed dynamically. Heuristics are proposed to reduce
the list of candidate faults but the average size remains quite large (in the experiments
performed by the authors the average size is mainly around 20).

All the bridging faults diagnosis methods presented in this section are analyzing
the voltage test results. Beside these methods, extended research has been done for
developing methods for locating bridging faults by using the Iddq test results, so
called Iddq diagnosis [87, 88]. The advantages of using Iddq diagnosis are that the
fault signatures are easy to construct and the diagnosis results are usually precise and
accurate. The disadvantages are that not all the devices will be Iddq testable and with
the decreasing feature sizes, the background current levels are increasingly making it
difficult to distinguish between good and faulty current levels. A proposed solution
to this problem is the concept of current signature [89, 90] which uses the difference
between the current levels to identify whether a defect is present in the circuit or not.
3.5 Open fault diagnosis

Opens in the conducting layer of an integrated circuit are next to bridging faults a common defect for the today's CMOS ICs with six or more metal interconnect layers [74]. While the problem of diagnosis of bridging faults has been intensively investigated, as shown in the previous section, little research has been made regarding the diagnosis of open defects. A method for diagnosing interconnect open faults has been proposed in [27]. The authors developed a diagnostic fault model (Figure 3.15) for capturing the potential faulty behaviors in the presence of an open defect on the interconnect layers. The logic net ABC is simulated and the erroneous responses caused by a 0/1 error at location A, B and C are recorded in the erroneous observation sets (EO) EO1, EO3 and EO5 respectively. Similarly, the 1/0 errors at location are captured in the sets EO2, EO4 and EO6 respectively. The diagnostic signature EO for the node A is the union of the sets EO1, EO2, EO3, EO4, EO5 and EO6. In the presence of an open only a subset of the branches is disconnected and causes faulty responses. The authors are using a path-tracing procedure along the sensitized path for each failing vector to identify the logic nets that can be potentially associated with the open defect. These nets are then simulated and the diagnostic signatures (EO) are computed. In addition to the diagnostic model the ranking method proposed in [75] is used to select the best candidate.

![Open diagnostic fault model](image)

Figure 3.15: The open diagnostic fault model

Another method for diagnosis of open faults [76] is focused on locating gates with tunneling opens. The method proposed consists of two steps. First, a commercial stuck-at fault diagnosis tool to generate an initial list of suspects uses the VLV (Very Low Voltage) test results. In the second step, Iddq(t) drifts results are used to reduce the initial suspect list. Most of the open faults as was shown in paragraph 3.2.3 are introducing extra delay or sequential behavior in the functionality of a device [44].
These faults are mainly detected with delay tests and considered as delay faults, hence the diagnosis methods described in the following section can be applied to localize them.

3.6 Delay fault diagnosis

For today’s high performance ICs it is very important that the timing-related defects which are causing the delay faults are detected and identified so corrections can be made in the design or in the manufacturing process. Therefore, in the last years testing and diagnosis of the delay faults has received an increasing interest from both academia and industry. In section 3.2.4, several delay fault models used for test pattern generation and diagnosis are presented.

One of the first approaches for delay fault diagnosis was based on the transition fault model [59]. The method performs a forward propagation of the input test vectors to determine the values for each signal line in the circuit under any single or multiple faults. This is followed by a backward implication of the values observed at primary outputs to deduce the values carried by each line. The deduced line values are used to determine if the line is fault free, has a fault or may have a fault. This method can handle multiple transition faults but it does not take into account the delay fault size and the list of suspect fault location is relatively large. Recent research [60] use the assumption of pattern dependence when diagnosing transition faults in order to obtain a higher diagnosis resolution.

To diagnose delay faults, the authors of [61] and [62] use the gate delay fault model. The approach identifies the probable fault location by using fault simulation of each failing pattern combined with path tracing from the failing primary outputs. In [61], two-valued logic simulation is used which does not take in consideration the delay faults caused by static hazards on the signal lines. In order to improve the diagnosis method, a six-valued symbolic simulation is used in [62]. The backtracking procedure identifies all the lines that can have a transition under the test. This procedure leads to a large list of fault locations that always will contain the correct one. To improve the accuracy of the diagnosis method backtracking only along sensitive lines (lines which can propagate the fault effect) is performed.

In the last years, the path delay fault model was more frequently used for delay fault diagnosis [24, 63, 64, 65, 66]. The authors of [24, 63] apply the effect-cause analysis to diagnose path delay faults. They use a five-valued logic to represent the state of a line (steady or transition) and only the lines with transitions (with or without hazards) can have excessive delays. For all the failing test patterns the set of suspect paths is built by backtracing. This set is relatively high and in order to reduce the simulation time only simulation of the passing test patterns is performed.
37. OTHER APPROACHES

To determine which paths are robustly tested (a path is robustly tested by a pattern if this pattern can detect the delay fault on this path independently of delays on other paths). If paths that are robustly tested by the passing test pattern appear in the suspect list, they are eliminated, which will improve the diagnostic resolution. Other directions for improving the path delay fault diagnosis proposed [64, 65] is to generate an explicit test for each suspect delay fault, if such a test is possible, in order to distinguish them.

A delay fault diagnosis method that identifies the likely subpaths that could have caused the delay failure and also pinpoints at the process parameter whose variation could cause the delay is presented in [66]. Monte Carlo techniques were used to generate fabrication process parameter combinations and the influence on the delay of each gate was determined. For each combination, the time when the outputs stabilize to their final value was calculated. This information was used to pinpoint the likely cause of the delay fault when a failing test result was analyzed.

3.7 Other approaches

The fault model used or the matching algorithm implemented can not precisely categorize several diagnosis approaches for fault diagnosis as is done in the previous sections. Some methods tried to eliminate or minimize fault simulation and used path-tracing and forward implication procedures to identify the fault-free circuit lines. They can implicitly diagnose multiple and various fault types, although the results obtained are often imprecise. Two of these approaches [22, 59] were presented previously in the sections related to the fault models that were given as examples by their authors.

Another diagnosis method that allows the localization of multiple faults of various types is presented in [20]. The method uses critical path tracing and the internal node logic values to eliminate candidate nodes. Then stuck-at fault simulation is performed for a reduced set of candidate faults. The simulation results are compared with the tester output and three different classes of defects are identified (according to how closely the defects behave as single stuck-at faults): SSF when a stuck-at faults explains all the falling and the non falling outputs for the falling patterns; non-SSF/single-site defects, a stuck-at faults explains all the falling outputs but is expected to cause additional failures; and non-SSF/multiple site defects when there is more than one stuck-at fault necessary to explain all the failing outputs of the failing test patterns.

A fault diagnosis technique that aims at capturing the unmodeled behavior is proposed in [35]. The effects of all possible behaviors that can be generated from a specific node are stored in the "X-list" which are then simulated. The simulation results are then compared with the tester output and a matching algorithm with three
CHAPTER 3. ELECTRICAL FAULT DIAGNOSIS

ranking counts is used to create the list of candidate faults. The method was applied on small circuits and good results were obtained. For larger circuits the simulation of the "X-list" will become prohibitive which will affect the performance of the proposed diagnosis method.

The fault diagnosis method developed by Ventakaraman and Drummonds [34] is able to locate multiple and more complex faults than the stuck-at fault. It uses the stuck-at fault simulation results combined with diagnosis fault models in order to locate stuck-at, bridging and interconnect open faults. Not all the circuit signal nets are simulated, a path-tracing algorithm based on critical path tracing starting at the failing outputs (flip-flops) is applied to create a small list of suspects. The simulation results are then used to construct the diagnostic fault models that are based on the notion of composite signatures. A composite signature is the union of the signatures of two or more stuck-at fault signatures. Four diagnostic models are built: the stuck-at, node, net and bridge model. The stuck-at model (Figure 3.16 a) contains the fault signature of a single stuck-at fault covers the short to power and ground and cell defects that fix the value of the cell’s output. The node model (Figure 3.16 b) covers opens on interconnect signal lines and cell defects that cause binary error of both polarities and is composed from two stuck-at fault signatures. The bridge model (Figure 3.16 d) is composed of the four stuck-at fault signatures of the shorted signal nets. The net model (Figure 3.16 c) is the union of the stuck-at faults on an interconnect’s stem and branches and cover the open faults on large interconnect with fan-out. All these diagnostic faults are called candidates and are ordered by using the three metrics proposed in [75].

More recently, Bartenstein et. al presented in [36] a diagnosis method which tries to locate defects in failing devices without referring to any fault model but concentrating on the location of the defect. They introduce the notion of logical defect that consist of the set of nets (or pins, as the authors call them) that can be affected by the defect. The logical defects as defined by the authors are not able to model in detail the behavior of the physical ones, they are suitable for logical simulators and test generators. Only a limited number of the failing patterns are analyzed, namely those that have a special property and are called SLAT patterns. A failing pattern has the SLAT property if all the observed failures for that pattern can be explained exactly by at least one single pin fault. Stuck-at fault simulation is used to determine the SLAT patterns and the pins which explain all the SLAT patterns will be collected together forming "multiplets". Usually, there are more multiplets which can explain all the SLAT pattern and thus an analysis is performed to determine the set of pins (called splats) such that each pin in the set belongs to a multiplet but no two pins from the same set (splat) belong to the same multiplet. Given the fact that not all the failing patterns are analyzed, only a subset has the SLAT property, the diagnostic precision is a matter of concern.
3.8 Conclusions

In this chapter, an overview on the electrical diagnosis methods is presented. The interest on software/electrical based localization methods has risen first to assist the guided probe techniques. More recent developments in test methods and fault models are also incorporated in the diagnosis methods.

In the beginning the diagnosis methods were based on simple fault models (single stuck-at fault) and were implementing simple matching algorithms. Later, with the introduction of more complex fault models (bridging faults, delay faults) these models were introduced in diagnosis techniques and the matching algorithms are also becoming more complex. Moreover, techniques for localizing a large spectrum of faults without focusing on one particular fault model were proposed. The simple methods suffer from imprecision, the computational effort necessary for the more complex ones it not entirely paid back from the results obtained (the suspect list is often too large).

There is a long way ahead to the ideal localization technique where in all cases only one suspect is pinpointed, which is then also the correct one. New IC process technologies introduce new defect types with different behavior as the ones we have now. Therefore, there is a continuous need for improving the diagnosis techniques so
they can keep up with the requirements of this very dynamic industry.
Chapter 4

Fault diagnosis based on the stuck-at fault model

4.1 Introduction

The stuck-at fault model, due to its simplicity, is widely used not only for test pattern generation but also for fault diagnosis purposes. The drawback of using a stuck-at based diagnosis approach is that the stuck-at model does not accurately describe the behavior of the most common types of defects that are present in today’s ICs, like bridges and opens. Building a complex and realistic fault model is a complicated task. For several types of defects, like for example feedback bridging faults or opens at transistor level, there is no accurate fault model developed. It seems that modeling every possible defect that can occur in a circuit is a mission impossible. Nevertheless, the new processes that are under development for sure will bring new types of defects and the fault modeling development process may not be able to keep up with the requirements. Therefore, in this chapter a new method which improves the existing diagnosis infrastructure that is based on the stuck-at fault model will be presented. The method uses stuck-at fault simulation combined with other fault models (bridging, open, and delay) to locate multiple faults in defective devices.

The simplified flow of the diagnosis procedure is presented in Figure 4.1. The first step is the identification of the faulty nets followed by a mapping with a fault model. A matching mechanism based on two ranking counts is used to order the list of candidates. If not all the failing vectors are explained by the best candidate, the explained failing vectors are eliminated and the procedure is started again for the remaining vectors. The diagnosis procedure stops when all the failing vectors logged on the tester are explained. The fault types that can be located are stuck-at, bridging
and interconnect open faults. All the steps of the diagnosis procedure will be described in the following sections. The method for diagnosing delay faults using stuck-at fault simulation is very similar with the one for locating stuck-at, bridging and opens and will be described in a separate section. Finally, experimental results and conclusions will be presented.

4.2 Identification of the faulty net

A defect is considered to be an imperfection occurred during the manufacturing process of an IC. Examples of defects are contaminants, extra or missing material, misalignment of the equipment relative to the wafer being processed, etc. The defects can be located inside the cells or at the interconnect level. These defects can affect the functionality of one or more signal lines (also called nets). For example an open in a via will affect the functionality of one net where an extra particle which creates a short between two nets can affect the functionality of both nets. Even when the defect is located inside the cell affecting one of the transistors, the malfunction can be observed at the output, since the net or nets (if the cell has more than one output) driven by this cell will be faulty. Based on this observation, the first step of the diagnosis procedure is to identify the nets whose functionality is affected by a defect present in the circuit. These nets are called faulty nets.
4.2. IDENTIFICATION OF THE FAULTY NET

When a test pattern is applied, a faulty net may produce two types of faulty logic values: a faulty logic 1 value when the good one is 0 and respectively, a faulty logic 0 value when the good one is 1. In order to detect this faulty behavior, the faulty value has to be propagated to an observable output (primary output or a scan chain flip-flop). This propagation condition is satisfied when testing for a stuck-at fault on that net, a stuck-at 0 when the faulty value is 0 or a stuck-at 1 when the faulty value is 1. However, a faulty net can take both faulty logic values under a set of test patterns, when some of the test patterns are applied it will behave as a stuck-at 0, when others are applied like a stuck-at 1. Therefore, all faulty behaviors of a net are captured using stuck-at fault simulation.

![Diagram](attachment:diagram.png)

Figure 4.2: The identification procedure flow

The procedure of the identification of the faulty nets consists of three steps (Figure 4.2). First a back coning procedure is applied to obtain a list of suspect nets. Considering all the nets from today’s complex ICs is not practical. Therefore, only a subset
will be processed further. For each output/flip-flop that failed on the tester the back coning procedure identifies all the nets from which there is a direct path to that output/flip-flop, each fanout branch of a net is considered separately from the stem. This set of nets is also called the logic cone of that output/flip-flop (Figure 4.3).

![Figure 4.3: Graphical representation of a logic cone](image)

If the single fault assumption is used (it is considered that only one fault is present at a time in a faulty device), then the list of nets obtained by intersecting all the logic cones of the failing outputs/flip-flops can be considered further. When there are more defects present in a device and they are located in distinctive logic cones, the list of nets obtained after intersecting the logic cones of the failing outputs/flip-flops will be empty. Therefore, in order to take also this situation into account, a union of the logic cones is constructed and this will form the list of nets which will be processed next.

The second step in identifying the faulty nets is the stuck-at fault simulation. All the stuck-at faults associated with the nets from the reduced list obtained in the first step are simulated. For each stuck-at fault a variable called suspect is allocated. When simulating a stuck-at fault in case an output is failing both on the tester and in the simulation, the value of the suspect variable allocated to that stuck-at fault is incremented. The stuck-at faults with \( \text{suspect} = 0 \) are dropped from the list and only the simulation results of the other stuck-at faults are stored in a fault dictionary. The nets whose stuck-at fault signatures are stored in the dictionary are considered suspect faulty nets. The suspect faulty net signature can be composed from both stuck-at 0 and stuck-at 1 fault signatures (similar with the net diagnostic model proposed in [34]), or only one of them, stuck-at 0 or stuck-at 1, depending on the value of the variable suspect.

The structure of the fault dictionary is simple (Figure 4.4). For each suspect faulty net first the stuck-at 0 simulation results are stored if the value of the variable suspect allocated to the stuck at fault is \( > 0 \). If \( \text{suspect} = 0 \) than a \( 0 \) is recorded. Next the stuck-at 1 simulation results are stored only if the value of its allocated variable
4.2. IDENTIFICATION OF THE FAULTY NET

<table>
<thead>
<tr>
<th>Fault</th>
<th>Pattern/pin</th>
<th>Pattern/pin</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Net A O</td>
<td>SA0 1/1</td>
<td>1/5</td>
<td>3/1</td>
</tr>
<tr>
<td></td>
<td>SA1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Net B O</td>
<td>SA0 3/2</td>
<td>3/3</td>
<td>20/2</td>
</tr>
<tr>
<td></td>
<td>SA1 1/3</td>
<td>4/2</td>
<td>6/2</td>
</tr>
<tr>
<td>3. Net B I at G1</td>
<td>SA0 3/2</td>
<td></td>
<td>20/2</td>
</tr>
<tr>
<td></td>
<td>SA1 4/2</td>
<td></td>
<td>50/2</td>
</tr>
<tr>
<td>4. Net B I at G3</td>
<td>SA0 3/3</td>
<td></td>
<td>35/3</td>
</tr>
<tr>
<td></td>
<td>SA1 1/3</td>
<td></td>
<td>56/3</td>
</tr>
<tr>
<td>5. Net C O</td>
<td>SA0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA1 1/7</td>
<td>1/8</td>
<td>18/7</td>
</tr>
</tbody>
</table>

Increasing number of test pattern

Figure 4.4: The structure of the fault dictionary

suspect > 0, otherwise a 0 is stored. The signatures of the faulty nets are stored in the form of the pattern/output pair, also called vectors, which detects one of the stuck-at faults associated with the net, in the increasing order of the pattern. As mentioned earlier, the fanout branches of the net are considered separately from the stem and their fault signatures are stored separately. An O follows the net name when the stem is considered and an I plus the name of the gate whose input is the net when a fanout branch is considered. For example in Figure 4.4 for the net A and C only one stuck-at fault signature is stored so a "0" follows the SA1 for net A and respectively, the SA0 for net C. For net B the signature of the stem (position 2) and that of two of its fanout branches (position 3 and 4) are stored. Net A has no fanout therefore only the signature of the stem is in the fault dictionary.

