UNIVERSITY OF SOUTHAMPTON

FACULTY OF ENGINEERING AND APPLIED SCIENCE

DEPARTMENT OF ELECTRONICS AND INFORMATION ENGINEERING

# FAULT-ORIENTATED TESTING OF MOS CIRCUITS

A thesis submitted for the degree of Doctor of Philosophy
of the University of Southampton

by

NEIL BURGESS

April 1986

UNIVERSITY OF SOUTHAMPTON

## ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE

DEPARTMENT OF ELECTRONICS AND INFORMATION ENGINEERING

Doctor of Philosophy

# FAULT-ORIENTATED TESTING OF MOS CIRCUITS

by Neil Burgess

It is widely accepted that the most efficient method of testing digital integrated circuits is to use a fault-orientated approach. Assumptions are made about the faults that can occur in a circuit, and the circuit is tested for the absence of these assumed faults. Over the last twenty years one set of assumptions - the stuck-at fault model - has emerged as being particularly suitable for testing digital circuits made from TTL SSI/MSI components mounted on a pcb. However, the stuck-at fault model is also routinely used to test state-of-the-art MOS VLSI circuits, despite the lack of evidence for its suitability for this task.

In this thesis, the appropriateness of the stuck-at fault model for MOS circuit testing is assessed by identifying the common NMOS failure modes, and by using the SPICE circuit simulator to determine their assoicated fault-effects, as well as by direct analysis of defective NMOS circuits. A switch-level fault model is proposed that more accurately reflects the common MOS faults, and a classical test pattern generation algorithm - the D-Algorithm - is modified using a branch of graph theory called path algebras to provide a new method of automatic test pattern generation. The method is shown to be readily extendable to CMOS circuits and some insights into "layout for testability" are given.

CONTENTS

All diagrams and references have been placed at the ends of their respective chapters.

## ACKNOWLEDGEMENT

This project was carried out under a Science and Engineering Research Council CASE award in conjunction with British Telecom Research Laboratories, Martlesham Heath. It is a pleasure to acknowledge the continuing assistance, supervision and keen attention of my university tutor, Dr R I Damper, and Mr S J Shaw, Mr D R J Wilkins and Mr K A E Totton, all of British Telecom. I would also like to thank Dr B A Carré, of the Department of Electronics and Information Engineering, for his time and much helpful advice on the theory of path algebras. Finally, I am grateful to many other friends and colleagues both at the University and at British Telecom for their suggestions and criticisms - in particular I would like to mention Mr G Proctor and Mr D Ramasinghe of British Telecom for their assistance with the SEM work.

# CHAPTER 1 - THE NEED FOR FAULT-ORIENTATED TESTING

## 1.1 Introduction

As a result of the technological advances of the past two decades, the electronics industry is now capable of mass-producing digital integrated circuits consisting of thousands of gates. This development has resulted in the production of highly complex computer systems that are cheaper, more powerful, and more reliable than their predecessors. However, the problem of determining whether or not a chip has been correctly processed in the first place is proving to be both increasingly difficult and expensive. Nowadays, the generation, evaluation and application of a test input sequence using automatic test equipment accounts for a large proportion (at least 20%) of the chip manufacturing costs (ref 1), and this proportion is growing with the complexity of the chips being produced today.

In its early days, digital circuit testing was simple: every possible input combination was applied to the circuit (pcb) and the outputs were checked for correct responses. A combinational circuit with n inputs thus required $2^n$ test vectors to exhaustively test it, and a sequential circuit with n inputs and m additional memory elements required at least $n^{m-1}$ test vectors for it to be fully tested (assuming the circuit could be not initialised into a known state (ref 2)).

This exponential growth in the number of test vectors needed to test exhaustively a circuit with increasing numbers of memory elements in the circuit led to the exhaustive testing strategy becoming economically unfeasible.

In 1959, Eldred devised the single stuck-at fault model to facilitate the rigorous testing of digital circuit boards (ref 3). Using a "fault-orientated" testing strategy, assumptions were made about the faults that were possible in a circuit and a set of test vectors generated to cover only these assumed faults. The model operated on the gate-level description of the circuit and postulated that, in the presence of a fault, one, and only one, gate input or output became permanently fixed at either logical 1 ("stuck-at-1") or at logical 0 ("stuck-at-0"). The test input sequence could be readily generated using this model such that, should a stuck fault be present, its fault-effect would be propagated along a "sensitive path" through the circuit to produce a functionally incorrect value at an output of the circuit. In addition, the test input sequence's fault coverage (or "effectiveness") could be evaluated by determining how many of the "stuck-at" faults were tested for by the sequence.

Although the stuck-at fault model most readily modelled faulty electro-magnetic relay armatures that had become permanently "stuck", the model proved to be both efficient and cost-effective for testing logic circuits made up of discrete components mounted on a pcb. The components (transistors, diodes, resistors) would be tested individually before assembly, and the board checked for connectivity. The assembled board would then be tested in the almost certain knowledge that any faults on the board would have been introduced by the assembly process. In this way, the logic circuit could be adequately tested using a test input sequence whose length was considerably less than that of an exhaustive test set. The stuck-at fault model was successful in this application for two main reasons:

* several assembly-related faults did produce "stuck" nodes; for example, a solder splash bridging a track to a power rail, a floating input to a DTL gate ("stuck-at-1"), or a shorted output transistor in a DTL gate ("stuck-at-0");

* other faults (broken leads, components inserted with wrong orientation) would have such catastrophic effects that any test sequence would be almost certain to uncover them.

Furthermore, the stuck-at fault model still proved to be adequate when SSI and MSI TTL integrated circuits became available, provided that tests for bridging faults (ref 4) on both the board and the chips were included in the test sequences. (As with DTL circuits, a floating input to a TTL circuit is effectively a "stuck-at-1" node, and an output transistor open-circuit will also produce a stuck node). The stuck-at fault model thus became widely used as it was both conceptually simple, and it reflected many of the possible failures that could occur in logic circuits.

In 1966, J P Roth developed the D-Algorithm (ref 5), which formalised the procedure of generating a test input vector to cover a specified stuck-at fault in any combinational circuit. Although the D-Algorithm is unable to generate tests for sequential circuits without its being severely modified, the ability to generate and evaluate tests automatically on digital computers using the D-Algorithm further increased the cost effectiveness of testing circuits using the stuck-at fault model.

Recently, however, there have been growing doubts over the appropriateness of the single stuck-at fault model for testing MOS LSI circuits, and these doubts are now discussed in detail.

## 1.2 Testing LSI Circuits (refs 6, 7)

The advent of LSI technology has brought with it major testing problems relating to the sheer size and complexity of LSI circuits:

* testing may not be performed in stages because the individual sub-circuits (adders, counters, multiplexors etc) and their interconnections are now integrated on a single chip;

- access to the sub-circuits on the chip has been greatly reduced as a result of the limited pin-out, and because one sub-circuit is often only accesible via a second sub-circuit;

- the increase in the number of gates on a chip both greatly increases the effort (man/computer time) involved in the generation and evaluation of the test input sequence, and increases the likelihood of multiple faults.

These points may be illustrated by considering a typical LSI chip of 3,000 2-input gates with 40 pins. There would be 18,000 possible gate input and output stuck-at faults; a number that, by fault-equivalence and fault-collapsing, might be halved for testing purposes. In order to evaluate the test input sequence, 9,000 copies of the fault-free circuit, each with a single stuck-at fault inserted, would need to be simulated, and the full test input sequence applied to each. This is equivalent to 9,001 testings of the original circuit, and the scale of the problem is readily appreciated. (The fact that multiple faults are more likely to occur does not matter unduly. They would almost certainly be detected, using the single stuck-at model, albeit incorrectly diagnosed, and since integrated circuits are irreparable, the inaccuracy of the diagnosis is unimportant for most applications).

To complicate matters, the observability and controllability of internal circuit nodes would be limited by the relatively small number of pins (primary inputs and outputs), since most nodes on the chip will not be directly accessible to a tester via the primary inputs and outputs. Generating a test to check an individual node deep within a chip could thus prove to be difficult, if not impossible, and would certainly prove to be expensive in terms of man/computer effort. The bus structure of many LSI circuits can prove to be a help to the tester here: it is relatively easy to control and observe a sub-circuit via a bus; however, if the control or observation of the sub-circuit is via a second sub-circuit (or its operation is closely dependent upon a second sub-circuit) the testing problems are made considerably more difficult.

Furthermore, the implementation details of LSI circuits are not usually disclosed by a manufacturer. For example, the user's manual that accompanies a microprocessor will describe the instruction set, the architecture of the processor at register level, and some system timing details. Lack of gate-level information obviously eliminates the use of the gate-level stuck-at fault model, besides which different manufacturers may implement the same LSI circuit in different ways.

These problems of large gate count, chip complexity, and lack of implementation detail led to the increased attractiveness of an alternative testing philosophy - functional testing -

whose appearance more or less coincided with that of the microprocessor. It is still widely used - despite its shortcomings - on complex circuits in the absence of any more efficient method. The functional approach simply attempts to check that the device-under-test functions as it ought to, thus bypassing the need for testing for specific assumed faults. For example, a functional test for a counter would be to initialise it, and then clock it through its states, checking that they occur in the specified sequence. However, consider an adder circuit: if the adder adds 3 and 8 correctly is that an adequate test? Should other combinations of numbers be used, and if so, which ones?

Random access memories have a highly regular structure and a very simple function. A functional test sequence could be written to check that a 1 or an 0 may be stored in any given location; however, such a test sequence could well miss all the pattern sensitive faults that are known to exist in memories. Similarly, a functional test could be written to check that when a 1 or a 0 is written into a location, data in no other locations are corrupted. Such a test set would prove to be prohibitively long unless assumptions were made about which other locations could be corrupted, and if these assumptions are made arbitrarily, in the absence of any information about the system's structure, the fault coverage of the test set would be degraded (ref 8).

The basic problem with the functional testing approach is that no assumptions are made about the ways in which faults in the circuit will manifest themselves, and thus the behaviour of a circuit containing a fault is completely unspecified. This results in test sets that are either extremely long or that cannot be guaranteed to have a high fault coverage. Recently, a more analytical functional testing approach has been developed (ref 9) in which the LSI circuit-under-test is modelled using graph theory on a register-transfer-level description of the circuit. Fault models are derived for the different functions of the different sections of the circuit. For example, the "data transfer" fault model includes "stuck lines" and bridging between adjacent lines, while the "control" fault model allows for erroneous instructions, including "No Operation", to be performed instead of or as well as the designated instruction. When this approach was applied to an eight-bit microprocessor, a hand-generated test set covered 96% of the gate stuck-at faults. This approach seems most promising; unfortunately, a test set that covers 100% of the stuck faults in a logic circuit does not necessarily test for certain common MOS circuit failures.

## 1.3 The problems of using the stuck-at fault model for MOS LSI testing

The way to avoid the problems associated with functional testing is to make assumptions about the ways in which the circuit-under-test can fail, and then test for the assumed faulty behaviour. This process of "fault modelling" may take place at any level of circuit representation (transistor, gate, function, system) but, obviously, the lower the level at which the fault model operates, the greater the accuracy with which the real faults in the silicon may be modelled and, it is tacitly assumed, the greater the man/computer effort required to generate a test input sequence (ref 10).