The goal of any diagnosis method is to determine which from the suspect faults is most probable to be the cause of the observed failure. For this purpose a matching algorithm is used to order the candidates. The matching algorithm compares the modeled/expected behavior of the fault, in our case the fault signature stored in the fault dictionary, with the observed behavior, the response obtained on the tester.

Let \( S = \{v_1, v_2, \ldots, v_n\} \) be the faulty net signature as stored in the fault dictionary which is composed from \( n \) vectors. \( T = \{t_1, t_2, \ldots, t_f\} \) is the set of the failing vectors which are observed on the tester. These two sets are graphical represented in Figure 4.5. For each faulty net the intersection of these two sets of vectors \( I = S \cap T = \{i_1, i_2, \ldots, i_r\} \) is calculated and let's consider that is composed from \( r \) vectors. Two ranking counts called matching (M) and prediction (P) are computed for each faulty
net and used to order the candidates. The ranking counts are defined as follows:

\[
Matching(M) = \frac{\text{Intersection}(I)}{\text{Tester Output}(T)} \times 100
\]

\[
Prediction(P) = \frac{\text{Intersection}(I)}{\text{Fault Signature}(S)} \times 100
\]

![Figure 4.5: Graphical representation of the ranking counts](image)

Given a set of failing vectors logged on the tester, the diagnosis procedure tries to find a fault candidate that can explain all these failures. Therefore, the *matching* count is first used to order the candidates. A fault for which the computed value of *matching* \( M \) is bigger than that of the other ones is considered the best candidate. If the candidates have the same value for \( M \) than the second ranking count (*prediction*) is used to differentiate them.

For example in Figure 4.6 three candidates \( F_1, F_2, \) and \( F_3 \) are shown together with their fault signatures \( S_1, S_2 \) and respectively, \( S_3 \). The signatures \( S_1 \) and \( S_2 \) contain the same number of vectors while \( S_3 \) contains a larger number of vectors. If we assume that the behavior observed on the tester is \( T \), the intersections with the fault signatures of the three candidates are \( I_1, I_2 \) and \( I_3 \). The values of the ranking counts associated with the three fault candidates calculated using the relation (1) and (2) are \( M_1, M_2, M_3 \) and respectively \( P_1, P_2, \) and \( P_3 \). Assuming for example that \( I_1 = I_3 > I_2 \) than the following relations can be established between the values of matching and prediction corresponding to the three faults:

\[
M_1 = M_3 > M_2, \quad P_1 > P_3 \quad \text{and} \quad P_1 > P_2
\]

Based on these relations, on the output list of the diagnosis procedure the candidate \( F_1 \) will be on the top of the list (\( M_1 = M_3 \) and \( P_1 > P_3 \)), followed by the \( F_3(M_3 > M_2) \), and the last one is \( F_2 \).
4.3. MAPPING WITH A FAULT MODEL

\[ \begin{align*}
M_1 &= \frac{I_1}{T} \times 100 \\
M_2 &= \frac{I_2}{T} \times 100 \\
M_3 &= \frac{I_3}{T} \times 100 \\
\end{align*} \]

\[ \begin{align*}
P_1 &= \frac{I_1}{S_1} \times 100 \\
P_2 &= \frac{I_2}{S_2} \times 100 \\
P_3 &= \frac{I_3}{S_3} \times 100 \\
\end{align*} \]

\[ \begin{array}{ccc}
T & I_1 & S_1 \\
& & \\
F_1 & \ & \\
\end{array} \quad \begin{array}{ccc}
T & I_2 & S_2 \\
& & \\
F_2 & \ & \\
\end{array} \quad \begin{array}{ccc}
T & I_3 & S_3 \\
& & \\
F_3 & \ & \\
\end{array} \]

Figure 4.6: Example of a ranking procedure

4.3 Mapping with a fault model

In the previous section the procedure for the identification of the best faulty nets candidates is described. Only the name of the nets whose functionality is most probably affected by the defect is found. This information is not very precise in case physical analysis follows to find the root cause of the failure. There is no indication if the defect is located inside the cell or at the interconnect level of the circuit or what the type of the defect is. The nets can be long, spread over a large area in the die and tracing them through all the metal layers is not a trivial task. If it would be possible to add more information regarding the fault type associated with this faulty behavior, the localization will be more precise and will save a lot of physical analysis time. The faulty net notion is a generalization of the faulty behaviors that can occur under the influence of a defect, more general than the fault models described in chapter 3. It is only giving a hint from where to start, more analysis is needed if a precise localization result is wished. It has been shown [30] that using a more precise fault model increases the diagnosis precision if the actual defect is of the targeted type. For example, if we can determine that the faulty net behavior is due to a bridging fault, then the location of the defect can be more precisely indicated as being the metal segments of the two supposed shorted nets which are located close to each other in the layout.

In the research described here, three fault models are used for mapping the faulty net behavior: stuck-at, bridging and interconnect open fault models. Simulating all
the faults that can be associated with a faulty net is not practical for today’s complex IC’s. Therefore, for each fault model a fault signature is built by using only the faulty net signatures stored in the fault dictionary.

Given the values for matching and prediction computed earlier for the faulty nets the situations presented in Table 4.1 can occur:

<table>
<thead>
<tr>
<th>Matching</th>
<th>Prediction</th>
<th>Fault type</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>100</td>
<td>Stuck-at Fault</td>
</tr>
<tr>
<td>100</td>
<td>&lt;100</td>
<td>Bridging fault with another net, or an open on one of the net’s fanout branches</td>
</tr>
<tr>
<td>&lt;100</td>
<td>100</td>
<td>Bridging fault or multiple faults from each at least one is a stuck-at fault</td>
</tr>
<tr>
<td>&lt;100</td>
<td>&lt;100</td>
<td>Bridging fault or multiple faults</td>
</tr>
</tbody>
</table>

Table 4.1: Fault type identification

In the first case M is 100 and P is 100 which means that all the failures are explained by one faulty net, so there is only one defect present in the circuit. A 100 for the value of P shows that the modeled faulty net behavior is exactly the same as the observed behavior. Given the fact that the modeled faulty net behavior is obtained by stuck-at fault simulation, the fault can be a stuck-at fault. In the second situation M is 100 and P < 100. Similar with the situation encountered in the first case there will be only one defect in the circuit but the fault type can not be precisely modeled only using stuck-at fault simulation. In this case the fault can be a bridge or an interconnect open and more analysis is necessary to determine more precisely the fault type. When the value of M is < 100 and P = 100, which is the third possible situation, there is more than one faulty net which can explain all the failures observed on the tester. The fault can be a bridging fault, in this case the functionality of at least two nets will be affected, or there are multiple faults present from which one is a stuck-at fault (the value of P = 100). The last situation is when both M and P are less than 100. There are more faulty nets present in the circuit and the fault types that can be associated are bridging faults or multiple faults.

In all four situations presented here more analysis is done in order to determine the fault model which can be associated with the faulty net behavior. This mapping procedure is not applied to the complete list of the suspected faulty nets, but only for the candidates from the top of the list which have the ranking counts bigger that a certain limit. Because we only consider bridging faults between two nets, the mapping
4.3. MAPPING WITH A FAULT MODEL

procedure is applied only to the faulty nets whose value for matching is at least half from the value of the top suspect, as is also used in [92].

As mentioned before, the reason for performing the mapping with a fault model procedure is to locate more precisely the fault that is present in the circuit. For each of the three fault models that can be associated with the faulty net, a fault signature is built based only on the information stored in the fault dictionary. The matching and the prediction counts are computed and compared with the ones of the faulty net. If the fault model assumed is matching closer to the observed behavior, the matching will be at least equal with the one computed earlier and the prediction will be bigger than the one of the faulty net. In this case the candidate will be the fault whose model scored better compared to the faulty net. The fault models used in our research (stuck-at, bridging and interconnect opens) are targeting defects at the interconnect level, thus are not able to accurately describe all the possible effects of the presence of a physical defect as has been shown in chapter 3. Therefore, the situation when none of these fault models will give better ranking score than the score of the faulty net can be encountered. In this case, none of the fault models considered by the diagnosis procedure will be mapped on the faulty net. However, the faulty net will remain as a candidate and will be reported in the output list.

4.3.1 Mapping with the stuck-at fault model

The procedure for mapping with the stuck-at model is shown in Figure 4.7.

```plaintext
For each (suspect faulty net N)
{
    if ((SA0 == 0) or (SA1 == 0))
    {
        calculate M, P;
        if (P == 0)
        {
            map stuck-at fault on net N;
        }
    }
}
```

Figure 4.7: The procedure for mapping with the stuck-at fault model

The method for obtaining the faulty net signature that is stored in a fault dictionary is based on stuck-at fault simulation. For each suspect faulty net the fault
signature is composed from the simulation results of both the stick-at 0 and the stick-at 1 faults associated with the net or only from the stick-at 0 or stick-at 1, depending on the value of their suspect variable (as was described previously). When for a faulty net only one of the stick-at faults is stored (because the value of the variable suspect of the other stick-at fault associated with this net is 0) only this information will be used to compute the ranking counts. If calculating the prediction rank for this faulty net we obtain 100, it means that all the vectors that detect that stick-at fault resulted in the simulation are also observed on the tester. In this case the observed faulty net behavior is exactly the same as of a stick-at fault and consequently, the fault model mapped is the stick-at.

4.3.2 Mapping with the bridging fault model

The simplified flow of the procedure for mapping with the bridging fault model is shown in Figure 4.8.

For each (suspect faulty net N)
{
  extract corresponding bridging faults( bridge fault list);
  build the composite signature;
  calculate $M_{\text{bridge}}, P_{\text{bridge}}$
  if ($(M_{\text{bridge}} < M_N) \lor (M_{\text{bridge}} == M_N) \land (P_{\text{bridge}} > P_N)$)
  {
    map bridge fault on net N;
  }
}

Figure 4.8: The procedure for mapping with the bridging fault model

For each faulty net from the suspect list, the corresponding bridging faults will be extracted from a realistic bridge list and considered as suspect bridging faults. This realistic bridge list is obtained by analyzing the location of the nets in the layout. We developed a program for extracting the net pairs that are adjacent in the layout so that a short can occur between them. This program uses as input a file created after the routing and placement of all the component and signal nets in the layout is completed. The output is a list of possible bridging faults, which contains the names of the nets and the coordinates of the segments that can be the location of a short. For each of the suspect bridging faults a composite signature will be built. The method for building the composite signatures is based on the technique introduced by Millman et
4.3. Mapping with a Fault Model

This method uses a stuck-at fault and the voting model [69] to describe the bridging fault behavior. Under the voting model, when two nodes are shorted together and are trying to drive to opposite values, the transistor network of these two nodes will each attempt to assert competing logic values to the bridge. The stronger network wins this competition, or vote, and asserts its logic value to the bridge.

Assuming that a test vector \( v \) detects a CMOS bridging fault \( A@B \), then the two bridged nodes have opposite fault-free logic values for this vector. One of the nodes will be outvoted by the stronger one and will be driven to a faulty logic value. This faulty value is propagated to an observable output in order to have the detection conditions for the bridging fault satisfied. However, these are also the conditions to be satisfied by the pattern \( p \) in order to be able to detect a stuck-at fault on the outvoted node. This is actually the key observation of this diagnosis technique: if a vector \( v \) detects a bridging fault, it also detects the stuck-at fault on the outvoted node. Therefore, \( v \) must be in the list of vectors that detects this stuck-at fault on the outvoted node. A more generalized form of this observation, which is the key for this approach, is the following: if a vector detects a bridging fault, then it must also detect at least one of the four stuck-at faults associated with the shorted nodes.

The complete list of vectors that detect a stuck-at fault is in fact the stuck-at fault signature stored in the fault dictionary. Based on this information, the basis of this diagnosis technique is the construction of composite signatures for each potential bridging fault. The composite signature of a bridging fault is the union of the four stuck-at faults associated with the shorted nodes.

Considering that the nodes \( A \) and \( B \) are the shorted nodes, the four stuck-at faults associated are \( A \) stuck-at 0, \( A \) stuck-at 1, \( B \) stuck-at 0 and \( B \) stuck-at 1 and the corresponding fault signatures are \( S_{A0}, S_{A1}, S_{B0}, S_{B1} \). The composite signature of the bridging fault \( A@B \) (Figure 4.9) will be:

\[
C_{A@B} = S_{A0} \cup S_{B0} \cup S_{A1} \cup S_{B1}.
\]

Figure 4.9: The graphical representation of a composite signature

The construction of the composite signature is demonstrated in the examples from Table 4.2 and 4.3. Table 4.2 contains the stuck-at fault signatures of three nets \( A \), \( B \), and \( C \) and table 4.3 contains the composite signatures of the three possible bridging faults that can occur between these nets.

In the original method [31], after all the composite signatures are built, they are compared with the observed behavior to determine the fault candidates. The algorithm used is simple, the fault signature of a bridging fault must be contained
60

CHAPTER 4. FAULT DIAGNOSIS BASED ON THE STUCK-AT FAULT MODEL

<table>
<thead>
<tr>
<th>Fault</th>
<th>Signature (Pattern/Bit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-SA0</td>
<td>1/3 3/1 7/4</td>
</tr>
<tr>
<td>A-SA1</td>
<td>2/4 6/4</td>
</tr>
<tr>
<td>B-SA0</td>
<td>1/1 2/5 6/3</td>
</tr>
<tr>
<td>B-SA1</td>
<td>4/3 5/3</td>
</tr>
<tr>
<td>C-SA0</td>
<td>2/2 5/4</td>
</tr>
<tr>
<td>C-SA1</td>
<td>3/2 6/4</td>
</tr>
</tbody>
</table>

Table 4.2: Examples of stuck-at fault signatures

<table>
<thead>
<tr>
<th>Fault</th>
<th>Composite Signature (Pattern/Bit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A@B</td>
<td>1/1 2/4 2/5 3/1 4/3 5/3 6/3 6/4 7/4</td>
</tr>
<tr>
<td>A@C</td>
<td>1/1 2/2 2/4 3/1 3/2 5/4 6/4 7/4</td>
</tr>
<tr>
<td>B@C</td>
<td>1/1 2/2 2/5 3/2 4/3 5/3 5/4 6/3 6/4</td>
</tr>
</tbody>
</table>

Table 4.3: Examples of composite signatures

in its composite signature. Not all the patterns that detect one of the four stuck-at faults will also detect the bridging fault. If a vector detects the fault A stuck-at 0 and drives the net B to 1, that vector will not detect the bridging fault A@B because both nodes are driven to the same logic value (1 in this case). Nevertheless, this vector will be included in the composite signature of the bridging fault. The composite signature is over-predicting the bridging fault behavior. Because of this the candidate list is very large (as resulted from the experiments performed in [85]) and there is a high number of incorrect diagnosis results (the candidate list does not contain the correct match).

Improvements for reducing the candidate list size were proposed in [85]. One improvement is to consider only the bridging faults determined to be realistic through analysis of the layout. The other is to apply a more complicated matching algorithm. Match restriction (only the vectors which place opposite logic values on the shorted nets are considered) and match requirement (the vectors which detect both shorted nets as a stuck-at fault are marked as required) are imposed when the composite signatures are built. Also a more complicated matching algorithm including three ranking counts is proposed. A more detailed description of these improvements has been done in chapter 3.4. In the experiments we performed [93] on bridging faults injected with a FIB in good dies, in 50% of the cases we obtained an incorrect diagnosis result (only one of the correct shorted nodes was pinpointed). If physical analysis follows, it is imperative that the location of the fault is correctly pinpointed because it can be destroyed when looking at another point. Therefore we need to improve the
4.3. MAPPING WITH A FAULT MODEL

precision of the diagnosis method. The improvement we proposed to achieve a higher
diagnosis precision is to use a bridging fault model when building the composite
signature. Moreover, the stuck-at fault signatures we consider are only the signatures
of the faulty nets that we stored in the first step of the diagnosis procedure. The
bridging fault models used are wired-AND and line dominant. From the analysis we
performed on library cells used by some CMOS processes and from the results of
the analysis done on some microprocessors and presented in [94], it is shown that
these two models describe the behavior of the majority of bridging defects in the
CMOS circuits. The wired-AND model assumes that the value on both shorted nets
will be equal with the result of an AND operation between the values on the nets.
The line-dominant assumes that the transistor network of the cell which drives one
of the nets is always stronger then the other one and will always win the vote. The
net with the stronger driver will always determine the logic value on the shorted nets.
These two fault models are applied according with the strength of the drivers of the
shorted nets. The information regarding the driver strength is included in the file used
for extracting the realistic bridging faults and is stored for each bridging fault. We
performed simulations to determine the bridge behavior for several combinations of
driver strengths. The results show that when the driver strengths have close values the
bridges behave sometimes as wired-AND and sometimes as line-dominant, therefore
both models are applied when building the composite signature. In case one driver
is at least twice stronger than the other, the line dominant model describes better
the bridge behavior and then this model will be applied when building the bridge
composite signature.

The procedure for building the composite signature for a suspect bridging fault is
now as follows:

- The stuck-at fault signatures for the shorted nets are extracted from the fault
dictionary. If one of the shorted nets is not a faulty net then a 0 is assigned for
each stuck-at fault signature associated with this net.

- The vectors that place the same logic value on the shorted nets are eliminated.

- The strength of the drivers of the two nets are compared and the bridge fault
model to be applied is determined.

- The composite signature is built according to the bridge fault model determined.
   If the wired-AND is applied, only the stuck-at 0 fault signatures of both nets will
   be included in the composite signature. If the line-dominant model is applied,
   the stuck-at signatures of the net with the "weakest" driver will be included in
   the composite signature.

Table 4.4 presents some examples of composite signatures build using the described
method. The stuck-at fault signatures considered are from the examples given in Table
4.2. The vector 1/1 places the same logic value on both A and B nets because it is testing for the same polarity stuck-at fault on both nets and it will not be included in any composite signature. The same observation holds for vector 6/4 for the nets A and C.

After the composite signatures for all the suspect bridging faults are built, the ranking counts matching and prediction are computed. These values are compared with the one obtained for the associated faulty net. If the values obtained are bigger than the previous one then the bridging fault is mapped over the faulty net and is considered to be most probably the cause of the faulty behavior observed. If we consider that the vector logged on the tester are 2/2, 3/2 and 5/4, the faulty net list will have as the best candidate the net C with M = 100 and P = 75. After mapping with the bridging model and calculating the ranking counts we obtain M = 100 and P = 100 and the bridging fault A@C will be mapped on the faulty net C.

4.3.3 Mapping with the interconnect open fault model

An interconnect open defect is a break in the interconnect wiring connecting a set of logic gates to their driver. If we consider the signal net N from Figure 4.10, an open can cause that one (O1) or more (O2, O3) of its fanout branches to be separated from the driver. The separated branches will have a floating logic value (as shown in section 3.2.3) which depends on the electrical and physical characteristics of the driver, the wiring and the environment around the defect. If a test set detects this defect, it induces faulty logic values on the separated branches and these values are propagated to an observable output. This means that the separated branches will be in the suspect faulty net list together with the stem. The fanout branches that are not affected will not be present.

The procedure for mapping with the open fault model is as follows (Figure 4.11):

The first step of the mapping procedure is to identify if the suspect faulty net is the stem or is only a fanout branch of a signal net in the circuit. This information is available from the back coning procedure and stored in the fault dictionary (see Figure 4.4). When the stem is considered the net is marked with an O, respectively, when one of the fanout branches is considered, net is marked with an I.

If the suspect faulty net to be processed is a fanout branch, it means that an
4.3. MAPPING WITH A FAULT MODEL

Figure 4.10: Examples of interconnect open defects

For each(suspect faulty net N )
{
    if(N == fanout branch)
    {
        map open fault on net N;
        eliminate stem from the suspect faulty net list;
    }
    else{
        search(suspect faulty net list);
        built fault signature of failing branches;
        calculate $M_{open}, P_{open}$;
        if ($P_{open} > P_N$)
        {
            map open fault on net N;
        }
        eliminate analyzed faulty nets branches from
        the suspect faulty net list;
    }
}

Figure 4.11: The procedure for mapping with the interconnect open fault model

open defect can cause that only this branch fails and the open model is mapped on
the faulty net. Because we already could map a fault model on this net, this suspect
faulty net will be eliminated from the list. The stem will be then also in the suspect faulty nets list with lower ranks. A fault on the stem should propagate also along the other fanout branches. The stem fault signature contains the vectors that propagate the faulty behavior along all the fanout branches. Therefore, the value of prediction of the stem will be smaller than the one of the separated fanout branch.

If the suspect faulty net to be processed is a stem, the procedure searches in the suspect faulty net list for all the branches of this net. The union of the fault signatures of the branches is built and used for calculating the matching and prediction. If the values obtained are bigger than the ones of the faulty net, an open fault is mapped on this net.

<table>
<thead>
<tr>
<th>Fault</th>
<th>Pattern/ Pin...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.Net N 0</td>
<td>SA0 3/2 3/3 20/2 20/3 35/3</td>
</tr>
<tr>
<td></td>
<td>SA1 1/3 4/2 6/2 50/2 56/3</td>
</tr>
<tr>
<td>2.Net N I at G1</td>
<td>SA0 3/2</td>
</tr>
<tr>
<td></td>
<td>SA1 4/2</td>
</tr>
<tr>
<td>3.Net N I at G2</td>
<td>SA0 3/3</td>
</tr>
<tr>
<td></td>
<td>SA1 1/3 56/3</td>
</tr>
</tbody>
</table>

Figure 4.12: Fault dictionary

For example let’s consider that the open O2 from Fig 4.10 is present in our circuit. Net N and the fanout branches feeding into gates G1 and G2 will be in the suspect faulty net list. Their fault signatures are shown in Figure 4.12. If we assume that the failing vectors observed on the tester are 1/3, 3/2, 3/3, 4/2, 56/3 the values for matching and prediction for our suspects are:

\[ M_{N_{-output}} = 100, \quad P_{N_{-output}} = 50; \]
\[ M_{N_{-lastG1}} = 40, \quad P_{N_{-lastG1}} = 100; \]
\[ M_{N_{-lastG2}} = 60, \quad P_{N_{-lastG2}} = 100; \]

The ordered list of suspect faulty nets will be:

1. Net N 0:  M = 100, P = 70
2. Net N I at G2: M = 57, P = 100
3. Net N I at G1: M = 43, P = 100

If we apply the mapping with the interconnect open model to this suspect faulty net list, the first candidate will be net N which is a stem. The search procedure will return the two failing fanout branches of net N which are also in the suspect faulty net list. The union of their fault signatures is 1/3, 3/2, 3/3, 4/2, 35/3, 50/2, 56/3.
and the new values for matching and prediction are $M_{open} = 100$ and $P_{open} = 100$. It can be easily observed that the value for matching is equal to the one of the stem, but the prediction increases significantly because the model we used is more close to the observed faulty behavior. In the output list of the diagnosis method an open fault will be mapped on net N and indicating which fanouts are failing and the fanouts will be eliminated as separate candidates.

### 4.4 The format of the output list

After the mapping procedure is completed, the ordered list of suspected fault candidates is given. The candidate can be a faulty net or a stuck-at, a bridging or an open fault if one of these fault models gave better scores than the associated faulty net. An example of an output list obtained after the complete run of the diagnosis procedure is shown in Figure 4.12. For each candidate, the output list contains the type of the fault (for example STUCK-AT 0 or 1, BRIDGE, OPEN or FAULTY NET), the name of the nets or nets affected by the defect and the values of the ranking counts (M and P).

To help the physical analysis, beside the names of the nets also information regarding the location in the layout can be given:

- For the stuck-at faults and faulty nets the output list can contain also the name and location of the driver together with the complete routing through all the metal layers.

- For the bridging faults the coordinates of the segments belonging to both suspected nets where the short is supposed to be located.

- For the open faults the coordinates of driver cell of the net suspected to have open, the routing of the net and the coordinates of the cells which have as input the failing branches.

| TOP 1: BRIDGE: dsp2.EPICS7.MAU_E70.MULT_PART_E70.N768, dsp2.EPICS7.MAU_E70.MULT_PART_E70.N2032 B fa1 bf1tx4 M = 100.00% P = 100.00% |
| TOP 2: FAULTY NET: dsp2.EPICS7.MAU_E70.MULT_PART_E70.N956 M = 100.00% P = 28.00% |
| TOP 3: FAULTY NET: dsp2.EPICS7.MAU_E70.MULT_PART_E70.N1256 M = 87.00% P = 17.00% |

Figure 4.13: An example of an output list
4.5 Delay faults diagnosis

Defects and/or random variations in process parameters often cause propagation delays to fall outside the desired limit, causing delay faults. In order to test a delay fault, a scan-based test consists of two patterns. The first pattern initializes the node or the path being tested and the second is generating a transition on that node or along that path. To detect a delay fault, the transition has to be propagated to an observable output. If a delay fault is detected by a pair of test patterns, the tester will record at the end of the propagation path (primary output/scan flip-flop) a different logic value from the one it is expected (see Figure 4.14).

![Diagram](image.png)

Figure 4.14: Example of the delay fault detection

Several fault models were developed for modeling the behavior of a delay fault and a detailed description can be found in chapter 3.2.4. For our diagnosis research, the transition fault model has been chosen. Under this model a defect that delays a rising or a falling transition at the inputs or at the output of a logic gate is considered to cause a slow-to-rise or slow-to-fall transition fault. The slow-to-rise transition fault temporary behaves as a stuck-at 0 fault (the net keeps its logic 0 value longer than is expected) respectively, the slow-to-fall behaves like a stuck-at 1. Based on this observation it seems possible to use the stuck-at diagnosis infrastructure for locating delay faults. Another advantage of this model is that the number of faults is linear in terms of number of gates present in the circuit. These two reasons were at the base of the decision of choosing the transition fault model for diagnosing delay faults.
4.5. DELAY FAULTS DIAGNOSIS

The initialization conditions for testing for transition faults are different from the ones applied when testing for a stuck-at fault. Nevertheless, the second delay test pattern has to fulfill the same requirement concerning fault effect propagation to an observable output as has a stuck-at pattern. For the example shown in Figure 4.14 the second pattern has to propagate the faulty value 1 at the output of the inverter to an observable output, the same as for a stuck-at 1 fault. Based on this observation we split the pair of input patterns and we consider the initialization and the propagation patterns in two separate sets. The condition necessary to be fulfilled for detecting a transition fault can be interpreted as follows:

- the second pattern has to be able to detect a stuck-at fault at the input/output of a gate.

- the first pattern has to set the logic value of the gates input/output at the same polarity as the stuck-at fault tested by the second pattern (a 0 for a stuck-at 0 and a 1 for a stuck-at 1).

The diagnosis of transition faults can now be resumed to the identification of the faulty nets detected by the second pattern and verification if the activation condition is true, namely the first pattern sets the proper logic value on the faulty net.

The diagnosis flow for locating transition faults is similar with the one from identification of faulty nets described in chapter 4.2 (Figure 4.15). The difference consists in the introduction of the verification step before the matching procedure.

The backconing procedure will identify all the logic cones of the failing outputs. The union of the logic cones is performed to obtain the list of faulty net candidates. Stuck-at fault simulation is then performed to identify the suspect faulty nets and their fault signature will be stored in a fault dictionary. The next step when diagnosing delay faults is the verification of the activation condition. For each vector (pattern/output pair) stored in the fault signature of a suspect faulty net the procedure verifies if the corresponding vector from the first set (the initialization test patterns) places the faulty net to a 0 if the vector is supposed to detect a stuck-at 0 on this net, respectively to 1 if a stuck-at 1 should be detected. The values placed by the test patterns from the first set on all the signal nets from the circuit are obtained by performing a "golden-simulation" of the circuit. All the signal nets values are stored in a binary file that can be easily accessed by the verification procedure. The vectors that do not satisfy the activation condition are eliminated from the fault signature. The remaining vectors will form now the fault signature of the delay fault that is suspected to affect that faulty net. If all the vectors of a suspect faulty net are eliminated, the faulty net is also eliminated from the list. The ranking mechanism is now applied to the new fault signature obtained after the verification step. The two counts matching and prediction are computed and they are used to order the list of the remaining suspect faulty nets.
First the matching is used to determine the best candidate, and if there is a tie, prediction is used to break it.

When applying the diagnosis method described before it was assumed that the two-pattern set applied detects a delay fault, independent of the delays that can be present in the circuit. The hazards that can occur (for example the situation in Figure 4.16) can invalidate a test, if the tester "samples" the output at the moment T. The good logic value will be read even though there are two delay faults present in the circuit. The presence of the hazards was not specifically taken into consideration by the diagnosis method. If a hazard invalidates a test pattern pair which according to the algorithm should detect a delay fault, the value of prediction calculated for that fault will be lower than in the case when no hazards occur because the test pattern
4.6 Experimental results

To verify the effectiveness of the diagnosis methods described previously, a set of controlled experiments and analysis of samples with real defects was performed.

In the controlled experiment, faults were injected via a FIB (Focused Ion Beam) machine in good devices. These devices originally passed all the tests developed for this IC. The IC has 35K gates implemented in a 0.35μm technology. It is a mixed-signal design with full-scan implemented in the digital part. The test pattern set used is a compacted scan-test set generated for the manufacturing test.

Single and double faults of three different types were injected as follows: in three samples, bridging faults between two signal nets were created; one sample contains a bridging fault between three signal lines; in one sample an open was injected; two samples have each two bridging faults and three samples contain each a stuck-at fault and a bridging fault. Figure 4.17 shows the SEM picture of a bridging fault injected in one of the samples. These types of experiments were performed to verify the effectiveness of the proposed method when dealing with single and multiple faults of different types.

Table 4.5 presents an overview of the faults injected and the results obtained after applying the diagnosis method. In the second column the number of test vectors which failed on the tester is given followed in the third column by the type of the fault injected in the corresponding sample. The notation A@B from the second column indicates that a short was injected between nets A and B.

The diagnosis results are presented in the columns 4 to 8. The results after the
first run are presented in columns 4 and 5 and the results after the second run of the procedure, necessary when not all the failures were explained in the first run, are presented in the columns 7 and 8. The notation used in these columns, for example 1 - M82, P100 indicates the position of the fault in the ordered list (e.g. 1) and the matching (M) and prediction (P) values calculated. In column 4 the results obtained after completing the faulty net identification step are presented. The name of the net and the value of M and P for the candidate with the best rank are shown. For almost all the samples, one of the nets whose functionality is affected by the injected defect is correctly identified as the best match. The value of matching for the faulty net identified as best match for all the samples, except sample 4, indicates that there are more nets affected by the defect. The final results confirm this observation.

The results after performing the mapping with a fault model are shown in column five. The best match obtained after the reordering was the correct one for all samples with one fault injected. The algorithm identified as best match one of the faults injected in the other samples.

When the value of matching for the candidate with the highest rank was < 100, the failures explained were eliminated and the procedure restarted (second run). New ranking counts are computed for the faulty nets using the new failing information. Then the procedure for mapping with a fault model is executed.

Columns six and seven present the results after the identification of faulty nets
<table>
<thead>
<tr>
<th>Bridge Type</th>
<th>Result 1</th>
<th>Result 2</th>
<th>Result 3</th>
<th>Result 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Bridge</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
</tr>
<tr>
<td>Double Bridge</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
</tr>
<tr>
<td></td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
</tr>
<tr>
<td></td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
</tr>
<tr>
<td></td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
</tr>
<tr>
<td></td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
</tr>
<tr>
<td></td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
</tr>
<tr>
<td></td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
<td>1.0100</td>
</tr>
</tbody>
</table>

Table 4.5: The results of the controlled experiments
and respectively, mapping with a fault model procedures are executed for the new fail information. For samples 6-11 the fault candidate with the highest rank is also the correct one. For sample 5, the correct fault is on the second position in the final ordered list, one of the shorted nets indicated as the best match is net B. Not all the failures observed on the tester were explained for samples 5 and 6 even if the faults initially injected were identified by the diagnosis algorithm. The reason is that the fault models used by the diagnosis algorithm are not "perfect", they don’t take in consideration all the possible situations which can occur in practice (It has been shown in chapter 3 that fault model capable of capturing all the faulty behavior caused by for example a short or a open defect is very difficult to obtain). When diagnosing bridging faults only shorts between two signal nets were considered (in sample number 5 three nets are shorted together). Another assumption used by the diagnosis procedure is that a net is affected by only one defect, which is not true for sample 6 when one signal net is shorted with other two nets at different locations. Even with these restrictions, the diagnosis algorithm did always indicate as the best candidates the correct ones.