The single stuck-at fault model is an example of one such set of assumptions, operating on the gate-level representation of a circuit, and it forms the basis of most current test pattern generation algorithms. However, there have been growing doubts over the validity of the stuck-at fault model for testing MOS LSI circuits, for two main reasons (refs 11, 12):-

- the stuck-at model operates on the gate-level representation of a circuit, which is not topologically equivalent to the layout of the circuit in silicon;

- there is no reason to expect that the majority of the possible physical faults on a silicon chip will produce stuck notes on an equivalent gate-level schematic.

The following example illustrates these points. Figure 1.1 shows the NMOS implementation of an XNOR gate, as used in the Z80. All four input vectors 00, 01, 10 and 11 are needed to cover all possible transistor stuck-on and stuck-off faults in the circuit. However, a test set that covers all the Boolean gate input and output stuck-at faults requires only 3 of the 4 inputs to be applied. (In fact, it requires any 3 of the 4 to be applied). Thus, a test set that covers all the stuck-at faults in a circuit need not have covered all the possible physical faults in the circuit. Furthermore if transistor X is stuck off or one of its contacts becomes open circuit then the circuit becomes a latch, since on applying the input ab = 11, the output retains its previous value, instead of being pulled up by transistor X. The problem is that information concerning the structure of the circuit (the layout and interconnection of the transistors) is lost by using the gate-level representation, and thus potential faults on the chip may not be readily inserted in the gate-level diagram, and as a result are not explicitly tested for.

## 1.4 The D-algorithm

As stated above the D-Algorithm was devised in 1966 by J P Roth of IBM to facilitate the automatic generation and evaluation of tests using the stuck-at fault model. In this section, I describe the algorithm's operation and notations.

The D-Algorithm uses a five-valued algebra to model both fault-free and faulty logic circuits. The five values are the conventional logic values 1, 0, and X ("don't care"), and two special symbols, D and D' (read "D-bar"), which are used to model discrepancies between a logic circuit's fault-free behaviour and its behaviour in the presence of a fault. The D symbol is used to denote a node whose fault-free logic value is a 1, but whose value in the presence of a fault is a 0. The D' symbol has the converse definition. For example, if an inverter's output were to be tested for a stuck-at-0 fault, its input would be set to 0 and its output to D.

Roth uses a "cube notation" for the algorithm, where a cube is an ordered set of n symbols identifying the logic values on each of the n nodes of a logic circuit being simulated. For example, figure 1.2 shows a simple logic circuit and the cube describing its current status. Each entry in the cube corresponds to a numbered node in the logic circuit and its associated logic value. The cube is thus a compact and convenient means of recording a circuit's behaviour. (Note that the circuit nodes are numbered such that any gate's output is described by a higher number than any of its inputs, and that fan-out "branches" are numbered separately from the "trunk" node).

To derive a test for a node on a logic circuit, a single fault is inserted into the network (using D or D'), the fault-effect propagated through the logic circuit to a primary output by setting appropriate gate inputs to 1 or 0 (usually called the "D-drive"), and finally, these "fixed value" nodes checked for self-consistency: in other words that no node is required to be at 1 and 0 simultaneously.

A particular set of cubes is associated with each of these processes. A "failure D-cube" is used to insert a fault, where the failure D-cube consists of a cube describing a single gate whose inputs are fault-free (taking the values 0 or 1) and whose output is faulty (D or D'). The failure D-cubes for a NAND gate are shown in figure 1.3.

"Propagating D-cubes" and "Non-propagating D-cubes" are used in driving the fault-effect to a primary output. These are single gate cubes which have at least one D or D' on the inputs, and, in the case of a propagating D-cube, a D or D' on the output, or a 1 or a 0 on the output for a non-propagating D-cube. The propagating and non-propagating D-cubes for a NAND gate are shown in figure 1.3 as well. During the D-propagation the test cube is built up from its initial state of all "X"'s by repeated "D-intersection" operations, in which the test cube is intersected with a propagating D-cube to propagate the fault-effect one gate nearer to the primary output. An example is shown in figure 1.4 where the test cube reads $(D_1\ 1_2\ D'_3\ X_4\ X_5)$ and the propagating D-cubes for a NAND gate are as shown in figure 1.3. The cubes $(1_3\ D_4\ D'_5)$, $(D_3 1_4 D'_5)$ and $(1_3\ D'_4\ D_5)$ etc may not be used since in each case the value on node 3 in the D-cube does not match the node value in the test cube (the X's may be "overwritten" by any logic value). The D-cube $(D'_3\ D'_4\ D_5)$ may not be used because Ds or D's may not be arbitrarily introduced to propagate a fault-effect. Therefore the D-cube $(D'_3\ 1_4 D_5)$ must be used and the resultant test cube is ($D_1$ $1_2$ $D'_3$ $1_4$ $D_5$). Also associated with the D-propagation is the "D-frontier" - a list of gates which have at least one D or D' on their inputs and an X on their output. This list comprises those gates that have not yet had a fault-effect propagated through them, and which therefore need to be intersected with the test cube.

The non-propagating D-cubes are used if a fault needs to be blocked, such as might happen in a circuit containing reconvergent fan-out. No gate can propagate a D and D' simultaneously, in which case a non-propagating D-cube must be used to change one of the potentially faulty inputs to a fault-free input by forcing a previous gate's output to 1 or 0.

The consistency check uses the "singular cover" of the logic gates - a compact form of a gate's truth table that uses X's as well as 1's and 0's. The singular cover of a NAND gate is again shown in figure 1.3. The algorithm will now be demonstrated by generating a test for node n6 of the circuit in figure 1.2 stuck-at-1.

The failure D-cube is ($0_2$ $0_5$ $D'_6$) since node 2 fans out to nodes 5 and 6 of which only node 6 is faulty (in TTL technology, this would correspond to n6 open circuit).

The test cube is:

$$(X_1\ 0_2\ X_3\ X_4\ 0_5\ D'_6\ X_7\ \ldots\ldots\ X_{13}).$$

The D' on n6 must now be propagated to a primary output by repeated intersection with the propagating D-cubes. First the fault must be propagated through gate 2. This is achieved by setting n7 to 0 (this also sets n8 to 0), which produces a D on n10. This corresponds to intersecting the test cube with the propagating D-cube $(D'_6\ 0_7\ D_{10})$ to give the resultant test cube:

$$(X_1\ 0_2\ X_3\ X_4\ 0_5\ D'_6\ 0_7\ 0_8\ X_9\ D_{10}\ X_{11}X_{12}\ X_{13}).$$

The D on n10 is in turn propagated by setting n11 to a 0, thus forcing n12 to D', and the fault is finally propagated to the primary output n13 by setting n9 to 1. The test cube now reads:

$$(X_1\ 0_2\ X_3\ X_4\ 0_5\ D'_6\ 0_7\ 0_8\ 1_9\ D_{10}\ 0_{11}\ D'_{12}\ D_{13})$$

The fixed values on nodes 2, 5, 7, 8, 9, and 11 introduced by the D-drive must now be checked for their consistency. This is done by examining the singular covers of the gates relevant to ensure that there are no discrepancies in the test cube. The algorithm starts with the highest-numbered node. Node n11 should be at 0, which implies n8 or n4 should be at 1. Node 8 is already at 0, so n4 must be set to 1, and added to the fixed value list, and n11 removed from the list. Node 9 needs to be a 1, which in turn implies (from the singular cover) that n1 or n5 must be 0. Node 5 is already 0, and so n1 is arbitrarily set to 1 and added to the list of fixed value nodes, and n9 removed. The node list now reads $\{1, 2, 4, 5, 7, 8\}$. Nodes 7 and 8 are both 0, which means n3 must be 0; n5 is 0 and thus n2 should be a 0 also, which it already is. Nodes 5, 7, and 8 are removed from the list, and n3 added. The list now consists of nodes 1 to 4, all of which are primary inputs. The consistency check has been successful, and the algorithm halts having generated the test input vector abcd = 1001 to test for n6 s-a-1. The final test cube reads

$$(1_1\ 0_2\ 0_3\ 1_4\ 0_5\ D'_6\ 0_7\ 0_8\ 1_9\ D_{10}\ 0_{11}\ D'_{12} D_{13}),$$

and thus the test has tested for nodes 10 and 13 s-a-0 and n12 s-a-1 as well. These stuck-at faults are removed from a list of target faults for the whole circuit and the entire operation is repeated for a new fault selected from the amended list of target faults. This immediately shows a bonus gained from this method : namely, many faults are covered without tests being generated for them thus saving a great deal on computer run-time in test pattern generation and evaluation.

An additional attraction of the D-Algorithm, although the example did not show this, is that it handles back-tracking procedures well. If a nodal logic conflict arises (for example, the consistency check may require a node to be set to 1 when it has already been set to 0 by the D-drive), the algorithm back-tracks to its last decision point, where a choice had been made for whatever reason, and tries a different option. It carries on in this way until either a solution has been found, and a test input vector generated, or no solution is found, which the algorithm is able to flag to the test engineer.

## 1.5 The overall aims of the project

So far, this chapter has reviewed the need for fault-orientated testing, and described its most succesful implementation - the D-Algorithm. In this section, the aims of the project are considered, and a preview of the structure of the thesis is given.

The main aim of the project was to assess the appropriateness and validity of the single stuck-at fault model for testing MOS logic circuits. This was achieved by determining what the common faults in MOS circuits and their relative incidence rates were, and by studying their fault-effects in MOS logic circuits. An extensive literature survey formed the major part of this investigation, with further information gleaned from personal contact with process and reliability engineers at British Telecom. (A study of faulty MOS LSI circuits using failure analysis techniques to generate this information would have proved to have been extremely time-consuming, as an MOS LSI circuit typically takes a fortnight to analyse. Even over three years, the data collected on MOS circuit faults would have had little statistical significance, owing to the small sample size.)

The fault-effects of the faults in the silicon were then studied by using the SPICE circuit simulator, and by conducting fault analyses of failed MOS test circuits. Both these methods were quick and simple compared to LSI circuit failure analysis, in which a chip is effectively dissected. Having thus built up a clear picture of the nature of the fault-effects in MOS circuits, the stuck-at fault model was evaluated by examining the relationship between the fault model and the in-circuit fault-effects.

If the conventional single stuck-at fault model proved to be inadequate in modelling many of the faults that did occur in practice, it would need to be modified to cope. If this was not possible, the stuck-at fault model would need to be replaced by an alternative simple yet physically realistic fault model that would be able to cover a much greater proportion of MOS faults.

Secondary aims of the project were to use the knowledge gained about faults and their associated fault-effects to determine test conditions (ambient temperature, supply voltage, etc) that readily uncover MOS integrated circuit faults. In addition, design for testability rules that take account of the common faults found in MOS structures were to be defined to ensure that the MOS fault model or any suitable replacement would remain physically realistic.