<table>
<thead>
<tr>
<th>Sample No</th>
<th>No. Failing Vectors</th>
<th>Faulty net identification (Rank- Match, Pred)</th>
<th>Fault Type Mapping (Rank- Match, Pred)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1 - M 100, P 2</td>
<td>Bridge 1 - M 100, P 50</td>
</tr>
<tr>
<td>2</td>
<td>97</td>
<td>1 - M 100, P 34</td>
<td>Bridge 1 - M 100, P 100</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>1 - M 100, P 18</td>
<td>Open 1 - M 100, P 100</td>
</tr>
<tr>
<td>4</td>
<td>127</td>
<td>1 - M 100, P 46</td>
<td>Bridge 1 - M 100, P 98</td>
</tr>
<tr>
<td>5</td>
<td>115</td>
<td>1 - M 100, P 40</td>
<td>Faulty net 1 - M 100, P 40</td>
</tr>
<tr>
<td>6</td>
<td>658</td>
<td>1 - M 100, P 100</td>
<td>Stuck-at 1 1 - M 100, P 100</td>
</tr>
<tr>
<td>7</td>
<td>247</td>
<td>1 - M 98, P 10</td>
<td>Open 1 - M 98, P 70</td>
</tr>
</tbody>
</table>

Table 4.6: Real defect diagnosis results

The second type of experiment performed was analysis of samples with real defects. The failure information from seven samples that failed the scan test and 8 samples
4.6. EXPERIMENTAL RESULTS

that failed the delay test was available. The results obtained after diagnosing the samples with scan test failures are presented in Table 4.6. These samples are from different products, manufactured with different technologies: samples 1, 3, 5 and 6 are manufactured with a 0.5\(\mu\)m process, sample 2 with a 0.35\(\mu\)m process and samples 4 and 7 with 0.18\(\mu\)m process.

The first sample was failing for only one scan vector and the number of suspect faulty nets with matching = 100 was 18. After running the second step, mapping with a fault model, the best fault candidate indicated was a short between two vias (M 100, P 50). The cross section of the ”victim” via revealed a pushup defect (Figure 4.18). The behavior observed was similar to that of a dominant bridge, it matched only 50% from the bridge signature (P 50). The signal net with the bad via was ”dominated” by the neighboring net via a crosstalk effect.

![SEM image of the via defect in sample 1](image)

Figure 4.18: SEM image of the via defect in sample 1

The second sample failed 97 scan vectors. Only one faulty net could explain all the failing vectors. After running the mapping with a fault type procedure, the best candidate indicated (M 100, P 100) was a bridge in metal layer 3. One of the shorted nets was the faulty net identified as the best candidate in the first step. A cross-section has been made at the position indicated and extra material between the suspected
shorted nets was identified (Figure 4.19).

![SEM image of the defect from sample 2](image)

Figure 4.19: SEM image of the defect from sample 2

Both defects on sample 1 and 2 could be modeled using the line-dominant bridging fault model. If no bridging fault model was applied the number of bridging candidates with the highest rank was 82 for the first sample and 8 for the second sample. When only one sample is available for failure analysis, this is mainly the case when customer returns are analyzed, it is important that the number of suspect locations is not high and that the correct position is at the top of the list. The etching procedures can affect the circuit where the next suspects are located. In that case the defect identification will not be possible.

After completing the faulty net identification step for the third sample, 8 nets were identified as having the highest rank. No bridging fault gave better ranking counts than the ones obtained in the first step, but when the open fault model was applied, for one of the suspect nets only 4 from the 12 branches were failing. The values computed for matching and prediction after mapping with the open fault model indicated it as the best match. The physical analysis confirmed the existence of an open in the metal line which was common to all the failing branches (Figure 4.20).

The diagnosis results for the sample 4 indicated that there is only one defect present (the value for matching for the best candidate is 100) and the fault which best explains the behavior observed on the tester is a bridging fault, one of the shorted net is also the faulty net indicated as the best candidate. The physical analysis method used to further locate the actual defect was OBITCH. A light spot appeared on the
Figure 4.20: SEM image of the open defect in sample 3

Figure 4.21: SEM image of the open via in sample 4
image created at the location indicated by the diagnosis algorithm. After deprocessing the defect found was an open via as shown in Figure 4.21. A similar failing mechanisms as the one encountered in sample 1 repeated here, the open via "weakened" the signal net and its logic value is determined by the stronger net that is in the neighborhood.

As can be observed from the results shown in the table VI, the best candidate for the behavior observed on sample 5 is a faulty net. None of the fault models used for mapping gave a better rank that the one obtained for the faulty net with which they were associated. That means then the fault is not a bridge with an adjacent net and because it affects all the fanout branches (the mapping with the open model procedure did not indicate that only some of the fanout branches are affected) it is assumed to be located on the stem or inside the driver. The physical analysis process began first by checking the integrity of the signal net and no defect was observed. After deprocessing and removing all the metal layers a bird’s beak recess was observed at one of the transistors inside the driver cell (Figure 4.22). This defect created a short between the poly and the active region from the substrate and it was affecting the behavior of the signal net which is the output of this cell.

![Figure 4.22: The image of the defect in sample 5](image)

The results obtained after analyzing the failing vectors of sample 6 indicated as
the best candidate two stuck-at faults. One is located on the input and the other one on the output of an inverter. These two faults are equivalent and the test set can not distinguish between them, nor can the diagnosis algorithm. The physical analysis started with the net that was the output of the inverter and an open on metal two was detected when a signal was transmitted with an E-beam equipment from the driver (Figure 4.23). The defect mechanism is similar with the one detected in sample 3. During the CMP polishing step scratches were created. Etching substances used in the following steps accumulated in these scratches and were not removed during the cleaning procedures. In time, these substances etched the oxide and the metal line under the scratches creating an open fault.

![SEM image of the open defect in sample 6](image)

Figure 4.23: SEM image of the open defect in sample 6

The top candidate indicated by the diagnosis algorithm after analyzing the failing data from sample 7 is an open. This fault explains 98% from the failing vectors, which means that there is at least one more defect present that affects the functionality of another net. The physical analysis process began by looking at the top candidate. The sample was deprocessed until metal 4 because this was the highest metal layer in which this net was routed. With the help of an E-beam the location where the signal stopped on the net was found. Optical inspection followed and Ti-residues close to the location of the open were observed. Two cross-sections were made: one through the part where the open is supposed to be and one through the region where the
78CHAPTER 4. FAULT DIAGNOSIS BASED ON THE STUCK-AT FAULT MODEL

Figure 4.24: Cross-section at the open defect location

Figure 4.25: Cross-section through the Ti-residues
4.6. EXPERIMENTAL RESULTS

residues have been observed. A bad via and Ti-residues have been observed on the net where the fault was pinpointed (Figure 4.21). On the second cross-section (Figure 4.25) cracks, holes, Ti residues, missing and a damaged metal line can be observed. These defects are affecting actually the failing net that explains the remaining 2% of the failing vectors.

The delay fault diagnosis algorithm was applied on the failing data collected after 5 samples were tested. These samples are passing the scan test, hence we assumed that the defects only affect the timing of these devices. All the samples are from one design manufactured in a 0.18μm process with 6 metal layers.

The results obtained after applying the delay diagnosis method are presented in Table VI. In the second column the number of test vectors that failed on the tester is given. The devices were tested at two supply voltages, 1.6V and 1.4V. Samples 1 to 4 failed the same vectors on both Vdd values, sample 5 failed more vectors when Vdd = 1.6V than for the lower value. The failing vectors logged when Vdd = 1.4V are included in the set obtained at the higher supply voltage.

<table>
<thead>
<tr>
<th>Sample No.</th>
<th>No. Failing Vectors</th>
<th>Diagnosis results (Rank-match,Freq)</th>
<th>Type of Delay Fault</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11</td>
<td>1 - M 81, P 83</td>
<td>STF</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>1 - M 57, P 20</td>
<td>STR</td>
</tr>
<tr>
<td>3</td>
<td>105</td>
<td>1 - M 100, P 100</td>
<td>STF</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>1 - M 100, P 100</td>
<td>STR+STF</td>
</tr>
<tr>
<td>5</td>
<td>14 at Vdd=1.4V</td>
<td>1 - M 86, P 50</td>
<td>STR+STF</td>
</tr>
<tr>
<td>5</td>
<td>2 at Vdd=1.6V</td>
<td>1 - M 100, P 100</td>
<td>STF</td>
</tr>
</tbody>
</table>

Table 4.7: Transition delay diagnosis results

Column 3 and 4 present the values of the ranking counts and the type of fault suspected to be the best candidate by the diagnosis procedure. The notation from the last column indicates whether there is a slow-to-fall (STF) or a slow-to-rise (STR) transition fault at one of the inputs or at the output of a certain gate. For all the samples the diagnosis method indicated as the best candidate a transition fault at one of the inputs of a gate. The signal nets supposed to have a transition fault have a fanout > 1, thus more branches, but the diagnosis algorithm indicated that only one of them was failing.

For sample number 1, the failing vectors which remained unexplained by the candidate with the highest rank shown in column 3 are from another scan chain than the ones explained. The diagnosis algorithm was applied to these remaining vectors and a slow-to-rise transition fault at an AND gate input was pointed as a suspect with M 100 and P 100.
In the case of sample 2, the two suspects which explain the remaining vectors after eliminating the ones explained by the top candidate are located one gate downstream, on the path to the failing flip-flop.

For samples 3 and 4 there is only one transition fault (a slow-to-fall for sample 3 and slow-to-fall and slow-to-rise for sample 4) which explains all the failures logged on the tester. All the test vectors that should detect the transition faults on these suspects have been observed on the tester.

When analyzing the failing data obtained at $V_{dd} = 1.6\,V$ from sample 5, the vectors which remained unexplained by the top candidate indicated by the diagnosis algorithm are exactly the same as the ones logged when the sample was tested at $V_{dd} = 1.4\,V$. The locations of the two transition faults are at two of the inputs of the same gate and it may indicate that there is a defect located inside this gate. This defect produces a delay when propagating a transition from one of the gate inputs for the higher supply voltage and from two of them at the lower voltage.

For the samples with scan failures, the correctness of the diagnosis results have been verified by doing physical analysis. For the samples with delay faults the analysis is not yet completed hence, the verification is done by using simulation. Delay faults are injected in the netlist at the locations indicated by the diagnosis algorithm and simulation is performed. We modeled the transition delay faults as buffers where the rise and fall delay can be easily modified. These buffers were inserted in the netlist at the inputs suspected to have a delay fault and this modified netlist was simulated with VERILOG. The value of the clock cycle for the two normal operational modes was set equal with the one used by the test program when the sample was tested. Also, the size of the delay faults inserted was equal with the one measured on the tester. Table 4.8 presents the size of the clock cycle between the two normal cycles (column 2) and the value and type of the delay inserted (column 3).

<table>
<thead>
<tr>
<th>Sample No.</th>
<th>Clock Cycle Size(ns)</th>
<th>STR/STF Fault Size(ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>26</td>
<td>0/26</td>
</tr>
<tr>
<td>2</td>
<td>24</td>
<td>24/0</td>
</tr>
<tr>
<td>3</td>
<td>42</td>
<td>0/42</td>
</tr>
<tr>
<td>4</td>
<td>21</td>
<td>20/20</td>
</tr>
<tr>
<td>5</td>
<td>21 at 1.4V</td>
<td>21/21</td>
</tr>
<tr>
<td>5</td>
<td>21 at 1.6V</td>
<td>0/20</td>
</tr>
</tbody>
</table>

Table 4.8: Delay fault simulation parameters

For all the samples, the simulation results matched the behaviour observed on the tester. To verify even more if the location predicted by the diagnosis algorithm are
the correct one, path delay test patterns were created to test the paths which contain the suspected fault location. With the exception of the fault indicated as suspect for sample 5 at 1.4 V and sample 2, the path test generation algorithm could generate test patterns for several paths passing through the suspected fault locations. The samples were again tested with the path delay test patterns and they all fail. These results give more confidence that the fault location pinpointed by the diagnosis algorithm is the correct one.

4.7 Conclusions

In this chapter a new fault diagnosis method for locating faults in full scan circuits has been presented. It uses the stuck-at fault simulation results combined with a mapping and ranking procedure to locate stuck-at, bridging, interconnect opens and delay faults. The input information are the complete datalog from the tester, test patterns, netlist, layout and the output is an ordered list of suspected fault candidates together with their location and type. Even though it is based on the simulation results of a simple fault model (the single stuck-at) the mapping and the ranking procedure added make the proposed diagnosis approach a promising localization technique. The method improves the existing techniques by increasing the precision and accuracy. It has successfully identified the location of single faults, stuck-at, bridging, open and transition faults, and also of multiple faults not only on samples that contained injected faults but also on samples with real defects. In all the cases the correct fault location has been indicated as the best candidate.
Chapter 5

Yield loss: causes and improvement methods

5.1 Introduction

Manufacturing yield in the semiconductor industry is an economic parameter of paramount importance. Due to the more aggressive product windows, the ability to rapidly achieve high yields and go to volume manufacturing decides if the company is successful or risks to go out of business. The total cost of a VLSI product is only partly determined by the silicon manufacturing costs, the level of its profitability is largely determined by the yield achieved during the manufacturing process. Increased yield loss results in fewer functional devices at the same manufacturing costs and this can lead to a loss of several million dollars per year.

![Yield learning phases](image)

Figure 5.1: Yield learning phases
Normally, whenever a new manufacturing process is introduced, the yield is low to start with. A rapid yield ramp up phase then follows which finally stabilizes at a high level in volume production. This trend is illustrated in Figure 5.1.

In all these stages it is very important that the causes of yield loss are quickly and correctly identified so correction actions can be taken to avoid their occurrence in the future. In-line inspection data, special test structures and embedded memories are widely used to detect and analyze yield loss. Inefficient wafer usage, wafer damage or mis-processing may be important contributors to the yield loss. However, the most important contributors to the yield loss are local process-product interactions. The manufacturing process of an IC consists of hundreds of steps and during these steps many factors can result in yield loss. Therefore, this chapter will first briefly describe a CMOS manufacturing process. Then, the following sections will present and classify some of the most important yield loss causes. An overview of the traditional methods for improving yield will be also given.

5.2 The structure of a CMOS manufacturing process

As mentioned in the introduction, modern IC manufacturing processes consist of many steps. The most important are the photolithographic, etching, oxidation, deposition, implantation, diffusion and planarisation steps which are frequently repeated throughout the process [95].

Figure 5.2 presents the flow of a basic CMOS process with an n-well in which PMOS transistors are implemented.

The first step of any process is the oxidation. A 30 to 50 nm thick layer of silicon-dioxide (SiO2) is grown on the entire surface of a monocrystalline p-type silicon wafer by exposing it to pure oxygen (dry oxidation) or water vapor at high temperature (wet oxidation). A layer of nitride (Si3N4) is then deposited over the SiO2 layer. The process used for deposition is called Chemical Vapour Deposition (CVD). Reactants in the vapour-phase are transported and react with the substrate surface creating a film (thin layer) of single crystal material and some by-products. The by-products are removed from the surface. For deposition of nitride a low-pressure CVD process performed in a chamber with medium vacuum (0.25-2.0 torr) and temperatures between 550 and 750°C is used. The chemical reaction performed is:

$$3 \text{SiCl}_2\text{H}_2 + 4\text{NH}_3 = \text{Si}_3\text{N}_4 + 6\text{HCl} + 6\text{H}_2.$$

The next step is a photolithographic process. A layer of photoresist is deposited over the nitride and it is exposed selectively to light. The exposure is done through a mask that defines where the active areas will be. The photoresist is then developed and the exposed areas of the photoresist are removed. The resulting pattern determines which silicon nitride will remain during the succeeding etching process.
There are several etching techniques grouped in two main categories: wet etching and dry etching. By wet etching the wafer is immersed in a chemical etching liquid and the etching rate is the same in all directions. In the dry etching methods which use plasma, the wafer are immersed in a plasma containing chlorine or fluorine ions that etch on the direction the ions are "sent" to the wafer surface. Another dry-
etching technique called sputter etching, uses gas ions as argon (Ar+) which physically dislodge the atoms on the wafer surface.

After completing the etching step, the photoresist is completely removed and an image of the mask pattern remains on the nitride layer (Figure 5.2.a). The next step is oxidation. A LOCOS (Local Oxidation of Silicon) oxide is grown only in the regions not covered by the nitride layer by using thermal oxidation. The wafer is exposed to oxygen at a high temperature (900-1200°C) and the following chemical reaction takes place:

\[
\text{Si (solid)} + \text{O}_2 \text{ (vapour)} = \text{SiO}_2 \text{ (solid)}.
\]

This LOCOS oxide separates the active areas (Figure 5.2.b). In the new deep-submicron processes Shallow-Trench Isolation (STI) is implemented to prevent any warpage and stress problems associated with the high-temperature needed when LOCOS is grown. This process grows in the beginning a thinner layer of oxide (10-14nm) followed by deposition of a 100-160nm thick nitride layer. After the photolithographic process, the active areas are etched to create trenches that are then rounded by a thermally grown oxide layer. After removing the resist, a thick (typically 700nm to 1000nm) oxide High-Density Plasma (HDP) is deposited.

After the completion of the oxidation process another photolithographic step is applied. The photoresist will now "open" the areas where the p-type transistors will be placed. The n-well areas are created by implantation followed by diffusion. The implantation process takes place in a vacuum chamber where ions are accelerated to the silicon under the influence of electric and magnetic fields. Phosphorus ions are commonly used for creating a n-well. The ions are initially collected at the silicon surface and they diffuse more deeply due to subsequent high temperature steps. For the creation of source and drain of the transistors a sequence of photolithographic and implantation steps are used. A separate mask is used for each type of the transistors in order to "open" the regions where the implantation will be made. The results obtained are shown in Figure 5.2.d. The drain and the source regions of the transistors are now created. The next step is the creation of the transistor gates and short interconnects between the transistors. The material used for this purpose is polysilicon. For some processes n-type polysilicon is used for both NMOS and PMOS transistor gates, for other n+/p+ dual polysilicon, n-type for NMOS transistors and p-type for PMOS transistors. The polysilicon layer is deposited on the wafer and photolithographic and etching steps follow for defining the desired polysilicon pattern.

After the polysilicon is deposited the transistors are now completely created. The next steps are creating the remaining interconnections between the transistors. First an oxide layer is deposed on the wafer. A layer of photore sist follows and after it is developed the contact holes are etched in the oxide layer. These holes allow the connection to the gates, drains and sources of the transistors. A metal layer is then
5.2. THE STRUCTURE OF A CMOS MANUFACTURING PROCESS

deposited and etched for creating the final interconnect pattern after the photolithographic step is applied. The device with the first layer of interconnection obtained after the completion of the etching step is shown in Figure 5.2.e.

![Diagram](image)

Figure 5.3: The conventional (a) and damascene (b) methods for creating the metal layers [95]

The increase in number of processing steps which include deposition and decreasing feature sizes of today's submicron and deep-submicron processes results in an increasingly uneven surface. The regions where more metal layers are deposited one on top of each other become much higher than the regions with less metal layers. If this difference in topology is above an certain limit (which is mainly a problem for processes that contains five or more metal layers), an out-of-focus situation occurs during the exposure of the resist. The vias that are located in the "lower" regions are not well defined and will cause the creation of failing contacts after dry etch (the vias are to short and no contact is made to the metal layer below it). Therefore, several planarisation steps for flattening the surface are necessary before the next deposition step is performed. The planarisation method used in conventional CMOS processes is based on the Spin-On-Glass(SOG) formation. The surface of the wafer is coated with a liquid at room temperature and rotated such as the liquid flows all over the wafer to equalize the surface. Next, the wafer is cured to form a hard silicate or siloxane film, which included also phosphorus to prevent cracking. With this technique the small gaps are easy to fill and the surface is locally well planarised. Another planarisation technique used is CMP (Chemical Mechanical Polishing). CMP is based on the com-
bination of mechanical action from a rotating polishing table and the simultaneous use on a chemical liquid (slurry) which contains polishing particles (e.g. silica or alumina) and an etching substance (e.g. ammonia). This method is well suited for the planarisation of rough areas, i.e. during the metallisation steps.

With the increasing transistor density in today's products, there are more metal layers necessary to create all the interconnections needed for constructing the desired functionality. The conventional method for creating an interconnect metal layer includes the following steps: oxidation, metal deposition, photolithography, dry etching of metal and again oxide deposition, etc. (Figure 5.3 a). The new processes use another technique called damascene metal patterning (Figure 5.3 b). The metal patterns are created by etching a layer of dielectric, overfilling them with metal and then polishing the overfill by using CMP (Chemical Mechanical Polishing) until the pad touches the dielectric. A new layer of oxide is deposited and the steps for creating the next layer of interconnect can be repeated.

![Diagram of a CMOS process](image)

**Figure 5.4: An advanced deep-submicron CMOS process [95]**

An advanced deep-submicron CMOS (Figure 5.4) process incorporates several different processing steps as the basic process described previously. To isolate the active
areas STI is implemented and a retrograde-well process is used to create the n-well and p-well for both transistors. To reduce the short-channel effect on the threshold voltage (so called threshold voltage roll-off) drain/source extension implants are created. These implants are less deep than the actual source/drain junction which allows a better control of the channel length. To reduce the RC delays and consequently improve the circuit performance, low-ohmic silicide layers are formed by silicidation (the reaction between Ti or other metals with Si) in polysilicon and source/drain regions. Titanium (Ti) is used in the contact holes to remove oxide and to create a better contact with the underlying silicide. Also, in the contacts, TiN (titanium nitride) is used due to its good adhesive properties. The contact holes are mainly filled with tungsten and also a TiN film is often deposited on the top of the metal layers and serves as a Anti-Reflecting Coating (ARC), it absorbs most of the radiation that penetrates the resist.

The processes presented here are only giving a general idea of the steps involved in the manufacturing of an integrated circuit. Different types of circuits have different processing requirements, for example RAMs require a technology that allows very high bit densities. For the new generation of ICs composed from billions of transistors working at 100 Ghz, there is a need for a sustained effort in developing new processing techniques.

5.3 Yield loss causes

Given the complexity of the manufacturing process there are many causes that may result in defects that produce yield loss. A possible grouping of the yield loss causes is as follows:

- human errors;
- equipment failures;
- instability in process conditions;
- material instabilities and inhomogeneities.

Humans control all the process steps mentioned in the previous section and all the equipment involved in a manufacturing process. There is always a chance that the parameters are not correctly established or a wrong recipe is applied due to misjudgment. Therefore, humans remain an (uncontrolled) cause for yield loss. Random fluctuations in the environment which surrounds the chip, such as variations in the temperature, instability in the process conditions as for example turbulent flow of gases in oxidation and diffusion are difficult to control and avoid. As a consequence, they are an important cause for yield loss.
The materials used in the manufacturing process have to be extremely pure to guarantee the high grade of reproducibility and reliability required for ICs. Any fluctuation in the purity and the physical characteristics of the chemical compounds can deteriorate the functionality of the device. Deposited material such as aluminum and resist can contain particles that cause defects. For example, a contamination in the interface between Si and SiO2 can lead to a defect known as "pin hole" (see Figure 5.5). A current flows through the oxide increasing the value of the transistor’s leakage current. In time this hole becomes bigger affecting the functionality of the device.

![Image of a gate oxide pinhole defect](image)

Figure 5.5: A gate oxide pinhole defect

It is very difficult to mention all the causes that produce defects during the manufacturing process and their effect. Some general cases were mentioned above. More examples of what can go wrong during some of the most important steps of the manufacturing process and the type of defects resulted are given next.

During a photolithographic step all the causes mentioned before can determine the apparition of defects. Equipment vibration can lead to inaccurate pattern images on the wafer and results in open or short circuits. Variation in temperature can cause the projected mask to exceed the required tolerance. High humidity results in a poor bond between the photoresist and the wafer and it will be etched during the next processing step. Dust particles on the mask can lead also to the apparition of spots on the wafer causing opens or short circuits.
5.3. YIELD LOSS CAUSES

One of the problems that can occur during the etching step due to difference in the topography of neighboring regions is when the contact holes are created. Not all the dielectric material is completely removed, a very thin layer of oxide will remain at the bottom of the contact creating an open (Figure 5.6). Another problem which can occur when wet-etching is used is "under-etching" (Figure 5.7) when the width of the layer is closer in value to its thickness. The width of the etched layer becomes smaller, in some extreme cases the layer can be completely etched away in some regions.

![Figure 5.6: The SEM picture of an open contact](image)

![Figure 5.7: The under-etch problem](image)

During the plasma etching or deposition step, the ions and electrons in the plasma can create significant electrical fields across the thin gate oxide, the so-called antenna effect (Figure 5.8). The charges that are build up on the antenna structures can stress the gate oxide affecting the performance of the device by producing voltage threshold shifts or gate leakage.
The most common defects associated with the CMP process are the scratches induced by the polishing. Etching substances remain in the large and deep scratches after the cleaning step and in time they will affect the metal layer situated under it by creating opens (Figure 5.9).

The gate oxide layer must be of high quality and very reliable. Any fluctuations in the oxidation process parameters as for example humidity temperature can cause the reduction of the specified thickness that leads to the deterioration of the transistors properties. The amplification factor of the transistor will be high and the transistor will produce more current that will make it faster. When signals have to arrive in a certain sequence, this speed-up can create malfunctions.
5.4 YIELD LOSS CLASSIFICATIONS

5.4 Yield loss classifications

The examples of yield loss causes described above are typical cases of what can go wrong during a manufacturing process. As mentioned before, these are only a small fraction of the entire spectrum of defects that can occur. Some of these defects are creating faults that are detected during testing and/or modify some of the parameters of the device. Only these defects are contributing to the yield loss and it is imperative to identify and characterize them so corrective actions can be started.

There are three main criteria used for classifying the yield loss [96]: manifestation, affected area and pattern. When applying the first criteria, manifestation, yield loss can be decomposed in functional yield loss and parametric yield loss. Functional yield loss is composed from dies which do not function, thus were failing one of more of the tests applied. Some of the causes of the functional yield loss are dust particles, scratches, contamination, severe processing variations that cause the device to be manufactured incorrectly and fail. Parametric yield loss consists of dies that do function but their performance is not between the specified limits. They are primarily caused by less severe processing variations that cause the device to behave differently form the specification, e.g. lower frequency, incompatible voltage range, etc.

According to the affected area classification criteria, we can distinguish two categories of yield loss: yield loss due to local defects and yield loss due to global defects. Local defects appear in a small area or points on the wafer. Only one die or a limited small number of dies are affected by these local defects. Typical examples of causes for local yield loss are the spot defects. Global defects affect a large area of the wafer or even the entire wafer. One example can be the use of a wrong recipe that will affect the entire wafer or a variation in the process parameters along the wafer (see Figure 5.10).

The last classification criteria, pattern, divides the yield loss in two categories: random and systematic. The most common causes for random yield loss are contaminants causing shorts or opens, scratches. They are randomly distributed across the die or the wafer and they can affect a die or a small group of dies. The systematic yield loss consists of dies affected by the same type of defect. The affected dies can have the same position on different wafers, can form a specific pattern on the wafer or can be randomly distributed across the wafer. Causes for systematic yield loss include: design errors and inadequate equipment, process recipes or materials, reticle defects, etc. (Figure 5.11).

These three criteria used for classifying the yield loss are not mutual exclusive. For example the systematic yield loss can be local or global, functional or parametric. The goal is to obtain and maintain a high yield (ideally 100%) as quickly as possible. Therefore regardless of the type of yield loss, all the causes have to be correctly identified and characterized in order to avoid them in the future.
5.5 Yield improvement techniques

Three distinctive sets of activities are involved in the process of improving the yield: the detection of yield loss, the identification of the failure source (defect diagnosis) and the corrective actions. In-line inspection, parameter evaluation using special test
structures and memories are widely used in the early and intermediate phase of the yield learning process (See Figure 5.1), as means for detecting and identifying yield loss.

5.5.1 Yield improvement using in-line inspection data

In-line inspections are routinely performed on product wafers at predetermined steps in the process flow. Equipment like KLA/Tencor visually inspect the wafer and the defects found are classified according to pre-established defect categories. These categories are determined based on process parameters, design characteristics like critical area density and known defect mechanisms from previous yield improvement activities. All the collected data is continuously stored in a database. Complex software systems [97] are developed to analyze this information and they are capable to identify the new defects added by each process step, the defects which become permanent and the defects which are removed by subsequent processing. When a significant defect is found, studies are performed to isolate and identify the process step which caused the apparition of this defect. The process of identification and characterization of the defect is quite fast because the database contains the coordinates where the defect is located, the defect size and the processing step performed before the inspection was done. However, not all the defects that are causing a device to fail are detected by the in-line inspection. With the increasing densities and reduced feature sizes the size of the defect that can cause a failure is drastically reduced and these are difficult to distinguish against a background that consists of large metal grains. Some of the defects like underetched contacts, line opens, gate oxide shorts and via opens are not optically detectable. If they cause the device to fail they will be detected at the wafer test and other methods are necessary to identify and characterize them.

5.5.2 Yield improvement using test structures

Test structures are special designs that are manufactured together with the product. The results of the electrical measurements performed on these test structures are used in characterizing and understanding the defects present in the device and in eliminating process-design mismatches. The test structures are actually short-loop monitors specially designed to quickly identify specific process problems. Known failure mechanisms derived from process development and from previous products together with device characteristics are taken in consideration when designing the test structure. Usually a list of the expected failure mechanisms is created at the beginning of the manufacturing of any device [96]. Some examples of the failure mechanisms that can be present on this list are: active to active shorts, poly to active shorts, metal to metal shorts, opens in the active, opens in poly, open contacts, opens in via, transistor
parameter variations, etc. Some of the test structures specially designed to detect and characterize some of these defect types will be presented next.

For the detection of shorts, which are one of the most common defect mechanisms in today’s IC’s, a so-called comb-comb structure is used [98]. This structure contains two conducting structures lying next to each other over a long distance. For the detection of open circuits a long conducting structure, called string is required. Using these test structures separately is not so efficient. Therefore, mainly a combination of them is implemented the so-called comb-string-comb structure [99, 100] (Figure 5.12).

![Comb-string-comb test structure](image)

Figure 5.12: A comb-string-comb test structure

The shorts can be detected by measuring the leakage current between the comb and the string, and the opens by measuring the resistance of the string. This structure gives also the possibility to determine the multiplicity of the defect [101] that is important for calculating the defect density parameter used in yield prediction models. The space between the comb and string and the size of the test structure is chosen according to the critical area of the device and the size of the defects expected. Since smaller defects are more dominant then the larger ones, the test structures for detecting them are smaller than the ones used for larger defects. A more complex test structure for detecting and characterizing the shorts in and between the layers is the harp test structure [102]. It contains vertical and horizontal parallel lines placed inside a given boundary pad frame. The connections between these parallel lines and the pads are allowing the measurement of resistance between any number of adjacent lines in order to detect single or multiple defects. Another test structure for detecting short and opens in the interconnect layers of a product is the NEST structure [103]. This test structure (see Figure 5.13.) consists of parallel lines designed as serpentines to fill the complete test chip area, each of the lines is connected to two pads for performing measurements to detect if the lines are shorted or open. By using this
structure the defect size distribution can be determined by comparing the number of detected defects dependent on the number of involved lines.

\[ \text{Figure 5.13: A NEST test structure} \]

The test structure for monitoring the thickness and the quality of the insulating oxides between different sets of conducting layers consists of several parallel plate capacitors. By measuring these capacitors, defects like pinholes, cleaning related defects or crystal defects in the underlying substrate can be detected.

Regular matrices consisting of identically designed devices are widely used for detecting and characterizing the variation in parameters of the submicron CMOS transistors. These matrices contain thousands of transistors that are organized in blocks that have a different orientation in the layout. The parameters of each transistor can be separately measured to obtain information on the mismatch parameters as a function of device dimension and the position in the die.