The thesis has the following structure:-

Firstly, in Chapter 2, a review of MOS failure modes is presented including VLSI short-channel device failure mechanisms. Chapter 3 describes hardware and software investigations of the fault-effects introduced by the failure mechanisms described in Chapter 2. Chapter 4 is an evaluation of the stuck-at fault model in the light of the fault-effects described in Chapter 3. Chapter 5 describes an NMOS test pattern generation system that uses graph theory - path algebras - to model realistically the faults that occur in MOS circuits. The thesis concludes with a discussion of related topics on testing, together with some suggestions for "layout for testability" techniques.

# REFERENCES

1 J. R. Hines (1983).
Predicting IC costs.
Semiconductor International, June 1983, pp 76-80.

2 E. F. Moore (1956).
Gedanken-experiments on sequential machines.
<u>in</u> "Automata Studies", ed C F Shannon and J McCarthy,
(pub Princeton University Press) pp 129-153.

3 R. D. Eldred (1959).
Test routines based on symbolic logic statements.
Journal of the Association for Computing Machinery, <u>6</u>,
pp 33-36.

4 K. C. Y. Mei (1974).
Bridging and stuck-at faults.
IEEE <u>TC-23</u>, pp 720-727.

5 J. P. Roth (1966).
Diagnosis of automata failures: a calculus and a method.
IBM Journal of Research and Development, <u>10</u>, pp 278-281.

6 R. G. Bennetts (1982).
An Introduction to Digital Board Testing.
(Pub Arnold).

7 M. S. Abadir and H. K. Reghbati (1983).
LSI testing techniques.
IEEE Micro, February 1983, pp 34-51.

8 J. A. Abraham (1981).
Functional level test generation for complex digital systems.
Proc 1981 IEEE Test Conference, pp 461-462.

9 S. M. Thatte and J. A. Abraham (1980).
Test generation for microprocessors.
IEEE Trans, TC-29, pp 429-441.

10 Z Barzilai, L Huisman, G Silberman, D Tang and L Woo (1983).
Simulating pass transistor networks using logic simulation machines.
Proc 20th IEEE Design Automation Conference, pp 157-163.

11 J. Galiay, Y. Crouzet and M. Vergniault (1980).
Physical vs logical fault models for MOS LSI circuits.
IEEE Trans, TC-29, pp 527-531.

12 P. Banerjee and J. A. Abraham (1984).
Characterisation and testing of physical failures in MOS logic circuits.
IEEE Design and Test, August 1984, pp 76-86.

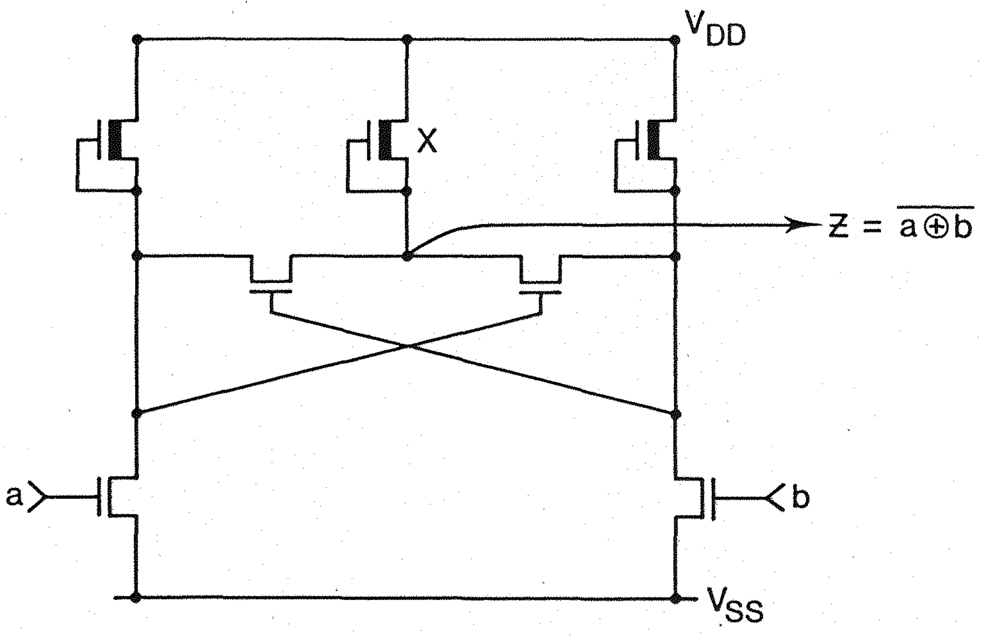
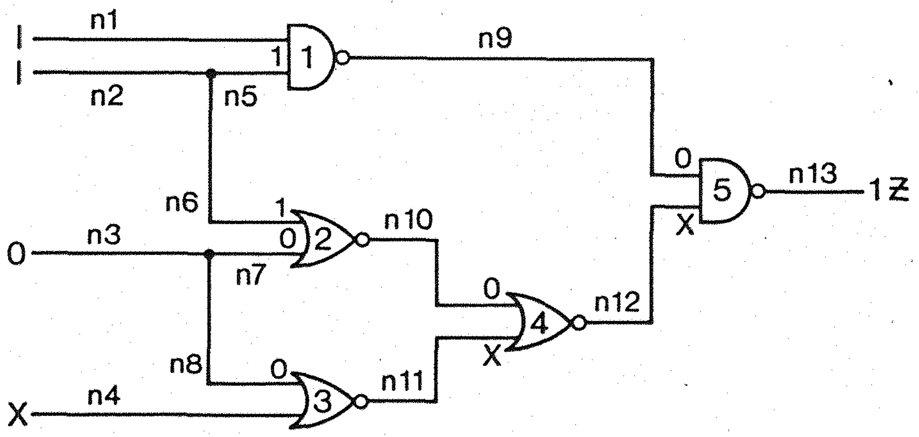
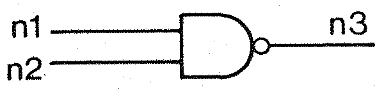


Figure 1.1 Z8Ø implementation of the exclusive-nor gate



cube: $\left( 1_1 1_2 0_3 X_4 I_5 I_6 0_7 0_8 0_9 0_{10} X_{11} X_{12} I_{13} \right)$

Figure 1.2 Simple logic circuit and cube



Failure D-cubes:

$(0_1\ 0_2\ D_3)$ , $(1_1\ 0_2\ D_3)$

$(0_1\ 1_2\ D_3)$ , $(1_1\ 1_2\ D'_3)$

Propagating D-cubes:

$(D_1\ 1_2\ D'_3)$,$(D'_1\ 1_2\ D_3)$

$(1_1\ D_2\ D'_3)$,$(1_1\ D'_2\ D_3)$

$(D_1\ D_2\ D'_3)$,$(D'_1\ D'_2\ D_3)$

Non-propagating D-cubes:

$(D_1\ 0_2\ 1_3)$,$(D'_1\ 0_2\ 1_3)$

$(0_1\ D_2\ 1_3)$,$(0_1\ D'_2\ 1_3)$

$(D_1\ D'_2\ 1_3)$,$(D'_1\ D_2\ 1_3)$

Singular cover:

$(0_1\ X_2\ 1_3)$,$(X_1\ 0_2\ 1_3)$,$(1_1\ 1_2\ 0_3)$

Figure 1.3 Nand gate with its D-cubes and singular cover

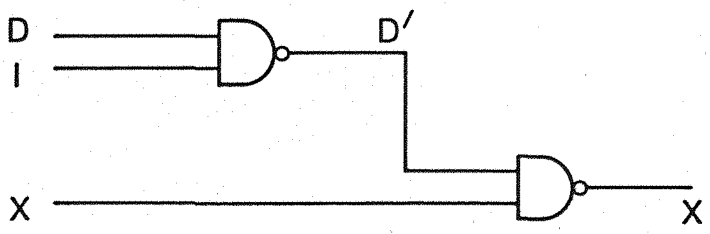


Figure 1.4 Simple circuit to demonstrate cube intersection

# CHAPTER 2 - A REVIEW OF MOS FAILURE MECHANISMS

## 2.1 INTRODUCTION

An extensive literature search was conducted to identify the major MOS failure mechanisms and to assess their relative incidence rates. Table 1 shows that these rates vary with manufacturer, circuit complexity and type, and technology. (Categories (b) and (d), and (c) and (e), can be used to describe the same mechanism - for example, excess p-n junction reverse leakage current, or metallisation open circuit - but this does not fully account for the wide variations shown). They can also vary randomly with time; that is, the same circuit produced by the same manufacturer on the same process line will exhibit different incidence rates of the various failure mechanisms at different times. This point confirms the decision not to attempt to generate this data by failure analysis of MOS LSI circuits.

Furthermore, there is no clear dichotomy between faults that are introduced during the chip fabrication sequence, and those that become effective during the field operation of the chip. For example, improper step coverage by a metal track, and moisture-induced corrosion of a metal track both produce open circuits in the track.

In this chapter, details of the common MOS failure mechanisms are presented, with some discussion of their likely fault-effects on digital circuit behaviour. The mechanisms have

| Reference Number | Technology | Circuits examined | Incidence of Failure Mechanism (%) | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | (a) | (b) | (c) | (d) | (e) | (f) | (g) |
| 1 | NMOS | microprocessor : infant failures | 2 | - | 53 | 20 | - | - | 25 |
| 2 | ?MOS | various : field returns | 57 | 1 | 8 | - | 12 | 14 | 8 |
| 3 | NMOS | DRAM : field returns and life-tested | 66 | 17 | 1 | - | 6 | 8 | 2 |
| 4 | CMOS | SSI : life-tested | 7 | 16 | 69 | - | - | - | 8 |
| 5 | CMOS | various : field returns | 25 | 35 | 20 | - | - | 15 | 5 |
| 6 | CMOS | SSI : life-tested | - | 75 | 1 | 3 | - | 19 | 1 |
| 7 | PMOS | RAM : life-tested | 25 | 45 | - | 10 | - | 6 | 14 |
| 8 | NMOS | various : various | 48 | 13 | 3 | - | 5 | 10 | 21 |
| 9 | ?MOS | memory : infant failures, manufacturer A | 51 | 24 | - | - | - | 7 | 18 |
| 9 | ?MOS | memory : infant failures, manufacturer B | 53 | - | 2 | - | - | 27 | 18 |
| 10 | NMOS | 16K DRAM : infant failures manufacturer C | 36 | - | 24 | 17 | 9 | 2 | 12 |
| 10 | NMOS | 64K DRAM : infant failures manufacturer C | 31 | - | 13 | 19 | 17 | 1 | 19 |
| 11 | ?MOS | various : life-tested | 22 | - | 32 | 2 | - | 3 | 41 |

Failure mchanisms: (a) oxide shorts

(b) surface instabilities

(c) metallisation faults

(d) diffusion faults

(e) photolithography - related problems

(f) packaging- and assembly-related failures

(g) others and miscellaneous (gross defects, not determined)

TABLE 1: RELATIVE INCIDENCE RATES OF MOS FAILURE MECHANISMS (refs. 1-11)

been divided into two categories:- those which have always been present in MOS integrated circuits, and those which have been introduced or exacerbated by the reduction in size of MOSFET's in the move towards LSI circuits.