The test structures mentioned until now are more focused on detecting the problems related with the process. They usually remain almost the same when moving from one generation of technology to the next. Only small changes resulting from a reduction in the minimum feature size have to be made. The design characteristics of the devices manufactured have to be taken also in consideration when test structures are developed to detect and characterize the causes of yield loss. For example the strings for detecting the open vias can be designed in such a way that the surrounding of each via resembles with the surrounding from the actual device. These test structures are also called MIMIC structures. Figure 5.14 shows an example of a MIMIC via test structure for a SRAM. Another possibility is to build the test structures directly on the top of the device. This method is called by its authors the spidermask approach [104]. A test vehicle consisting of more test structures is build on the top of a product. The structure of this test vehicle is the same as the one of the device until a certain step (for example until metall has to be created). Then the available lines from the
product are contacted to form a comb structure. At the same time, by picking up the right lines from the product structures a string for detecting the open vias can be created. The form of these structures is not always perfect but a good correlation with the actual product characteristics can be made.

![Diagram of a MIMIC test structure for a SRAM](image)

Figure 5.14: A MIMIC test structure for a SRAM

There are several advantages and disadvantages when using test structures for detecting and characterization of yield loss causes. As mentioned before, the test structures are short loop monitors such that if the failure mechanism they target occurs it can be quickly detected and characterized. This is a major advantage especially because now time-to-market puts an increasing pressure on rapid yield ramp up. A disadvantage is that not all the possible failing mechanisms can be predicted up front. Therefore, these new failing mechanisms can not be taken in consideration when the test structures are designed. Also, to be able to accurately reflect the problems that may occur in the manufacturing process of a certain device, the test structures have to be on the same wafer with the product. However, since the available space is limited, not all the available test structures can be included. The consequence is that some failing mechanisms cannot be captured. One solution is to use a part of the actual IC as a test monitor for some specific failure mechanisms. For example the metall-vial-metall2 design of an SRAM can be build as a via-string test structure. Nevertheless, using this approach only a very limited number of test structures can be replaced.
5.5. YIELD IMPROVEMENT TECHNIQUES

5.5.3 Yield improvement using embedded memories

The usage of memories as yield improvement vehicles has many advantages. They undergo the same process flow as that for products, not only several steps as the test structures, making them a full flow process monitors. Due to their regular structure the identification of the physical location of the failure is very easy. Bitmaps generated from the test results of the memory array provide a physical map to the failing location. For these reasons many manufacturing facilities are running a memory device (usually an SRAM) as a yield improvement vehicle. Even more, new complex System-on-Chip designs, which are a growing market, contain always an embedded memory. No extra test vehicle has to be manufactured for yield improvement purposes, the test results from the embedded memories can be used for detection and characterization of the yield loss causes encountered.

As mentioned earlier, the bitmaps generated from the test results of the memory array are a powerful tool for identification of the failure location. The bitmaps indicate the failing bits as detected by the test procedure and the operation they failed (read 0, read 1 or both). The translation to physical coordinates is relatively easy due to the regular structure of the memory array. Once the physical location is known, the process of identification and characterization of the defect that cause the failure is started. This process is time consuming and can not be preformed in high volume. Therefore alternatives are necessary. Some well known and often encountered bitmap patterns can be easily "read" and understood by an experienced product engineer. The main problem is that for the complex manufacturing processes, the number of easily decipherable bitmaps is a small fraction of the variety that occurs. One solution proposed in [105] is to generate a defect to bitmap signature dictionary. Defects like disks of extra or missing material are inserted in a random manner in the layout. For each defect, the resulting deformed circuit (if any) is extracted and a list of all possible fault types is obtained. These faults are then simulated and their corresponding fail bitmaps are generated. These bitmaps are then collated into unique bitmap patterns, called bitmap signatures. The bitmaps generated from the tester results are compared with these bitmaps signatures to identify the defects that cause the failures. It is very difficult to predict all the defect types that can occur during the manufacturing process. Therefore not all the defects have their bitmap signature in the dictionary so they can not be identified by the method described.

Another approach [106] is using the in-line inspection results correlated with the bitmaps to identify the defects that are causing the yield loss. The bitmaps generated from the test results of the memory are translated into physical x-y coordinates within the wafer. The in-line inspection data already contains the spatial distribution of the defects detected in the wafer. These two information are then overlapped to determine whether for some bits the defects recorded by the in-line inspection are located near
enough to cause a fall on that bit (Figure 5.15.). Pareto maps can be built at wafer or lot level to obtain an overall view of the main problems that are affecting the yield. One limitation when applying this method is that not all the defects can be detected by in-line inspection. To overcome this problem, the approach described in [107] uses falling signatures predicted by realistic fault extraction when the correlation with in-line data is not successful. The fault extraction procedure examines the circuit layout to determine which short and opens are possible for a given defect size. These faults are then simulated to obtain the fault signature that will be then compared with the bitmap from the tester for identifying the possible cause of the failure.

![Figure 5.15: An example of the overlapping procedure. The gray shapes are representing the falling bits from the bitmap and the black shapes are the defects identified by the in-line inspection equipment.](image)

### 5.6 Conclusions and discussion

Yield loss represents a major concern for every semiconductor company. The causes have to be carefully analyzed and improvement actions taken to be able to achieve and maintain high yield levels. With each new generation of ICs, the manufacturing process becomes more complex to meet the requirements. In this chapter, a brief description of a basic and a deep-submicron CMOS process is made. The reason for it is to give an idea of the multiplicity of causes that can lead to defects and of course, to yield loss. The examples presented in this chapter are only a small part of the defects that can occur during a complex manufacturing process.

Three traditional methods for detecting and identification of yield loss problems
have been presented together with their advantages and disadvantages. New processing steps and the use of new materials and substances demanded by the higher integration densities requirements from the market are introducing new defect mechanisms. Even more, the products are heading towards Systems-on-Chip designs and they also will introduce defect mechanisms specific to them. Therefore, there is a need for a sustained effort in developing reliable methods for detection, identification and characterization of all the possible defects.
Chapter 6

Statistical diagnosis for yield improvement

6.1 Introduction

The complexity and level of integration demands in new designs continue to increase. More functions are being integrated on the same substrate as the industry moves toward system-on-chip technology. As a consequence, the size of logic blocks on a chip has increased tremendously.

In the previous chapter the traditional yield improvement techniques based on in-line inspection, test structures and memories were described. These techniques are only to a limited extend able to bring the yield at the desired high levels. The high integration density limits the sensitivity of the equipment used to perform the in-line inspection to small defects. Not all the failure mechanisms can be predicted upfront so that special test structures can be designed to identify and detect them. There are differences in the observed type of defects between logic and memories due to different transistor count, number of metal layers, topology and other factors. Therefore, some of the important failure mechanisms do not affect memories and hence, can not be detected with memory structures. Hence, the final stages for yield improvement have to be based on defects found in the random logic of an IC. Unique and rare defects are obviously of a lesser interest and all the attention is focused on systematically repeating defect mechanisms.

This chapter describes a new statistical diagnosis method to support fast yield improvement activities. The method identifies repetitive failure mechanisms that appear in the logic part of the design. It uses as input data the final wafer scan-test results, after the die has passed the scan continuity test. The complete toolflow identifies first
the suspects’ location in the form of net names by using the diagnosis algorithm. Then it makes the translation into layout co-ordinates to be viewed on the screen as a fail histogram. This is illustrated by the example shown in Figure 6.1. Further, complex analysis procedures are applied in order to pinpoint directly to the yield loss cause, either in the design (for example layout structures difficult to process, high density areas) or in the process (for example a via processing problem, insufficient cleaning after etching step).

Section 6.2 presents an overview of the previous research on yield improvement methods identifying defects in the logic part of a design. In section 6.3 a description of the new diagnosis algorithm will be given followed by the experimental setup. Section 6.5 presents the data analysis methods and results. In section 6.6 two case study analysis are described and finally, section 6.7 concludes the chapter.

6.2 Previous work

Previous research in yield improvement methods by identifying defects in the logic part of the circuit focused on the characterization of defect types through the creation
6.3 Diagnosis algorithm description

As mentioned earlier, the statistical diagnosis method has to identify repetitive failure mechanisms, and has to be able to function in a production environment. Therefore, several boundary, up front conditions have to be fulfilled:
- Because test time and tester capture memory are an issue, only the first or the first few failing vectors will be datalogged.

- The diagnosis algorithm must be executed fast enough thus maximizing the efficiency of on-line implementations.

- To identify systematically repetitive failure mechanisms, statistical analysis has to be performed on a large amount a data, hence test results for entire lot(s) have to be analyzed.

A dictionary-based approach, cause-effect analysis as defined in chapter 3, has been chosen. The fault dictionary is created off-line using fault simulation, before the actual tests take place.

The fault model used to create the fault dictionary is the simple stuck-at fault model. This is the same model used by the ATPG tool that generated the test patterns. The fault dictionary contains the fault signature of each stuck-at fault and no fault collapsing is used. This fault signature is in fact an ordered list of test vectors (pattern/pin number combination) which should detect the specific stuck-at fault.

The structure of the fault dictionary used is shown in Figure 6.2.

<table>
<thead>
<tr>
<th>Pattern/Pin</th>
<th>Pattern/Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, SA0 on net1</td>
<td>2/2</td>
</tr>
<tr>
<td>2, SA0 on net2</td>
<td>3/12</td>
</tr>
<tr>
<td>3, SA1 on net3</td>
<td>8/10</td>
</tr>
<tr>
<td>4, SA1 on net1</td>
<td>1/5</td>
</tr>
</tbody>
</table>

Increasing pattern number (max. 100 entries)

Figure 6.2: The structure of the fault dictionary

Because we datalog only the first or the first few failing vectors, there is also no need to store the complete fault signatures in the fault dictionary. For our research, the maximum size of the fault signature is set to 100 test vectors for each stuck-at fault.

A positive side effect of this limitation is that the dictionary size is drastically reduced. Hence, the searching and comparing procedures are performed faster. Given the ever-increasing complexity of ICs, this is an important benefit.

The input data to the statistical diagnosis algorithm are the failing vector from the tester and the already mentioned stuck-at fault dictionary. The algorithm searches in the fault dictionary for the stuck-at faults detected by the test vector that failed on the tester. For each failing die therefore, an output list of suspected failing nets is produced. Because we are only able to datalog the first failing vector the output
list can contain a large number of suspects and in most cases only one of these is the correct one.

Two fault diagnosis algorithms especially tailored towards the above mentioned boundary conditions are considered.

For each failing die, the simple algorithm just searches in the fault dictionary for the stuck-at faults detected by the test vector that failed on the tester. No ranking methods are applied, all the stuck-at faults have equal likelihood of appearance. The resulting output list contains the signal nets associated with the selected stuck-at faults. As an example, considering the dictionary from Figure 6.2 and that the failing vector logged on the tester is 4/10, the output list will contain two suspects, net 1 and net 2.

Traditional approaches to order the suspects and to select the best candidate [12, 13] are based on the assumption that the complete fail information is used. To select the best candidate they use the information resulted after intersecting the fault signature with the tester output together with complex matching criteria (a more detailed description of these methods was presented in section 3.3.1). Since in our case this complete fail information is not available, a new criteria to reduce the suspect list is used. The complex algorithm we propose, works as follows: The output list will contain only the nets associated with the stuck-at faults for which the test vector collected from the tester happens to be also the first one that detected that particular stuck-at fault when simulated (the first vector in the fault signature). As an example, looking again at the fault dictionary from Figure 6.2, this would mean that if the failing vector is 8/10 both stuck-at faults no. 1 and 3 contain this vector in their signature but only stuck-at no. 3 will be considered. The output list will contain only net 3.

This method can be expected to fail at times, as the most common defects in CMOS technology are shorts and bad vias, known to behave differently than pure stuck-at faults [44]. In this case, the faults that have the smallest number of vectors before the vector collected from the tester appears in their signature are considered. Once again referring to Figure 6.2: assume the vector from the tester is 4/10, then the only stuck-at fault considered is no. 1 and only net 1 will be in the output list. Even so, the search can sometimes fail because of the limitations of the stuck-at model and because only the first 100 test vectors were stored for each stuck-at fault.

6.4 Experimental setup

The purpose of the statistical diagnosis method is to identify the repetitive failing mechanisms, the so-called "hot-spots" of the design. A "hot-spot" is an area on a die where a certain defect is systematically appearing, due to a specific topology structure
or a higher sensitivity to defects of that region, or a production problem at a specific layer. The output of the diagnosis algorithm described in the previous section is a list of suspect signal nets whose functionality is affected by defects. This list of nets is further processed together with information regarding the routing of the nets, critical area density, via density, in-line inspection data etc. in order to pinpoint to the specific defect type or process problem that may be the main cause of the yield loss. In order to be able to find the cause of the yield loss the diagnosis algorithm has to be effective, the correct location of the defect has to be included in the suspect list and the number of suspects has to be small. A set of experiments has been performed to determine the performance of the diagnosis algorithm. The complete description of the experiments is presented in the next section.

Moreover, two case studies and the results obtained are also described in the subsequent sections.

6.5 Data analysis results

6.5.1 Performance analysis

The goal of the first set of experiments performed is to determine the effectiveness of the statistical diagnosis algorithm. The common practice is to inject defects by FIB (Focused Ion Beam) in good devices. The "treated" devices are then tested and the diagnosis algorithm applied. The algorithm is efficient when the output list is small (ideally 1 suspect) and when the "correct" defect location is included. The diagnosis method described in section 6.3 is developed to analyze failing data from an entire lot, hence such a traditional approach is not feasible. Instead, the actual location of the fault is identified using the method described in chapter 4.

The device chosen for running the experiments is a medium size (75k gates) IC, manufactured in a 0.25μm technology, which was in volume production at Philips Semiconductors. It is a mixed signal design, where the digital part has three cores and for all a full scan test was implemented. Failing data from two lots (25 wafers/lot) was collected. 985 dies were failing the scan test after having passed the scan continuity test. The three cores are tested sequentially and from the data collected it appeared that only two of them were failing. The third core, due its small size, had an insignificant number of fails. The design styles and synthesis constraints used for these cores were different due to their specific functionality requirements. That is the reason that the experimental results are presented separately for each core.

The diagnosis method described in chapter 4 uses as input the complete datalog information. Due to limited tester capability, only a maximum of 1000 failing vectors was collected. Figure 6.3 shows the frequency of the first failing pattern for one of the lots.
6.5. DATA ANALYSIS RESULTS

The test set is compacted by the ATPG tool, and hence the first patterns are detecting a large number of stuck-at faults. The average number of failing vectors collected per die for these two lots is 389. Therefore, it is difficult to distinguish between two suspects when only one failing vector (which mainly is from the first test patterns as can be observed in Figure 6.3) is analyzed. Hence, a more extended analysis has been done considering as input also 4, 8 and 16 failing vectors.

![Frequency of the first failing pattern](image)

Figure 6.3: Frequency of the first failing pattern

For the entire set of failing dies of the two lots from the first set of experiments, first the diagnosis algorithm described in chapter 4 (also called full diagnosis) was applied in order to obtain the location of the faults. In case the full diagnosis indicated that more defects were present in a die, only a maximum of 3 faults (namely those which were failing the majority of test vectors) were considered. Then the statistical diagnosis algorithm was applied considering 1, 4, 8 and 16 failing vectors as input. The results from the full diagnosis method were used to determine whether the suspect net from the output list of the statistical diagnosis algorithm is the correct failing net or not. If for a specific die the net pinpointed as suspect in the full diagnosis output is also in the output list of the statistical diagnosis, the net is considered a correct failing net.

Two performance parameters were computed:

- **Hit rate**: the number of dies which have found the correct failing net divided by the number of dies analyzed

- **Signal/Noise (S/N) ratio**: the number of correctly reported failing nets (sig-
(signal) divided by the number of incorrectly reported ones (noise).

In comparison to the diagnosis algorithm described in chapter 4, which considers as input the complete fail vector information from the tester, for both algorithms described earlier (simple and complex), we can expect:

- A reduced Hit rate of finding the correct failing net (as determined by the full diagnosis) because only few failing vectors are analyzed (the full diagnosis algorithm analyzes the complete datalog from the tester).

- The algorithm in general will find more suspect nets than a perfected full diagnosis algorithm. Thus, they add noise (incorrectly reported failing nets) to the correctly reported ones (signal).

Of course, we would like to have the highest possible Hit rate whilst at the same time maintaining a high S/N ratio. It can be expected that both algorithms have their distinct advantages and disadvantages. The simple algorithm produces an output list with a large average number of suspected nets, which however almost always contains the correct failing net. The complex algorithm on the other hand reduces the Hit rate but also reduces the suspect list size and thus increases the S/N ratio. In the following section we will give an extended analysis of the S/N ratio and Hit rate as a function of two parameters:

- Simple vs. complex algorithm.

- Number of failing vectors recorded on the tester: 1, 4, 8 or 16.

- Design characteristics.

Table 6.1 presents the average size of the on-line diagnosis suspect nets list.

<table>
<thead>
<tr>
<th>No. Failing</th>
<th>Simple algorithm</th>
<th>Complex algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vectors</td>
<td>1 4 8 16</td>
<td>1 4 8 16</td>
</tr>
<tr>
<td>DSP</td>
<td>78.5 20.4 13.2 10.3</td>
<td>21.5 11.7 7.8 6.7</td>
</tr>
<tr>
<td>SC</td>
<td>53.8 13.3 11.3 10.6</td>
<td>13.5 6.7 5.6 5.1</td>
</tr>
</tbody>
</table>

Table 6.1: The average size of the statistic diagnosis suspect nets list

The results presented here were obtained by considering 1, 4, 8 and 16 failing vectors collected from the tester. In the first 4 (data) columns the statistical diagnosis algorithm was using only the simple algorithm. The last 4 columns present the results when the complex algorithm is used. The figures show that when using the complex algorithm the average suspect list size is less than half as compared to the results obtained by the simple algorithm. Increasing the number of collected failing vectors
6.5. DATA ANALYSIS RESULTS

to 4, 8 and 16 further decreases the average size of the suspect list to figures which are feasible for further statistical and physical analysis (see last column from Table1).

In Figure 6.4, the results of the analysis of the performance of the statistical diagnosis are shown. The analysis is done for the simple algorithm (Fig 6.4. a)) and for the complex algorithm (Figure 6. 4. b)). For both situations the S/N ratio and Hit rate are computed when 1, 4, 8 and 16 failing vectors are processed.

The highest value for the Hit rate is obtained when the simple algorithm is used but then the S/N ratio is very low (Figure 6. 4. a)). The Hit rate decreases when more failing vectors are processed for each die and when the complex algorithm is applied. When a complex algorithm is used the S/N ratio is improving with up to 400% with a loss of maximum 25% for Hit rate. Considering more failing vectors as input data results in a maximum of 800% S/N ratio improvement with a maximum of 20% loss in Hit rate. The reason is that we used a simple fault model (single stuck-at) to generate the test patterns and the fault dictionary. Concerning the minimum value for the Hit rate (0.75), the S/N ratio increases with 1000% (one order of magnitude) in comparison with the situation when the Hit rate has the maximum value (Figure 6. 4).

The increasing of S/N ratio with the increasing of the number of failing vector processed was to be expected. The fact that Hit rate has small variations during the whole interval it is analyzed and moreover, it is situated at relatively high levels, proves the effectiveness of the diagnosis algorithm.

A more detailed analysis of the results obtained is presented in Figure 6.5. We computed separately the S/N ratio and Hit rate for all the dies for which the complex statistical diagnosis output list has 1 suspect, less or equal to 2, 3, 4, etc. This is the unit for the X-axis: 1 represents the dies for which the output list of the statistical diagnosis contained only 1 suspect, < 2 for the dies with less or equal then 2 suspects and so forth. The "no rec" reflects the situation when the failing vectors collected at the tester could not be found in the fault dictionary. The reason again being that the fault dictionary is built by simulating only stuck-at faults and because only the first 100 test vectors were stored for each stuck-at fault. One can easily observe that the lowest values for both parameters (S/N ratio and Hit rate) are obtained when only one failing vector is processed. When more failing vectors are processed, both the S/N ratio and the Hit rate are increasing, where the highest values are obtained for 16 failing vectors.

The reason we performed this detailed analysis is to study how we can further increase the S/N ratio, be it at the cost of the Hit rate. A potential way of doing this is to discard the devices for which the output list, as produced by the statistical diagnosis, has a number of suspect nets above a certain limit n. For both cores, when 16 failing vectors are processed, the Hit rate curve increases linearly until the number
Figure 6.4: S/N and Hit rate analysis results for: a) simple search method and b) complex search method
Figure 6.5: Detailed analysis of the S/N ratio and the Hit rate for the DSP core (a) and SC core (b)
of suspects per die is about 6. Then it tends to stabilize and for the categories number of suspect nets per die > 20 increases again till it reaches its maximum value. If the limit n is set to 6, which means that only the results which contain 6 or less suspect nets are further used to create the fail histogram, the S/N ratio increases from 0.18 to 0.38 (more than 200%) for the DSP core. This goes with a loss in Hit rate of (only) 15%. When only 1 failing vector is processed for the DSP core this limit has to be set to 10 to obtain the same loss for the Hit rate with a gain of only 30% for the S/N ratio. A similar situation can be observed for the SC core. In the case where 16 failing vectors are processed the limit has to be set at 6, and to 10 when only one failing vector is processed.

We can define the Performance of the analysis method as the product of Hit rate and S/N ratio:

\[ \text{Performance} = \text{Hit Rate} \times \frac{\text{S/N Ratio}}{} \]

If we plot the performance as a function of the suspect list size (for 16 logged failing vectors only), we arrive at a plot shown in Figure 6.6.

![Figure 6.6: Performance analysis results](image)

One can easily observe that the performance curves of these two cores are different. The performance of the DSP is more robust while SC shows a strong dependency against the size of the suspect output list.

As mentioned earlier, the design styles and synthesis constraints used for these cores were different due to their specific functionality requirements and this is the
6.5. DATA ANALYSIS RESULTS

reason for the differences noticed. The blocks have comparable size and number of
scan flip-flops, 2463 for DSP and 2282 for SC. However, the structure and size of the
combinatorial logic is different. This can be observed by looking at the number of
the stuck-at fault reported by the cores, 271k for DSP and 137k for SC. The DSP core
has the average depth of the combinatorial logic in a logic cone of a flip-flop 4 times
bigger than for the SC core, and contains in average approx. 27 times more faults
than a logic cone in SC. This means that for the DSP core, far more faults propagate
into few failing flip-flops resulting in a low S/N ratio.

If we assume that the two blocks span a wide area of design styles, and using the
analysis on the plots from Figure 6.6, it seems reasonable to cut at a suspect list size
of approx. 6 to 10 nets and hence not consider devices with larger lists for further
analysis.

Figure 6.7 shows the influence of the number of failing vectors considered on the
performance of the diagnosis algorithm. The line corresponds to a cut at 8 nets,
whereas the error bars correspond to cuts at 6 and 10 nets respectively. One can
easily observe that the performance is a logarithmic function of the number of failing
vectors. Since on most production testers test time increases proportionally (or even
over-proportionally) with the number of logged vectors, cost figures for the analysis
grow exponentially with the desired performance of the analysis method.

![Performance as a function of logged falling vectors](image)

Figure 6.7: Performance analysis results

After analyzing these plots, the best balance between the S/N ratio and the Hit
rate is obtained when the complex algorithm is applied and more failing vectors (16 in our experiments) are processed. The average size of the final suspect list is small (see Table 6.1) and physical analysis on all the suspect locations becomes feasible. The drawback is that the test time is increasing as well as the memory requirements for the tester. When applying this method in a production environment a trade-off between test time, $S/N$ ratio and Hit rate has to be made.

6.5.2 Fail histogram

![Fail Histograms for the DSP core](image)

Figure 6.8: Fail histograms for the DSP core

The purpose of the statistical diagnosis algorithm is to identify repetitive failing mechanisms, the so-called "hot-spots". After the fail information of one lot had been processed, the suspect nets obtained were visualized by creating a fail histogram. A
3D version of a fail histogram is presented in Figure 6.1. The fail histogram helps to highlight systematically (the high peaks from Figure 6.1) or randomly distributed (the flat regions) failing mechanisms.

Another type of representation of a fail histogram is shown in Figure 6.8. It contains only the results from the DSP core. The histogram shown in Figure 6.8.a is created using the full diagnosis results while the one from Figure 6.8.b contains the statistical diagnosis results. In the statistical diagnosis histogram, there are more peaks than in the full diagnosis histogram (for example the regions marked on Figure 6.8.b). These are the false peaks, which pop up due to the noise signals in the results. When we set the limit of the suspect list size to 10 (Figure 6.8.c), respectively to 5 (Figure 6.8.d), these false peaks are disappearing because the results which are inducing the highest noise level are not considered anymore. This is important as it is widely accepted that it is preferable to avoid misleading results when performing fault diagnosis [111, 112].

The next step after the identification of the existence of a repetitive failing mechanisms is to identify and characterize it’s root cause. This is important because corrective actions to avoid the occurrence of these failures have to be established. A link to a specific design and/or process problem can be created starting from the output of the statistical diagnosis algorithm. For example, a comparison with the critical area density of the design can indicate if the peaks from the fail histogram correspond to regions with high density routing. A more complex analysis of the layout can indicate if in those regions a special structure which is difficult to process correctly is located (for example a via next to two wide metal lines). In this case the correction actions consist in modifying the routing parameters and style of the design.

The complete identification and characterization of the failure cause can be done only by physical analysis. As described in chapter 2, this procedure is very complex and time consuming. The output list of the statistical diagnosis algorithm contains often more than one suspect (as was shown in Table 6.1). To distinguish between them, in-line inspection data can be used if it is available. The locations of the suspect nets pinpointed by the statistical diagnosis algorithm are compared with the locations of the defects detected by KLA/Tencor equipment. The defect close to one of the suspects is most probably to be the cause of the failure. The physical analysis can now be directed towards analyzing only one defect location.

An important application of the statistical diagnosis output is to more accurately determine the kill ratio (given by the in-line inspection equipment) for various inspection steps. The kill ratio of an inspection step is calculated as being the number of dies with defects detected by the inspection equipment which are failing on the tester divided by the total number of dies on which defects are optically detected. A more accurate determination of the kill ratio is done by correlating the coordinates of the defects observed at the in-line inspection with the locations of the suspect
nets pinpointed by the statistical diagnosis algorithm. The adjusted figures for kill ratio can range, for example from the initial 43% or 23% to 100%, respectively to 0% (more details on how the kill ratio is calculated and adjusted are given in section 6.6.2). The new adjusted figures for the kill ratio can pinpoint to steps on which yield improvement activities have to focus (steps with a kill ratio of 100%), or to steps for which the in-line inspection equipment parameters have to be modified so it would not detect nuisance defects (steps with kill ratio 0%).

### 6.6 Case study results

This section presents the results obtained after applying the statistical diagnosis method to two different designs, manufactured with different technologies and situated in a mature phase of their manufacturing process.

#### 6.6.1 Case study 1

![Figure 6.9: Fail histogram](image)

Figure 6.9 presents the fail histogram created with the results obtained after ap-
plexing the statistical diagnosis method to one of the lots mentioned in section 6.5.1. The devices which had in their output list a number of suspects larger than 5 were discarded. In the left top corner of the diagram a high peak can be observed. This peak contains the statistical diagnosis results of 42 failing dies. These dies are originating from 17 wafers (one lot consists of 25) and their location is randomly distributed over the surface of the wafer. The high peak from the left top corner of the histogram does not correspond to a high density area, as it also appears in the diagram obtained after normalizing with the critical area. The failure mechanism that is located in that area is a systematical one but affects only this lot. It has not been observed in the results obtained for other lots.

The fault location pinpointed by the statistical diagnosis algorithm matched exactly with the full diagnosis results; it is a stuck-at 1 fault at the output of a NAND gate with 4 inputs. Unfortunately, the devices were no longer available so determining the physical defect which caused this failure mechanism, was not possible.

### 6.6.2 Case study 2

The results presented in this section are from an experiment performed on a mixed-signal design, processed in a BIMOS2000 technology. This process has reached its mature stage but the yield of the digital part of the design was not at the desired level. The measurements performed on the test structures present on the wafer did not reveal anything relevant regarding the cause of this yield loss. Then our analysis was performed on fail data logged during the final wafer test. The wafer final test program stores a maximum of 10 failing vectors, and these vectors are being processed by the diagnosis algorithm. The data collected is from three different lots, from two lots only the fail data from one wafer was available (27 and 33 failing dies), from the third lot the fail data from 10 wafers (168 failing dies) was collected. For one wafer, the test program was modified to collect maximum 200 failing vectors. This fail information was used as input for the full diagnosis algorithm to determine the Hit rate and the S/N ratio. The values obtained are 0.75 for the Hit rate and 0.4 for the S/N ratio when the statistical diagnosis analyses only 10 failing vectors. These values are high enough to give confidence in the statistical diagnosis results.

Figure 6.10 presents the fail histogram of the output list after analysing the fail data from all the dies. In this histogram, only the digital area is represented, the routing of the nets in the analogue part was not available in the format which can be processed by the statistical diagnosis method.

One can easily observe that there are three regions were most of the failing nets are concentrated. When a comparison with the location of the cells in the layout was made (see Figure 6.11) it was observed that these regions corresponded to the instantiations of one type of cell, called \textit{mcfo}. Also from the distribution of the driving
cell of the nets from the output list of the diagnosis algorithm, we found that this cell was falling over-proportionally, compared to the number of instantiations (see Figure 6.11).

This particular cell is instantiated 294 times in the digital part and occupies 18% from the digital surface. There are 36% from the failing dies were the mcfo cell is considered as suspect. This cell is relative big, but has a similar density and characteristics as the other cells which have a lower falling rate.

When closely analysing the layout of the entire chip, one can observe that the mcfo cell drives very long nets outside the digital area into the analog part of the circuit. Figure 6.13 presents the layout view of the failing nets driven by the mcfo cells. Further, these nets are routed very densely. We can thus estimate that the effective critical area contributing to digital fails is increased by 10 - 15% by these nets alone. Thus, the geographical distribution of fail locations as well as the driving cell distribution behaves essentially as expected.

In this case, further yield improvement must be governed by understanding random fail mechanisms. This is normally done by interpreting in-line inspection data. For
Figure 6.11: The layout view of the digital part

Figure 6.12: Yield loss v.s driving cell type

each inspection step a parameter called *kill ratio* is calculated as being the number of
dies with defects detected by the inspection equipment which are failing on the tester
divided by the total number of dies on which defects are optically detected. The
process steps which have a high kill ratio contain a high number of defects that may
cause failures and these are first targets for yield improvement activities. However,
an accurate understanding of the kill ratio is often difficult to achieve.

Figure 6.13: Layout view of the failing nets

The kill ratio values obtained are not very accurate. If the inspection at two process steps observe a defect on a particular failing die, it can not be established which defect is the cause of the failure and which may be an artefact. Therefore, this die will be considered when the kill ratio of both steps is calculated. For an efficient yield improvement action it is imperious to have the kill ratio determined as precisely as possible so the increase in yield (and profits) is maximum.

The kill ratio values can be adjusted by making the correlation between the in-line inspection data and the suspect nets obtained from the statistical diagnosis algorithm. Figure 6.14 presents some examples of the correlation made between the suspect failing nets and in-line data. We consider to have a match if there is a defect detected by the KLA/Tencor equipment in the neighborhood (an area equal with the tolerance of the equipment) of one of the suspect nets as indicated by the diagnosis algorithm. For each step, the kill ratio will be now calculated as being the number of matches for that step divided by the number of dies where a defect was observed at that step.
6.6. CASE STUDY RESULTS

![Figure 6.14: Examples of correlations with in-line inspection data](image)

For our device, in-line inspection data was available for 4 of the wafers analyzed and correlations were made between the failing net locations and the defect locations as detected visually by the KLA/Tencor equipment. These 4 wafers contained 47 failing dies and for 14 of them the KLA did not record any defects, for another 14 dies the defects recorded are not close to the failing area. For 19 dies the defects observed at the on-line inspections are within the resolution of the KLA equipment matching with the suspected nets. Many of the KLA signals on die 41 can be ruled out as the cause of the device failure, whilst on die 10 even a failure outside the digital area can be attributed to cause a scan failure due to the detailed analysis performed. Die 7 shows that even a relatively large S/N ratio does not adversely affect the usefulness of the method.

The in-line inspection is performed at 23 processing steps and for all the observation points a kill ratio of the defects detected is calculated based on the correlation with the diagnosis output. For three steps with large enough statistics to be of significance, the kill ratios obtained are 15%, 21% and 15% respectively. With the more
accurate correlation method presented here, we can adjust these figures to 100%, 50% and 63%, whereas other inspection steps yielded as low as 0%. This allows to accurately target yield improvement work to relevant process steps.

6.7 Conclusions

In this chapter a new diagnosis method was presented. The method supports (low) yield analysis and yield improvement related actions in the mature phase of the yield learning process. It brings statistical analysis capabilities (commonly used for regular structures like memories) now also into the domain of random logic.