## 2.2 COMMON MOS FAILURE MECHANISMS (refs 12-14)

### 2.2.1 Photolithography-related failures

MOS integrated circuits are fabricated as a sequence of up to nine patterned layers on the surface of a silicon wafer. These patterns are transferred from a set of masks to the silicon surface by a photolithographic process that has become standardised in industry (ref 15).

Faults may be introduced during the photolithography process by any of the following mechanisms:

Mask defects (extra or missing details), dirt and photoresist impurities can result in open and short circuits, poor ohmic contacts between layers of interconnect, and increased leakage currents due to the creation of extra conductive paths in the substrate. These defects occur randomly across the mask and thus whether or not a defect has any effect on a circuit's performance depends on its location.

Dirt on the wafer surface may also affect the etching process by creating defects in the photoresist or the layer of material being patterned, that can result in pinholes being formed in the layer. Other etching problems include under-etching, leading to high resistance or open circuit contact windows, and incomplete diffusions; and over-etching, leading to extended diffusion areas whose associated depletion layers may short together.

Finally, the linewidth of a detail on the mask and the linewidth of the same detail on the processed chip can differ by 1 μm or more. This is due to the wet, isotropic etching of the material under the photoresist after the resist has itself been etched. This variation in linewidth affects the gain of the MOSFET's on the chip, although it may be allowed for by making the relevant shapes on the mask a little larger or smaller. However, the final size of the processed transistor is not precisely predictable and thus the transistor's gain (or drive capability) is not guaranteed. This problem is exacerbated at smaller linewidths because the error of 1 μm does not scale with the linewidth of the process technology unless anisotropic etching techniques are used.

### 2.2.2 Gate oxide-related failures (ref 16)

Many device failures can be attributed to oxide-related defects such as:-

* ionic contamination of the gate oxide;

*    charge accumulations at the oxide-silicon interface ("slow trapping");

*    lateral spreading of charge across the field oxide;

*    pinholes in the gate oxide layer resulting in its subsequent dielectric breakdown;

This last mechanism is usually caused by faulty processing or by particulate contamination of the air or process chemicals and gases, and results in shorts between the gate electrode and the channel, source or drain of a device (ref 17). (Electrical screening at higher-than-rated value is performed to eliminate devices which have oxide regions of only marginal quality).

The contamination of silicon dioxide by mobile alkali ions (particularly sodium) has been a major problem in MOS integrated circuit manufacture. Sodium is one of the most abundant elements and appears in the oxide after the etching of the aluminium layer, which itself contains sodium ions. Under the influence of the applied gate voltage, the alkali ions move toward the oxide-silicon interface under the gate regions, where they accumulate in the potential well (Figure 2.1). This accumulation of positive charge induces negative charge at the 'top' of the silicon in the channel of the NMOS device, resulting in a decrease in the device's threshold voltage or an increase in the subthreshold conduction.

Two other charge accumulation mechanisms have been identified. Firstly, a fixed surface-state charge is always produced because of the way in which the thermal oxidation process terminates the silicon lattice and forms the oxide. This fixed charge is positive, very close to the interface and its magnitude is independent of the oxide thickness. This phenomenon is well-known and taken into account in calculating threshold voltages of transistors. The second mechanism - "slow trapping" - involves interfacial trap sites that capture holes and which are believed to be caused by incomplete silicon-oxygen bonds. These traps can capture charges such as are produced by thermal electron-hole generation or by ionising radiation in the oxide, or charge carriers that have tunnelled from the silicon surface. Both these mechanisms result in a reduced threshold voltage or increased subthreshold conduction for the same reasons as were mentioned in connection with the ionic contamination.

Under certain circumstances, the silicon dioxide 'upper' surface may become conductive as a result of positive charge spreading out from positively biassed metal lines. This positive charge is transported by lateral ion movement or by the surface becoming conductive in the presence of moisture. In this latter case, leakage currents between neighbouring conductors can be created. Usually, however, the charge induces extended inversion layers in the substrate below the field oxide producing either high leakage currents (away from the channel) or a conductive path between two diffused regions, resulting in a short circuit.

### 2.2.3 Metallisation-related failures

The following metallisation failures have all been reported:-

- faulty etching of the metallisation;

- micro-cracks in the aluminium lines where an oxide step is crossed;

- electromigration failures - a high current density phenomenon whereby the metal atoms in the conductor migrate due to the force exerted on them by the large numbers of charge carriers colliding with them. This produces breaks in the metal track at a point where a photolithography fault has substantially narrowed a track thus causing an increase in current density at that point;

* corrosion of the metal as a result of moisture within the chip;

* contact window failures due to the mask misalignments or the Al-Si interactions (so-called 'alloy spikes') that can take place, resulting in high resistance or open circuit contacts, or contacts that are shorted to the substrate.

All of these failures may be reasonably characterised as shorts or opens in the metal lines.

### 2.2.4 Other Failures

Silicon bulk crystal defects are another source of circuit failures and are either introduced during crystal growth or else are process-induced (ref 18). These defects include dislocations, stacking faults, swirl defects and various point defects. Their major fault-effect in MOS circuits is junction leakage (in bipolar technologies, they produce transistor "pipes") leading, for example, to charge-storage failure in a dynamic RAM (ref 19). However, they also produce a reduction in channel carrier mobility and an increase in the threshold voltage of a device, although these last two effects are only significant in submicron VLSI transistors.

Cracks or pits in the passivation layer are, in general, caused by mishandling or by a mismatch between the thermal expansion coefficients of silicon and the glass passivation layer, thus introducing stress and strain during subsequent thermal processing. Defects of this kind are usually spotted during visual inspection before packaging and, as such, tend not to be a testing problem.

Packaging- and assembly-related failures are also not a major test problem because their effects are, on the whole, so catastrophic, resulting in shorts or opens. Examples include the 'purple plague' (a gold-aluminium reaction that results in an open circuit instead of a bond between the silicon chip and integrated circuit external pin), some forms of corrosion that are related to the hermeticity of the encapsulation, and the chip breaking as a result of mechanical shock.

## 2.3 COMMON FAILURES OF SCALED-DOWN DEVICES

The increasing integration of devices onto silicon chips has been achieved by reducing the dimensions of the devices (minimum feature size 3-5 $\mu$m) and by developing new processing techniques. These trends introduce new failure mechanisms as well as exacerbating existing ones, but the outlook is not entirely gloomy. For example, high performance MOS technology has been developed by reducing the channel length, gate oxide thickness and junction depth of the MOS device while retaining the same supply voltages to maintain compatibility with other MOS technologies and to keep noise problems to a minimum (ref 20). The increased field across the gate oxide will thus accelerate time-dependent breakdown or degradation of MOS devices, as discussed above. However, the use of polysilicon gate electrodes and of phosphosilicate glass 'gettering' layers with silicon nitride reduces both the extent of the ionic (Na+) contamination and its mobility.

### 2.3.1 Failures Introduced by LSI Processing Technology

LSI circuit fabrication would not have been achieved without ion implantation being used instead of diffusion techniques. Using ion implantation, it became relatively easy to control the low-concentration doping of the silicon substrate necessary for threshold-voltage control of MOSFET's, and it became possible to use "self-alignment" techniques rather than photolithiography steps, thus eliminating the photolithography-related failures from certain steps of the fabrication process (Section 2.2.1). However, implantation, together with the dramatic heat changes used in silicon chip processing, produces dislocations, stacking faults and other crystallographic faults in the silicon substrate. These defects are often 'decorated' by impurity atoms (for example, copper, iron, gold, etc) and, besides producing excessive p-n junction leakage currents, can (if particularly numerous) increase the threshold voltage and reduce the transconductance of a MOSFET as noted earlier.

Furthermore, when silicon dioxide is thermally grown on a wafer surface, 45% of the resultant oxide layer grows into the wafer. Crystallographic damage at the wafer surface is thus included in the gate oxide layer as a result, providing a source of defects that make the gate oxide prone to dielectric breakdown at any weak points thus introduced (ref 21). This phenomenon has only become evident in LSI circuits because the thinner gate oxides have meant that increased electric fields appear across them, making the oxides more prone to breakdown.

Anisotropic etching techniques have become widely used in LSI integrated circuit fabrication because they greatly reduce the 'sideways etching' that leads to large discrepancies between the linewidths on the mask and those on the chip. These "dry etching" techniques use high energy reactive ion bombardment of the silicon wafer surface to remove unwanted material, instead of acid bath "wet etch" techniques, thus providing greater directional control over etch profiles. However, anisotropic etching has its associated problems, the main one being that 'fillets' may be left between tracks over a step, thus shorting them together (see Fig 2.2). This is a particular problem in the etching of the polysilicon layer, where anisotropic etching is used to make sure that the transistor dimensions on the chip do not differ from those on the mask. In addition, the high energy ion bombardment induces crystallographic damage at the wafer surface thus introducing device performance degradations as described above (ref 22).

### 2.3.2 Failures in LSI Metallisation

The use of a smaller linewidth for the metallisation exacerbates existing failure modes such as electromigration and contact window failures (ref 23). The use of multi-level metallisation schemes has given some of these an added significance. In order to avoid electromigration problems in the smaller lines, the aluminium tracks are made 'taller': this means that the second-level metal lines have to cross steeper, bigger, oxide steps, thus increasing the chances of a microcrack.

The two levels of metallisation are connected by contact windows called "vias". Any contamination or oxidation of the exposed aluminium in the first layer will produce high-resistance contacts or smaller-than-expected "via" areas leading to electromigration-related failures. The smaller contact windows between the first metal layer and the silicon also increase the incidence of failed contacts, but the use of metal silicides, or alloys, together with heat-treatment techniques has reduced the likelihood of this failure's occurrence (ref 24). Breakdowns of the dielectric between conducting levels may also occur (ref 25).

### 2.3.3 Failures in LSI Memories

Alpha-particle-induced errors have received wide attention since their discovery in dynamic RAM's (ref 26). An alpha-particle entering the silicon substrate with an energy of 5 MeV penetrates to an average depth of 25 $\mu$m, generating some 2.5 million electron-hole pairs along its path as it loses energy. In n-channel dynamic RAM's, the electrons collect in the storage cell and the holes disperse in the substrate and, as a result, the information stored (as presence or absence of charge) may be lost because of the smaller amounts of charge used in larger RAM's. These 'soft' errors are random, recoverable (if detected), and it is only possible to test a cell's liability to be so affected.

It has also been found that interactions among neighbouring memory cells are common. These interactions are pattern sensitive - that is, dependent on the data being stored - and are so common that testing strategies have been devised to test for them (WALKPAT, GALPAT, etc). The physical mechanism causing these interactions is capacitative coupling between

between physically-adjacent memory cells, and they have appeared in memories because memory processing technology is a relatively advanced art, utilising the smallest linewidths. These interactions are starting to appear in other devices - for example, the register arrays of microprocessors that have passed their manufacturers' quality assurance tests (ref 27) - and will become more and more common as the scale of integration increases.