As input, the method uses only a limited number of fail vectors from the final wafer test results, and a stuck-at fault based dictionary. The output is a list of suspected nets whose functionality is apparently affected by defects. This output list covers the statistical information for a complete lot, rather than just for a single die. The derived fall histogram (like the one in Figure 6.1), showing the location in the layout of the suspected nets, is a powerful instrument to quickly detect repetitive failure mechanisms. These results can act as input for physical analysis for a detailed identification and characterization of the defects that are causing the failures. A promising alternative is to perform more complex statistical analysis (e.g. a comparison with a specific topology structure, correlation with in-line inspection data) on the diagnosis output which can result in a correlation to a specific defect mechanism or a process step. Whichever approach is chosen, they all will speed up the yield improvement process dramatically.

The experiments described in section 6.5.1 are focused on analyzing the performance of the proposed method by using two parameters: $S/N$ ratio and Hit rate. Possibilities for improvements are also investigated. Two directions for improving the $S/N$ ratio are proposed and the influence on the Hit rate is analyzed. The first direction considers more failing vectors (than the initial only 1) as input data. The second is to use a filter when representing the results. Only the output lists with less than $n$ suspects will then be considered. The experimental results show that the best results are obtained when more failing vectors are processed. The drawback is that the test time is increasing as well as is the capture memory requirements for the tester. Hence, when applying this method in a production environment, a trade-off between test time, $S/N$ ratio and Hit rate has to be made.

In this chapter two case study results are presented. The results obtained show that the method proposed is able to successfully identify repetitive failing mechanism (case study 1) and to pinpoint towards suspect process steps by adjusting their kill ratio (case study 2).
Chapter 7

Summary

Failure analysis and diagnosis are becoming key activities in today's IC manufacturing. First reason of course being that ICs continue to get more complex, more (metal) layers, more transistors on a die combined with the integration of different kind of circuitry (digital, analog, RF, memory, ...). This also complicates the diagnostic approaches, both physical and electrical. Second important reason for the increasing importance of failure analysis is the fact that Time continues to mean Money. Getting to high yields in a short time in manufacturing ICs is of utmost importance. An efficient and effective failure analysis approach in order to understand the (failures in the) manufacturing processes is then key, next to be able to subsequently improve the processes and indeed achieve the high yields quickly. Also the observation and expectation that the physical failure analysis approach will have to rely more and more on an effective electrical fault diagnosis approach, brought us to the decision to put the focus of the research on electrical failure analysis and diagnosis.

There are two main activities in which fault diagnosis plays an important role: failure analysis and low yield analysis. The objective of these two activities is different but the end result is the same, improving the manufacturing process and achieving higher yields.

The objective of failure analysis is to find the defect(s) in a given die, which fail a manufacturing test or a customer application. Fault diagnosis is the first and the most critical step in the failure analysis process. A new (electrical) fault diagnosis method (described in chapter 4) has been developed. The method improves the existing stuck-at based diagnosis infrastructure and is able to locate not only stuck-at faults but also bridging, interconnect opens and transition delay faults. In various practical experiments the method has proven to precisely and correctly pinpoint to the fault location, while also mentioning the most probable root cause(s) of the fault. The results of the experiments show the (good) performance of the method.
The goal of Low Yield Analysis (LYA) is different from that of failure analysis. It is to improve the yield by identifying new, systematically repeating defect mechanisms and to drive corrective actions in the wafer fabs. For this purpose a large number of devices has to be analyzed, preferably on-line, so the results are available as soon as possible for a quick feedback to the manufacturing process. Traditional techniques for LYA are based on using embedded memories and special test structures as yield improving monitors. The complexity and level of integration demanded in the new designs continue to increase as the industry is moving towards System-on-Silicon technology and consequently memories and special test structures are no longer able to bring the yield quickly to the desired high levels. Therefore new (statistical) techniques, based on analyzing the defect mechanisms that occur specifically in the logic part of the design, are needed. A completely new statistical diagnosis method focusing on yield improvement has been developed and implemented. This method (described in Chapter 6) uses only a very limited number of fail vectors from a tester, and so can be used in a production environment without influencing the manufacturing process. As such, the method fills the need for a structured approach to identify systematically repeating failure mechanisms in the logic part of a circuit, which are the main cause for yield loss after the memories and test structures brought the process to an almost mature process.

The chapters 2, 3 and 5 give a short overview of the state of the art in IC technology, indicate issues in failures in IC processing and in more detail discuss the state of the art in failure analysis and diagnosis techniques.

Summary of the achievements

Besides the literature study, as being a more or less standard activity for a PhD study, and in this thesis reflected in chapters 2, 3 and 5, there are key contributions in the development of new methodologies for electrical fault diagnosis, in implementing the proposed methodology in software tools, and in applying the approaches to practical cases. The developed approaches compare to the state of the art situation in the world, and (hence) also to the situation within Philips Semiconductors. The software developments enhanced the Philips proprietary CAT tool suite, in which also a diagnosis tool was available. This is reflected in the following key contributions.

- The flow of the diagnosis method is new. It first identifies the faulty nets and builds a partial dictionary and then does the mapping with the fault model. This has as results that the correct location of the defect can always be identified even when the fault models used for the mapping are not very effective in describing the faulty behavior.

- The bridge localization part is improved, it is more precise than the until now published approaches.
- A program for extracting the possible bridging faults was developed in order to consider only the realistic faults and to be able to precisely pinpoint to the fault location.

- The part localizing opens is simpler than the existing approaches and at least as efficient and effective as those.

- The delay fault diagnosis method was newly developed. The method is using stuck-at fault simulation results and it is able to precisely and correctly locate slow-to-fall and slow-to rise transition faults.

Using the new software, several experiments were executed, all leading to very good results as has been shown in chapter 4.

The statistical diagnosis approach proposed is completely new; At least until now there haven’t been any publications which compare to this proposal. With this new method also some experiments have been done, and they too gave quite promising results (chapter 6).
Bibliography


BIBLIOGRAPHY


BIBLIOGRAPHY


Samenvatting

Fout diagnose wordt steeds belangrijker bij het fabricage proces van geïntegreerde schakelingen (Integrated Circuits, ICs). Ten eerste worden ICs steeds complexer. Zo neemt het aantal metaal lagen toe, komen er meer transistors op een "die", en bovendien worden steeds meer verschillende schakelingen gegraveerd in een product, zoals digitale, analoge, RF en geheugens. Dit alles compliceert natuurlijk ook het diagnose proces, zowel wat betreft het elektrische als wat betreft het fysische aspect. Tweede belangrijke reden voor het steeds belangrijker worden van fout diagnose is dat "tijd is geld" nog steeds opgaat, ook bij het fabriceren van ICs. Het fabriceren van ICs is nooit foutvrij, en het bereiken van een hoge opbrengst in een zo kort mogelijke tijd is dus erg belangrijk. Een efficiënte and effectieve fout diagnose, teneinde de fout oorzaken te begrijpen en zo snel mogelijk te corrigeren, is een absolute voorwaarde om de hoge opbrengsten te kunnen bereiken. Ook de verwachting dat de fysieke fout diagnose meer en meer zal moeten kunnen rekenen op een effectieve elektrische fout lokalisatie, bracht ons ertoe om onze onderzoek te richten op elektrische fout analyse en diagnose.

Er zijn twee hoofd activiteiten waarin elektrische fout diagnose een belangrijke rol speelt: fout analyse en lage opbrengst analyse (Low Yield Analysis, LYA). De oorsprong van de twee genoemde activiteiten is verschillend, maar het einddoel is hetzelfde, namelijk het verbeteren van het productie proces en het verhogen van de opbrengst.

Het doel van fout analyse is het vinden van defecten in een product (een "die" op de "wafer"), welke als fout werd gemerkt in een productie test of in een toepassing. Fout diagnose is de eerste en meest kritische stap in het fout analyse proces. We ontwikkelden een nieuwe elektrische fout diagnose methode (beschreven in hoofdstuk 4). De methode verbetert de bestaande, op het "stuck-at" foutmodel gebaseerde, methoden en is in staat om niet alleen "stuck-at" fouten, maar ook sluitingen, open verbindingen ("opens"), en transitie vertraging fouten te lokaliseren. Onze methode geeft niet alleen aan waar er een defect is, in welke beogde verbinding in de schakeling, maar ook wordt de meest waarschijnlijke oorzaak van het defect vermeld. De resultaten van de gedane experimenten geven een idee van de (goede) kwaliteit van

139
onze methode.

Het doel van de LYA (lage opbrengst analyse) verschilt van die van fout analyse. Bij LYA gaat het met name om het vinden van, eventueel nieuwe, defect mechanismen en het starten en sturen van correctieve acties in het fabricage proces. Om dit te bereiken moet een groot aantal producten, "dies", worden geanalyseerd. Dit moet bij voorkeur tijdens het fabricage proces zelf, om een zo snel mogelijke terugkoppeling naar het fabricage proces mogelijk te maken. Traditionele technieken voor LYA zijn veelal gebaseerd op het gebruik van zogenaamde opbrengst "monitoren" zoals geheugens en speciale test structuren. De complexiteit en het niveau van de integratie zoals gevraagd in nieuwe processen en producten (System-on-Silicon) hebben ertoe geleid dat de genoemde geheugens en test structuren niet meer voldoende zijn om de gewenste hoge opbrengsten te bereiken. Daarom moeten nieuwe technieken worden ontwikkeld gebaseerd op het analyseren van defect mechanismen die zich met name voordoen in het logische deel van de schakeling. Wij ontwikkelden zo'n statistische diagnose methode, direct gericht op het behalen van een hogere opbrengst en het verbeteren van het productie proces. Deze nieuwe methode (beschreven in hoofdstuk 6) gebruikt maar een gering aantal foutvectoren van de tester, en kan dus prima worden gebruikt in een productie omgeving. In die zin vervult deze methode de vraag naar een structurele methode tot het identificeren van systematische en repeterende defect mechanismen in het logische deel van een schakeling, zijnde de voornaamste oorzaak van opbrengst verlies nadat de geheugens en de test structuren hun werk hebben gedaan. De hoofdstukken 2, 3 en 5 geven een kort overzicht van de stand van zaken in de IC technologie, geven aandachtspunten weer aangaande potentiële fouten bij het fabriceren van IC’s, en discussiëren in meer detail de stand van zaken in fout analyse en diagnose technieken.

Samenvatting van de bereikte resultaten

Behalve de literatuur studie, een meer of minder standaard activiteit voor een proefschrift en in dit proefschrift gedaan in de hoofdstukken 2, 3 en 5, zijn er een aantal essentiële bijdragen te vermelden. Als voornaamste behaalde resultaten zijn te noemen de ontwikkeling van nieuwe methodes voor elektrische fout diagnose, het implementeren van de ontwikkelde methodes in software gereedschappen, en het toepassen van de methodes en de gereedschappen in de praktijk. De ontwikkelde methodes en gereedschappen zijn "state of the art" te noemen in de wereld van IC fout analyse en diagnose.

De software ontwikkelingen zijn uitgevoerd op de bestaande Philips CAT tool suite, waarin zich ook fout diagnose gereedschappen bevinden. De bijdragen zijn terug te voeren tot de volgende essentiële verbeteringen:

- De gebruikte gereedschappen volgen nu een andere volgorde. Eerst worden de foutieve verbindingen gevonden en een partiële "dictionnaire" opgebouwd,
waarna geprobeerd wordt een en ander af te beelden op een foutmodel. Dit resulteert in een correcte locatie van het defect, zelfs als de gebruikte fout modellen niet erg effectief zijn in het beschrijven van het foutieve gedrag van het defect.

- Een programma om de mogelijke kortsluitingen te vinden is ontwikkeld. Dit was nodig om alleen de realistische fouten te beschouwen, en zo nauwkeurig mogelijk de precieze locatie aan te kunnen geven.

- Het lokaliseren van kortsluitingen is verbeterd. Het is nu preciezer dan de tot nu toe gepubliceerde methodes.

- Het lokaliseren van "opens" is simpeler dan bestaande methoden, en zeker niet slechter.

- De diagnose methode voor transitie fouten is nieuw ontwikkeld. De methode gebruikt de stuck-at fout simulatie resultaten en is in staat om zeer precies en correct de "slow-to-fall" en "slow-to rise" transitie fouten weer te geven.

Gebruik makend van de software zijn een aantal experimenten uitgevoerd, welke vrijwel allemaal goede tot zeer goede resultaten hebben opgeleverd, zoals beschreven in hoofdstuk 4.

De statistische methode die wij voorstellen is, voorzover ons bekend, volledig nieuw. Ook met deze statistische methode zijn experimenten gedaan, welke zeer veelbelovende resultaten hebben opgeleverd (zie hoofdstuk 6).
Acknowledgment

This thesis describes the results of the research performed for the last four years in the field of fault diagnosis at the Eindhoven University of Technology and at the Philips Research also in Eindhoven. This research was not possible without the help and support of many people, to whom I am very grateful.

First of all I would like to thank my supervisor, Prof. Rene Segers, who gave me the opportunity to perform this work. His continuos support, guidance, enthusiasm, advice and comments helped me to stay on the right direction to complete this research, for which I am largely indebted.

Secondly I would like to thank the members of the so-called diagnosis monitoring group at Philips: Willem Beverloo, Stefan Eichenberger, Guido Grontboud, Friederich Hapke, Maurice Lousberg and Ruediger Solbach, for making time to participate to our bi-annual meetings and for their valuable feedback on the work performed and on this thesis. Also the discussions we had in between these meetings and especially their help in performing experiments and collecting data (without which this work was not complete) have been of great value. Thanks also go to Bert Otterloo and Victor Zieren who were so kind to revise the chapter about failure analysis. Also to the (many) other people I collaborated with at Philips Semiconductors Nijmegen and Hamburg and Philips Research and to many more which are not mentioned explicitly, I am very grateful.

I also would like to thank all members of the promotion commission for taking time to read this thesis and for their valuable comments.

Last but not least I would like to thank my friends and especially my mother who is always close to me, even though there are many km between her and me. To my beloved mother I dedicate this thesis.
About the author

Camelia Hora was born in Oradea, Romania. She graduated from Technical University Timisoara, Romania, Faculty of Automation and Computer Science, in 1992.

After her graduation she worked as a teaching assistant at Oradea University, Romania. From 1993 until 1998 she occupied the position of teaching assistant at Timisoara University of Technology, Romania, Department of Computer Science.

In May 1998 she visited Eindhoven University of Technology during an European Exchange Program. She returned to Eindhoven in September as PhD student in the Department of Information and Computer Science.

At 1 October 2002 she joined the ED&T/Test group at Philips Research Laboratories as a test consultant. Her research interest is in the area of IC-test, specifically in Design for Test, testability and electrical failure analysis and diagnostics.
Stellingen
behorende bij het proefschrift

On Diagnosing Faults in Digital Circuits
van
Camelia Hora
Eindhoven University of Technology
November 2002

1. We are what we think. All that we are arises with our thoughts. With our thoughts we make the world.
   - Buddha

2. Our greatest glory is not in never failing, but in rising every time we fall.
   - Confucius

3. Electrical fault diagnosis is like looking for a needle in the haystack. With the remark that the haystack is getting larger and the needle is getting smaller, following Moore’s law.
   - this thesis

4. There is no better remedy against stress and frustrations than a fruitful shopping round.

5. Even though it is doubted by many, the stuck-at fault model once again has proven to be efficient and effective in detecting and localizing defects in digital ICs.
   - this thesis

6. Electrical fault diagnosis is increasingly becoming a key factor for a fast ramp-up of new processes and products. Specifically new statistical diagnosis methods are going to have a significant contribution in getting to higher yields quickly.
   - this thesis

7. Failure analysis and fault diagnosis are undervalued activities. There is hardly to be found another domain in the silicon arena which requires a combination of skills in so many different areas like design, test, mathematics, physics, chemistry, etc.

8. The term failure analysis gives a too negative image. The focus is too much on the word failure, while the activity more on analysis. Hence, a better name would be device analysis.

9. A full paperless “economy” has hardly a chance in The Netherlands, given the amount of paper one has to “process” at every step he/she wants to make.

10. Things are as bad or good as you see them. So better look at them in a positive way.

11. The only real things we carry with us till the end are our feelings, our loves, our sufferings, our hatreds and our adversities. I’m asking myself: we, at the end of our journey, what we leave behind? I suppose we can leave some feelings, less hatreds, some sufferings but mainly love.
   - Nichita Stanescu

12. It is more difficult to write a good summary than a good article.