### 2.3.4 Failures in Short-Channel VLSI Devices

(Refs 28-31)

There are three widely reported failure mechanisms in VLSI devices (feature size $\leqslant 1.5\,\mu m$) that result from the increased electric fields within the scaled-down MOSFET:

- hot electron injection into the gate oxide;
- drain-substrate junction avalanche breakdown resulting from the activation of the parasitic bipolar device;
- punchthrough.

In saturation, an NMOS transistor has a large electric field in the drain depletion region, which produces greatly accelerated electrons near the drain. Here, impact ionisation collisions scatter these 'hot' electrons which may have sufficient energy to surmount the oxide-silicon potential barrier and may then be trapped in the gate oxide. ('Hot' electrons may also be produced from thermal hole-electron pair generation in the substrate, from where some electrons (and holes) may tunnel into the gate oxide). Over a period of time, the trapped charge will cause instability in the form of a threshold voltage increase. The trap density in the oxide has been increased in VLSI devices as a result of radioactive damage caused by electron-beam lithography.

An npn parasitic bipolar transistor exists in the n-channel MOSFET structure (Fig 2.3), and in short-channel devices may become active, as follows. The above-mentioned impact ionisation collisions also produce hole currents that flow in the substrate (the electrons flow to the drain) and which can forward bias the source - substrate junction. If this happens, an excess electron current flows between the source and the drain resulting in the eventual avalanche breakdown of the drain-substrate junction, and an NMOS transistor whose drain is stuck at the substrate voltage.

The third short-channel effect is punchthrough, and is similar in effect to the bipolar "latch-up" phenomenon just discussed. The depletion layer associated with the drain spreads across the channel and reaches the source depletion layer, producing a large increase in the subthreshold leakage current.

In a long-channel device the drain and source depletion layers do not penetrate very far into the channel region, and thus the channel potential barrier is solely a function of the applied gate voltage along most of the device's length. However, in short-channel devices where the source and drain depletion regions have merged, the channel potential barrier is lowered resulting in an increased subthreshold leakage current. As the drain voltage is increased, extra charge is imaged in the source region, and produces a still larger subthreshold current.

Design rules exist to minimise these effects, but process variations in the channel length can lead to a fraction of the devices having a shorter channel length than the design rules allow, thus producing faulty devices.

REFERENCES

1 J. Galiay, Y. Crouzet and K. Vergniault (1980) Op cit.

2 Anon (1980).
How some Japanese integrated circuits fail. Electronics International, 53, 6, pp 142-143.

3 H. A. Batdorf (1978).
Reliability evaluation program and results for 4K DRAM.
Proc 16th Int. Rel. Physics Symp, pp 14-18.

4 P. Brambilla, F. Fantini, P. Malberti and G. Mattana (1981).
CMOS reliability.
Microelectronics and Reliability, 21, pp 191-201.

5 L. J. Gallace and H. L. Pujol (1976).
Failure mechanisms in COS/MOS integrated circuits.
Electronic Engineering, December 1976, pp 65-69.

6 G. M. Johnson and M. Stitch (1977).
Microcircuit accelerated testing reveals life-limiting failure modes.
Proc. 15th Int. Rel. Physics Symp, pp 179-195.

7 R. Pappu, E. Harris and M. Yates (1977).
Screening methods and experience with MOS memory.
Microelectronics and Reliability, 17, 1, pp 193-197.

8 C. G. Peattie, J. D. Adams, S. L. Carrell,
T. D. George and M. H. Valek (1974).
Elements of Semiconductor Device Reliability.
Proc. IEEE, 62, 2, pp 149-168.

9 D. S. Peck (1978).
New concerns about integrated circuit reliability.
Proc. 16th Int. Rel. Physics Symp, pp 1-6.

10 D. L. Crook and W. K. Meyer (1981).
Redundancy reliability.
Proc 19th Int. Rel. Physics Symp, pp 1-10.

11 Reliability Analysis Centre (1981).
"MDR-18: Memory/Digital LSI".

12 D. G. Edwards (1982).
Testing for MOS IC failure modes.
IEEE TR-31, pp 9-17.

13 N. D. Stojadinovic and S. D. Ristic (1983).
Failure physics of integrated circuits.
Physical Status Solidi (a), 75, pp 11-48.

14 J. Wood (1980).
Reliability and degradation of silicon devices and integrated circuits.
In 'Reliability and Degradation'. Ed. M. J. Howes and D. V. Morgan (pub. Wiley, London). pp 191-236.

15 J. Mavor, M. Jack and P. Denyer (1983).
Introduction to MOS LSI Design.
(Pub. Addison-Wesley).

16 N. E. Lycoudes and C. C. Childers (1980).
Semiconductor instability failure mechanisms review.
IEEE TR-29, 3, pp 237-248.

17 J. M. Duffalo and J. R. Monkowski (1984).
Particulate contamination and device performance. Solid State Technology, March 1984, pp 109-114.

18 K. V. Ravi (1981).
Imperfections and Impurities in Semiconductor Silicon.
(pub Wiley).

19 H. Strack, K. R. Mayer and B. O. Kolbesen (1979).
The influence of stacking faults on MOS memories.
Solid State Electronics, 22, pp 134-140.

20 S. Rosenberg, D. Crook and B. Euzent (1979).
HMOS reliability.
IEEE Trans, TED-26, 1, pp 48-51.

21 K. Yamabe, K. Taniguchi and Y. Matsushita (1984).
Thickness dependence of dielectric breakdown failure of thermal $SiO_2$ films.
Proc. 22nd Int. Rel. Physics Symp, pp 184-190.

22 S. W. Pang (1984).
Dry etching induced damage in Si and GaAs.
Solid State Technology, April 1984, pp 249-256.

23 F. Fantini (1984).
Reliability problems with VLSI.
Microelectronics and Reliability, 24, pp 275-296.

24 G. Ottaviani and J. W. Mayer (1980).
Mechanisms and interfacial layers in silicide formation.
In 'Reliability and Degradation', op-cit, pp 105-149.

25 J. J. Gajda, G.J. Lindstrom and D. J. DeLorenzo (1981).
Interlevel insulation reliability evaluation.
IEEE Trans, TCHMT-4, pp 509-514.

26 T. C. May and M. H. Woods (1979).
Alpha-particle-induced soft errors in dynamic memories.
IEEE Trans, TED-26, pp 2-9.

27 V. V. Nickel (1980).
VLSI - The Inadequacy of the stuck-at fault model.
Proc. 1980 IEEE Test Conference, pp 378-381.

28 P. K. Chatterjee, P. Yang and H. Shichijo (1983).
Modelling of small MOS devices and device limits. IEE Proc., 130, pt. I, 3, pp 105-125.

29 P. E. Cottrell, R. R. Troutman and T. H. Ning (1979).
Hot electron emission in n-channel IGFET's.
IEEE JSC-14, pp 442-455.

30 Y. El-Mansy (1982).
MOS device and technology constraints in VLSI.
IEEE Trans, TED-29, 4, pp 197-203.

31 H. Masuda, M. Nakai and M. Kubo (1979).
Characteristics and limitation of scaled-down MOSFET's due to two-dimensional field effect.
IEEE TED-26, 6, pp 980-986.

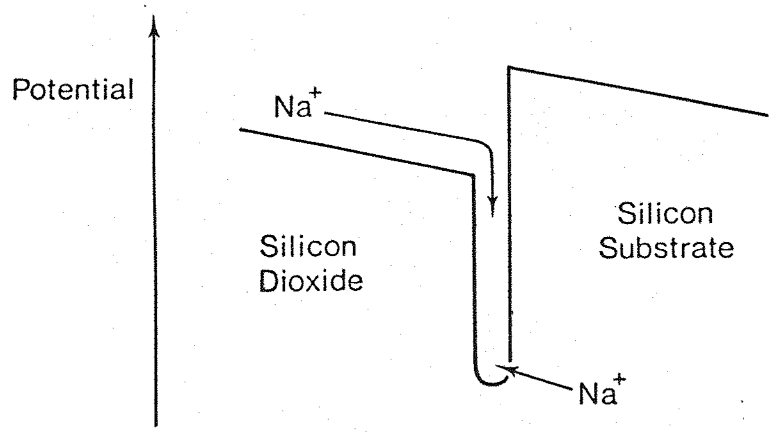


Figure 2.1 Ion accumulation at the silicon dioxide - substrate interface

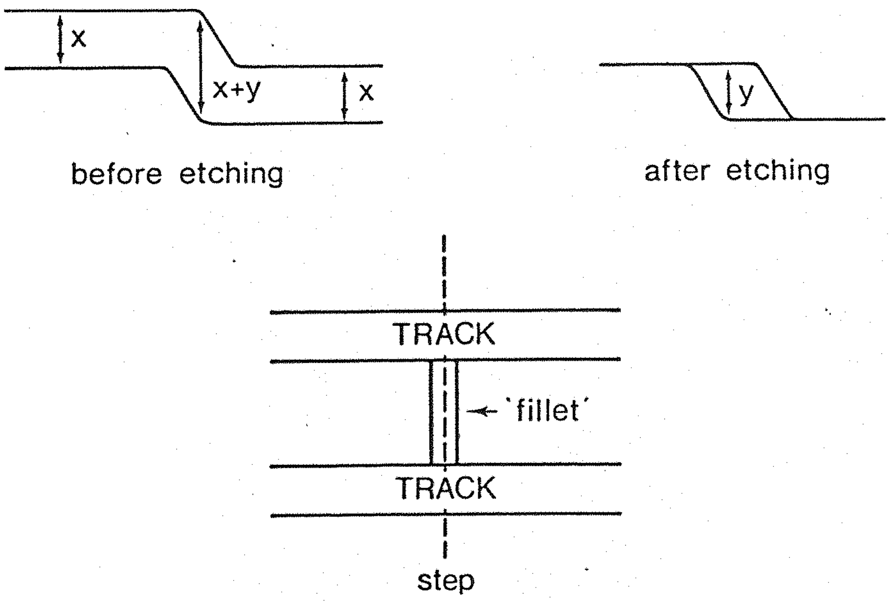


Figure 2.2 Production of fillets by anisotropic etching

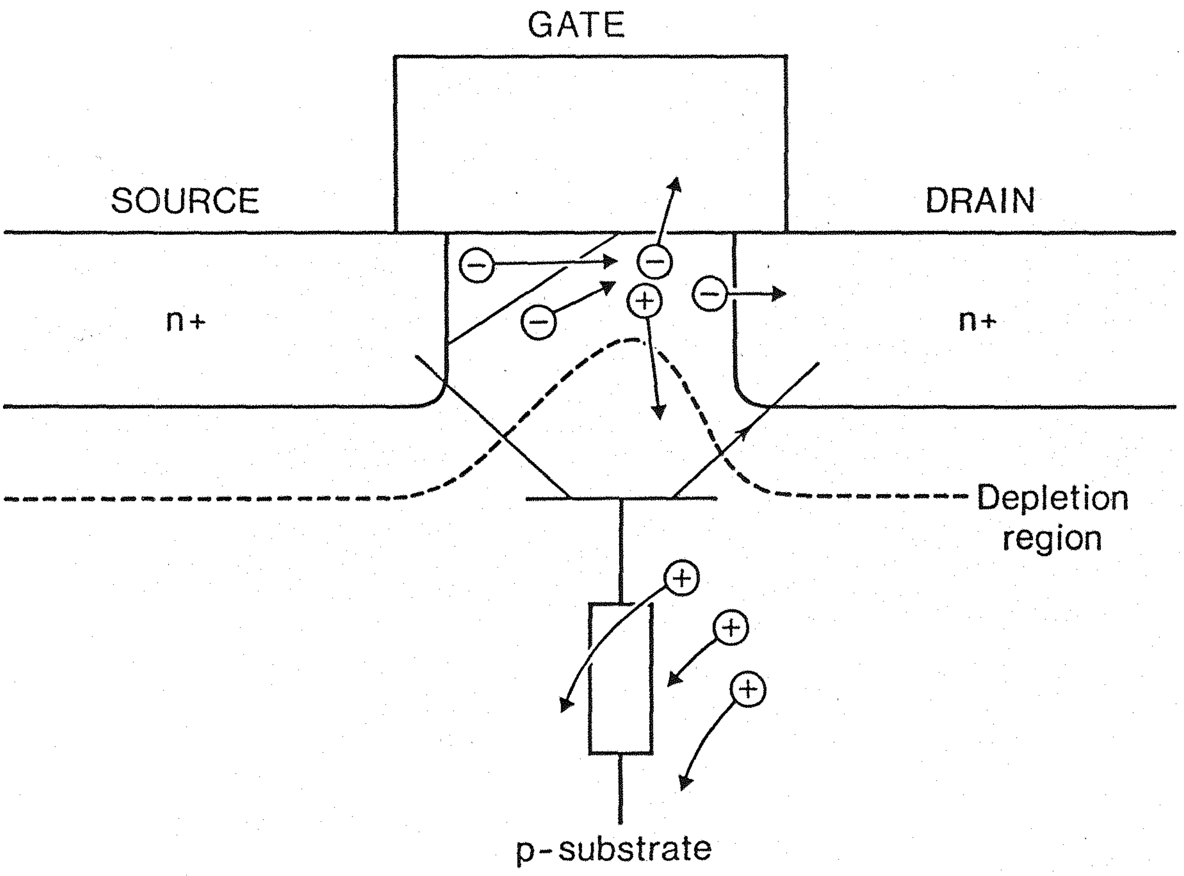


Figure 2.3 Short channel effects in an N-channel MOSFET

# CHAPTER 3 - INVESTIGATION OF FAULT-EFFECTS IN NMOS CIRCUITS

## 3.1 Introduction

The previous chapter reviewed the various ways in which MOS integrated circuits can fail. The effects of these failures on the performance of an MOS logic circuit need to be investigated, and evaluated in digital terms, because fault models operate at the digital, rather than electrical, level. The investigation will be conducted in two ways:-

1 by inserting the faults into a simple logic structure on a software simulator;

2 by studying failed test circuits to examine the divergence from fault-free behaviour of a circuit containing a fault. The fault itself should prove to be readily identifiable in such circuits, providing the opportunity to study the relationship between a fault and its associated fault-effects.

These approaches were chosen, rather than attempting to insert faults directly onto an LSI circuit, because of the difficulty of introducing faults into the circuit in a controlled manner during processing. In addition, a circuit simulator more accurately models MOS integrated circuit behaviour than an equivalent circuit made up of discrete components could, because of the enormous differences between the device characteristics (transconductance, substrate currents etc) of discrete and integrated components. This chapter describes the analysis of fault-effects in NMOS logic circuits, starting with the simulator-based work.

## 3.2 Fault insertion using SPICE

The fault-effects were evaluated by carrying out simulations on a string of inverters using a set of parameters describing a 3 μm NMOS process.

The SPICE circuit-level simulator (ref. 1) was chosen to perform the simulations - even though it was not originally designed to perform simulations of faulty devices - for three main reasons:-

- functional-level and logic-level simulators do not have the facilities necessary to specify MOS faults (for example, shifted threshold voltage) at the appropriate level;

- a set of SPICE parameters describing a 3 μm NMOS process was readily available from other work at British Telecom Research Laboratories;

- SPICE is widely used throughout industry, and is nowadays a recognised standard.

The faults were inserted singly into a chain of six inverters, and dc and transient analyses were subsequently performed on the chain. The chain length was set at six for two reasons:

- Using SPICE's transient analysis facility, it proved to be very difficult to specify accurately an input waveform to the faulty inverter that exactly matched a fault-free inverter's output waveform. In the event, a step input (with zero rise-time) was applied to the chain and the fault was inserted into the first inverter in the chain whose fault-free output waveforms exactly matched its predecessor's output waveforms. This turned out to be the fourth inverter in the chain.

- Two inverters followed the faulty inverter in the chain so that the impact of the fault on a following inverter that was itself driving an inverter could be evaluated if necessary.

The faults were introduced either by adjusting one of the SPICE MOSFET model parameters from its fault-free value, or by adding an extra component (ref 2). Table 1 shows a list of faults that covers those reviewed in the previous chapter together with the means by which each was inserted into an inverter.

## TABLE 1 - FAULT INSERTION USING SPICE

| | Faults | Means of Insertion |
|---|---|---|
| 1. | Open and short circuits (more correctly, abnormally high and low ohmic resistances respectively) in the layers of interconnect and excessively high resistance contact windows. | - resistor connected onto MOSFET terminals (high values of resistance in series with devices were used to model open circuits). |
| 2. | Excessive drain/source-substrate p-n junction reverse leakage current. | - SPICE parameter JS |
| 3. | Large threshold-voltage shift | - SPICE parameter VTO |
| 4. | Transconductance degradation | - SPICE parameter KP |
| 5. | Gate-oxide breakdown | - resistor connected between MOSFET terminals |

| cont'd | Faults | Means of Insertion |
|---|---|---|
| 6. | Excessive subthreshold leakage current (punchthrough) | - SPICE parameter ETA |
| 7. | Width and length variation | - SPICE parameters W and L. |

Care was taken in choosing the SPICE parameters to be adjusted, as some parameters are used to model effects in the MOSFET, other than those described in the SPICE User Manual. Also, if certain parameter values are user-specified, other non-user-specified parameters values may be re-calculated by the software and used to over-write expected default values. To check these points, fault 3 was simulated a second time by placing a voltage source on the MOSFET's gate terminal - no appreciable differences between the two sets of simulations were found. Fault 2 was also simulated a second time by connecting diodes between the substrate terminal and the source and drain terminals. Differences were noticed in this case because SPICE models the reverse leakage current as a linear dependence on the applied voltage, rather than use the classical diode equation. Since the former equation, rather than the latter, more closely models results obtained from experiments, the parameter JS was used to simulate this particular fault (ref 3).

The gate propagation delay and the high and low noise margins were used to investigate the fault-effects. The voltage at which the inverter's input and output voltages were identical for a normal device was defined as $V_{INV}$ (figure3·1). The propagation delay was then defined as the time between the 'normal' $V_{INV}$ being applied to the faulty inverter's input and $V_{INV}$ appearing at the output.

The two points on the dc transfer characteristic where the slope equalled -1 defined the parameters $V_{OH}$, $V_{OL}$, $V_{IH}$ and $V_{IL}$ which, in turn, were used to define the high and low noise margins (figure 3.1). However, the usual definitions of noise margins ($V_{OH}$-$V_{IH}$ and $V_{IL}$- $V_{OL}$) are not adequate for faulty logic circuit descriptions since these definitions implicitly assume that the driving logic gate and the driven logic gate have identical characteristics. In a real circuit, where a faulty gate would both drive and be driven by fault-free gates, the noise margins would be the difference between the faulty gate's output (input) voltage levels and the fault free gate's input (output) voltage levels alone, as follows (see figure 3.2)

$(V_{OH_f} - V_{IH_n})$ Output high noise margin

$(V_{ILn} - V_{OLf})$ Output low noise margin

$(V_{OHn} - V_{IHf})$ Input high noise margin

$(V_{ILf} - V_{OLn})$ Input low noise margin (subscript f means quantity measured on a faulty inverter; subscript n means quantity measured on a normal inverter).

Thus, $V_{OHf} - V_{IHn}$ is the amount of noise that may be tolerated on a faulty inverter's output high signal without causing a following inverter to operate in its region of high gain ($V_{OL} < V_{out} < V_{OH}$ - see figure 3.1) - in other words, without making the faulty inverter's output noise-sensitive. If this quantity is negative, a noise-sensitive node has been produced. If the signal has been driven through one or more pass transistors, a noise sensitive node is created if $V_{OHf} - V_{IHn} < V_{TH}$, the threshold voltage, because a logic high signal is lowered by one threshold voltage when propagated through an NMOS pass transistor.

$V_{OHn} - V_{IHf}$ is the amount of noise that may be tolerated on a fault-free inverter's output high signal without causing a following faulty inverter to operate in its region of high gain. This definition may also be re-defined to take account of threshold voltage drops introduced by pass transistors where relevant.

$V_{ILn} - V_{OLf}$ is the amount of noise tolerable on a faulty inverter's output low signal without causing a following fault-free inverter to operate in its region of high gain. $V_{ILf} - V_{OLn}$ is the amount of noise tolerable on a fault-free inverter's output low signal (no threshold drop) without causing a following faulty inverter to operate in its region of high gain.

These figures of merit (propagation delay and noise margins) were chosen because they represent concepts familiar to digital design engineers, and thus a 'feel' for the fault-effects may be readily obtained. However, these definitions of the noise margins in terms of the slope values, and the propagation delay in terms of the 'normal' $V_{INV}$ are rather arbitrary. This means that, in the final analysis, the precise values of the 'figures of merit' are less important than the gross changes in inverter behaviour, reflecting the fault-effects.

Figure 3.3 shows the open and short circuits that were inserted, and figures 3.4 and 3.5 show the results of the simulations. The particular values of resistance, threshold voltage, leakage current etc chosen for the simulations are not significant in themselves, other than that they cover a wide range of abnormal behaviour, often up to the occurrence of a stuck-at fault-effect as shown on the faulty inverter transfer characteristics (fig 3.4).

The simulation results show that, in the presence of a fault, the performance of an NMOS inverter is degraded in one or both of two ways: propagation delays are, in most cases, increased, and the output voltage levels are shifted, resulting in decreased noise margins. The magnitude of these effects depends, as expected, upon the severity of the fault. In extreme cases, the fault-effects tend to manifest themselves as "stuck nodes". For example, the drain contact window failure (a, fig 3.3) displays both voltage shifts (fig 3.4 (a)) and dramatically increased propagation delays (fig 4(c)) whereas the reverse diode leakage failure results in a much lowered $V_{0H}$ (fig 3.4(b)) and little alteration in the propagation delays (fig 3.5 (h)). Conversely, the buried contact failure ((e), fig 3.3) produces no change in the dc characteristics of the inverter, but increases the 0-to-1 propagation delay enormously (fig 3.5 (a)).

Enhancement transistor gate-to-channel region shorts were not simulated because the SPICE MOSFET model does not have a channel node - the channel current is modelled by a current source connected between the source and drain nodes. Intuitively, however, this fault would be expected to cause the inverter's input to be "stuck-at-0" (ref 4) (see also fig 3.4(c)).

Enhancement transistor gate-to-source shorts ((g), fig 3.3) have, for the purposes of presentation of results, been "redefined" as shorts from the output of the driving inverter to ground, and the results shown are those for the driving inverter rather than the faulty inverter (fig 3.4(c)).

The negative propagation delay shown for large values of ETA (fig 3.5 (g)) arises as a result of the definition of the delay as the time between $V_{INV}$ being applied to the input and $V_{INV}$ appearing at the output. The dc transfer characteristics for devices with high source-drain leakage current are such that $V_{INV}$ appears at the output when the input voltage is less than 0.5V. Thus, when the input voltage rises from $V_{ILn}$, the output voltage has fallen through $V_{INV}$ from $V_{OHf}$, before the input voltage reaches $V_{INV}$ - and this by definition is a negative delay.

Overall, the results show that stuck-node fault-effects represent a subset of all possible fault-effects, examples of other effects being serious timing delays and reduced noise margins, which may give rise to intermittent, "soft" errors in an LSI device.

## 3.3 Analysis of faulty NMOS circuits

The question remains as to whether fault-effects predicted by the simulations are physically realistic and representative of failures in typical NMOS LSI circuits. There are two main reasons for this uncertainty:-

* the fault insertion was done on SPICE, a circuit simulator, rather than on real circuits. No simulator can represent circuit behaviour completely accurately on account of the simplifications used in device modelling and in

the internal mathematics of the simulator. Furthermore, SPICE is conventionally used to simulate devices under normal conditions, and thus its simulations of faulty devices might not be as accurate as those of fault-free devices.

- only faults within single devices have been simulated. The string of inverters does not provide opportunities to study bridging faults between signal-carrying inter-connections.

To examine these issues, the simulator-based work was argumented by examining faulty NMOS circuits, using the knowledge gained from the simulations to guide the investigation. Two circuits were examined: British Telecom's BT 381 Quad 256-bit shift Register, and test chips ("drop-ins") from Southampton University's own silicon processing facility.

Drop-ins are included on every wafer processed at the facility to monitor the processing. The NMOS drop-ins contain contact chains, Van der Pauw sheet resistance tests, individual transistors, isolation and continuity tests for the different conducting layers, and minimum geometry inverters.

Some of the faulty inverter transfer characteristics obtained from the drop-ins are shown in figure 3.6. Results from the SPICE simulations that closely match the drop-ins' characteristics have been superimposed on the same diagrams. It is, of course, impossible to determine from the drop-in transfer characteristics the faults present in the inverters. However, subsequent investigation of the drop-in circuits using a transistor curve tracer confirms that the same fault mechanism is indeed present in both. Thus, the simulations presented above can be considered realistic, and non-stuck faults are confirmed as forming a significant subset of all failures.

15-stage ring oscillators are also included on the drop-ins, and their free-running rate is typically 15 MHz. However some oscillators run more slowly as a result of non-catastrophic changes in the processing (variations in ion implant dose and energy, gate oxide thickness etc) or, as the SPICE simulations showed, by fault conditions in an individual transistor.

Having thus confirmed the simulation results as being realistic, there was now a need to assess the open and short circuit faults that may be found on a chip. To do this, forty faulty shift register chips were examined for optical defects resulting from photolithography errors, using a scanning electron microscope (SEM) with voltage contrast and a high-powered optical microscope. Under an SEM with voltage contrast, interconnections show up as light or dark regions, depending on their electrical potential (0 V = light, + 5 V = dark), against a grey background. The SEM thus facilities the rapid location of a faulty cell in a shift register, which would then be examined under the optical microscope for photolithographic faults.

The BT 381 Quad Shift Register (QSR) chip consists of four 256-bit quasi-static shift registers, two of which are expandable to 257 bits, and all of which may be cleared to contain all 0's. Figure 3.7(a) shows the layout and gate/transistor equivalent circuit of a single cell. $\phi_1$ and $\phi_2$ are non-overlapping clocks, with $\phi_3$ 'following' $\phi_2$ by 10 ns, thus permitting all three clock signals to be generated from a single clock input pin, 'CK'. The clock frequency may be anything between 0 and 2 MHz, although CK should not be low for more than 5 ms at a chip temperature of 125°C . When CK goes low, the data is shifted: $\phi_2$ goes low (closely followed by $\phi_3$), and then $\phi_1$ goes high, the data being dynamically stored at point X. The data is latched on CK going high: $\phi_1$ goes low, and then $\phi_2$ , followed by $\phi_3$, goes high, thus statically storing the data as a result of the feedback loop through $\phi_3$ .

Forty faulty QSR chips were examined, of which six had pin or bonding faults, twelve had faults that were not visible, and the rest had visible photolithography-related faults. A cross-section of the visible faults is presented in figure 3.7(b)-(m) with the fault-effects being inserted on the circuit diagram below the layout schematic. For the most part, the faults consist of wrongly-etched shapes or wrongly-defined features, and result in "stuck-at" faults, whereby a node on the circuit diagram is shorted to one of the power rails. Examples of non-stuck faults are shown in figures 3.7(e) and 3.7(f) where a stuck-on transistor results in data being latched through two successive QSR stages in one clock period. Similarly, figure 3.7(g) shows a fault that, if less severe, would produce an abnormally low-gain load transistor, which, from the SPICE simulations, we know would cause timing problems and logic level degradations. Figure 3.7(j) depicts a problem which produces an output stuck-at fault since whenever $\phi_1$ is high, $\phi_2$ holds the output of the cell low, as $\phi_1$ and $\phi_2$ are non-overlapping clocks. Finally, figure 3.7(m) shows a similar fault-effect for a different fault: when $\phi_2$ is high (and $\phi_1$ low) the input to the first inverter is held low and thus the cell output is held low. On the following clock phase ($\phi_1$ high, $\phi_2$ low) the cell output is still low by charge storage at the input to the second inverter, and thus the cell output is effectively s-a-0.

Overall, these results show that photolithography-related defects can introduce open and short circuits that may produce stuck nodes in a more complex "random logic" circuit, as well as "hard-to-detect" non-stuck node faults (interconnect open and short circuits, including parameter degradation).

### 3.4 Revised SPICE simulations of faulty contact windows

Measurements made on the contact window chains on the drop-in chips and other work done at BTRL show that contact windows fail in three distinct ways:

- * high resistance (up to 20 k-ohms):
- * open circuit (displaying capacitance of up to 0.1 pF);
- * rectifying behaviour (also with capacitance effects).

These results show that modelling a contact window failure simply as a resistance (up to 10 M-ohms) is not physically realistic. The last two effects are manifestations of the same fault: a thin film of material left in the contact after etching, thus preventing a direct contact between metal and silicon. If the film is less than 100 Angstroms thick, tunnelling currents are significantly reduced in one direction and the contact displays rectifying behaviour as a result; otherwise, the contact may be adequately modelled as a capacitance. (Polysilicon - silicon contacts will not form rectifying contacts because both materials are n-type silicon). SPICE simulations were performed using the more appropriate components to investigate realistically the contact window failures.

Restricting the resistance value to 20 k-ohms produces no significant fault effects in the drain contact ($R_D$) and the buried contact ($R_B$), but the stuck node effect in the source contact window ($R_S$) is maintained (figure 3.4). The severe timing degradations introduced by the higher values of $R_D$ and $R_B$ ($\geqslant$ 1 M-ohm) may be discarded as being unrealistic (figures 3.5(a) and 3.5(c)).

The purely capacitative contacts consistently produce stuck output nodes, which is to be expected for the source and drain contacts, where one side of the capacitor is held at a fixed voltage. However, the stuck-at-0 output predicted by SPICE for a buried contact that has become purely capacitive as the result of a fault appears to be in sharp contradiction with an effect observed under be the SEM. There, an open circuit contact (capacitor) behaves in a fault-free fashion until observed using the SEM whereupon the electron beam forces one plate of the contact to a fixed voltage, preventing propagation of a signal through it. This anomaly is due to SPICE not modelling the charge present in a piece of interconnect connecting two capacitors in series (the two capacitors here being the faulty contact and the gate of following MOSFET).

A normal, 'ohmic' contact is a Schottky barrier diode (SBD) whose depletion region is so thin that its reverse leakage current, $I_S$(actually majority carrier leakage) is several orders of magnitude greater than it would be in a properly-formed SBD. This is because electrons can readily tunnel through the potential barrier formed by the very thin depletion region, and thus a rectifying contact properly-formed SBD may be formed if the potential barrier exceeds a critical height. This can occur if a thin (< 100 Angstroms) oxide layer remains on the silicon before the aluminium is deposited, or if surface states, due to incomplete covalent bonds and other effects, exist in larger than normal quantities. These surface states are analogous to the surface states found between the channel and gate oxide of a MOSFET, and have an associated fixed charge that affects the width of the depletion layer in the silicon. Experimental evidence of this effect (taken from ref 5) is shown in figure 3.8, and similar results have been obtained at BTRL.

Since tunnelling currents are inhibited, the reverse leakage current in the SBD is restricted to thermal electrons with sufficient energy to traverse the barrier and is described by the equation

$$I_S = BT^2 \ \exp \ (-q\phi/kT)$$

where B is a constant and $\phi$ the barrier height. This restriction on the current has much the same effect as putting a high value resistance in the contact, and this is confirmed by SPICE. $I_S$ for a 100 $\mu m^2$ aluminium - n-type silicon SBD is typically 2 x $10^{-11}$ A (ref 6). By increasing this to 5 x $10^{-7}$ A for a 25 $\mu m^2$ contact window, a normal normal inverter dc characteristic is obtained (figure 3.9(a)). However, if $I_S$ = 5 x $10^{-8}$A for the $V_{SS}$ contact, a characteristic looking very similar to that for a source contact resistance $R_S$ = 20 k ohms is obtained (figure 3.8(b)). Decreasing $I_S$ to 0 produces an output stuck fault as the contact is now purely capacitative. (The SPICE diode model consists of a voltage-controlled current source in parallel with a voltage-dependent capacitance).

The transient analyses show that the introduction of SBD's slows the inverter down since there is less current available to charge an output node, and a higher resistance path to ground along which the output node may discharge. As seen in section 3.1, this can also be produced by inserting a high resistance in the contact, but the SBD's appear to introduce a smaller delay than the resistance, as shown in Table 2 below.

Diode in Source contact

| I (A) | $T_r$ (ns) | $T_f$ (ns) |
|---|---|---|
| $5 \times 10^{-7}$ | 1.74 | 1.75 |
| $1 \times 10^{-7}$ | 1.68 | 1.80 |
| $8 \times 10^{-8}$ | 1.64 | 1.84 |
| $6 \times 10^{-8}$ | 0.99 | 1.90 |

Resistor in Source contact

| R (ohms) | $T_r$ (ns) | $T_f$ (ns) |
|---|---|---|
| 30 | 1.76 | 1.74 |
| 1K | 1.74 | 1.80 |
| 10K | 1.63 | 2.85 |

## REFERENCES:

1. A Vladimirescu and S Liu (1980)
   The simulation of MOS integrated circuits using SPICE2.
   ERL Memo, University of California, Berkeley, Feb 1980.

2. P Banerjee and J A Abraham (1984). Op. cit.

3. N E Lycoudes and C C Childers (1980). Op. cit.

4. M W Sievers and A Avizienis (1981).
   Analysis of a class of totally self-checking functions implemented in an MOS LSI general logic structure.
   Proc 11th Int Symp. Fault Tolerant Computing, pp 256-261

5. E Vieujot-Testamale et al (1983).
   Properties of the contact on ion cleaned n- and p-type silicon surfaces.
   Solid-state Electronics, 26, pp 325-331.

6. D Hodges and H Jackson (1983).
   Design and Analysis of Digital Integrated Circuits.
   pub McGraw-Hill

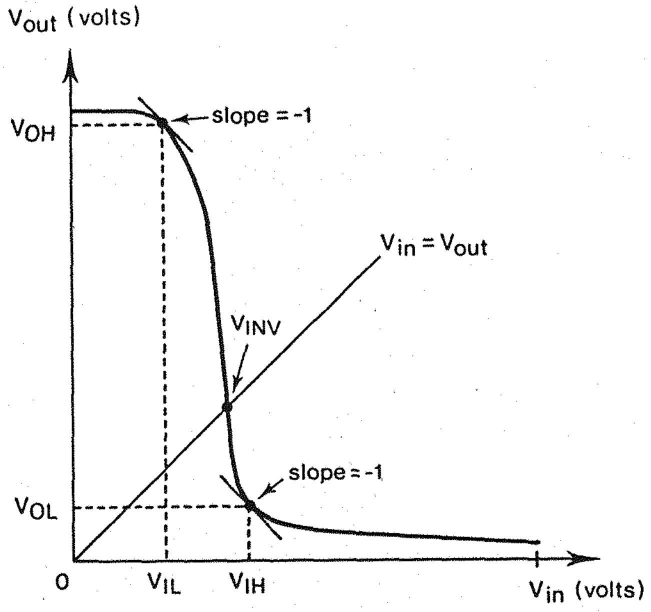


Figure 3.1 Derivation of figures of merit from a fault-free inverter's transfer characteristic

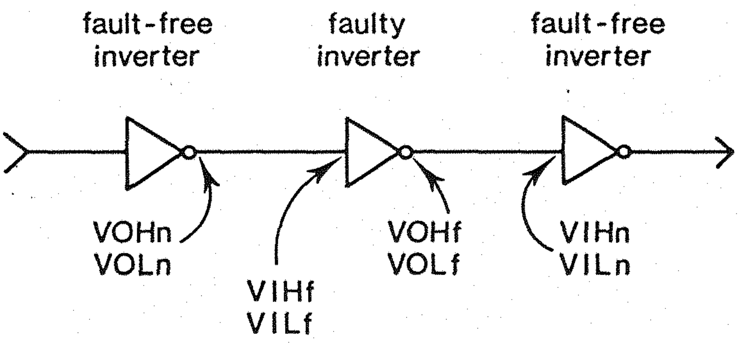
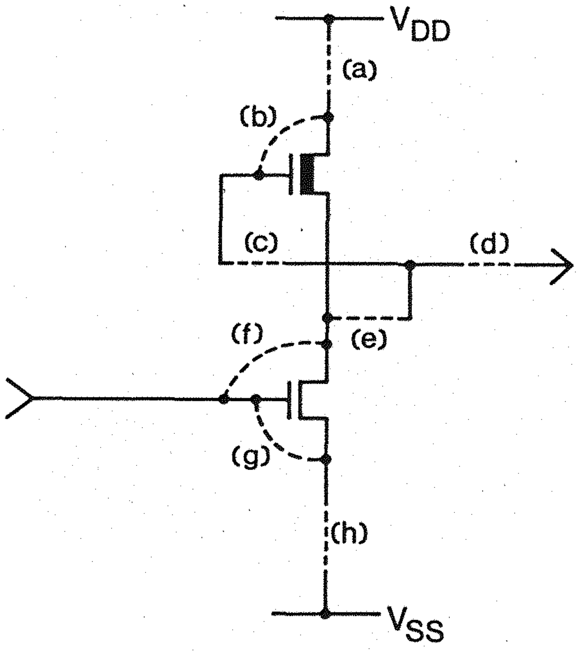
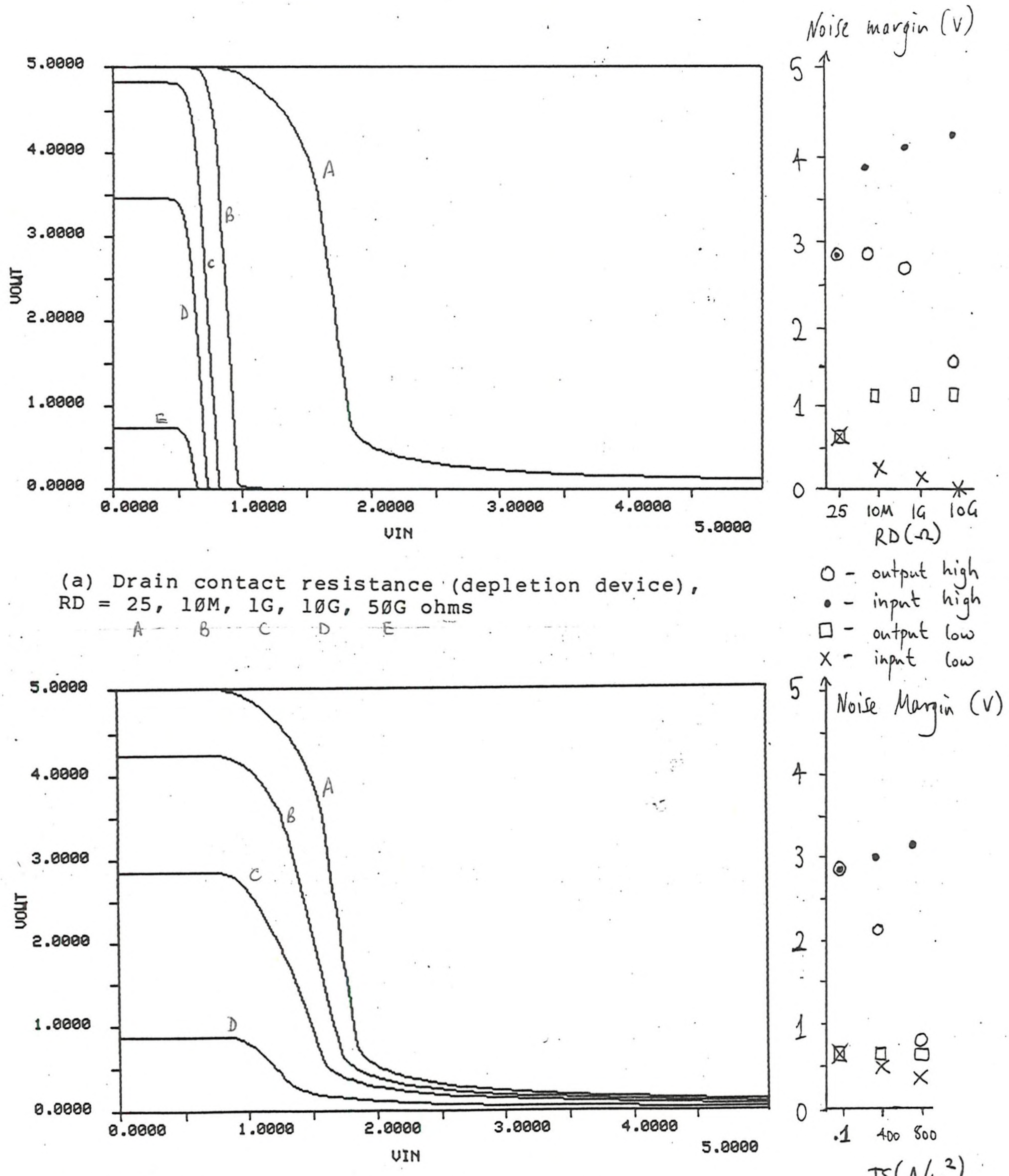


Figure 3.2 Chain of inverters showing parameter location



(a) Drain contact window open circuit, RD
(b) Depletion device gate-drain pinhole, RGDD
(c) Depletion device gate-source open circuit, RGSD
(d) Output line open circuit, RO
(e) Buried contact open circuit, RB
(f) Enhancement device gate-drain pinhole, RGDE
(g) Enhancement device gate-source pinhole, RGSE
(h) Source contact window open circuit, RS

Figure 3.3 Possible open and short circuits in an inverter



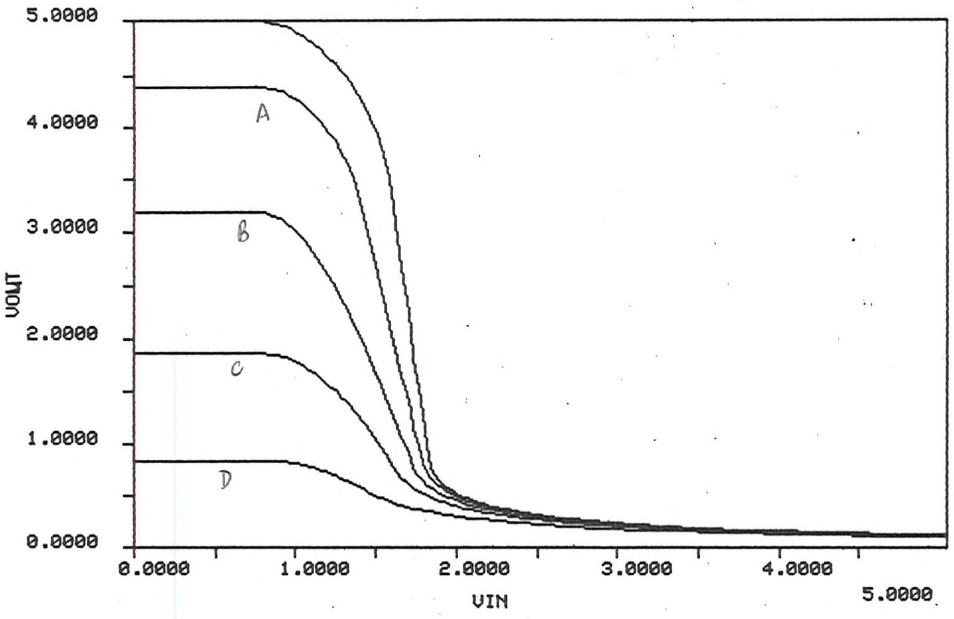
(a) Drain contact resistance (depletion device), RD = 25, 1ØM, 1G, 1ØG, 5ØG ohms
A B C D E

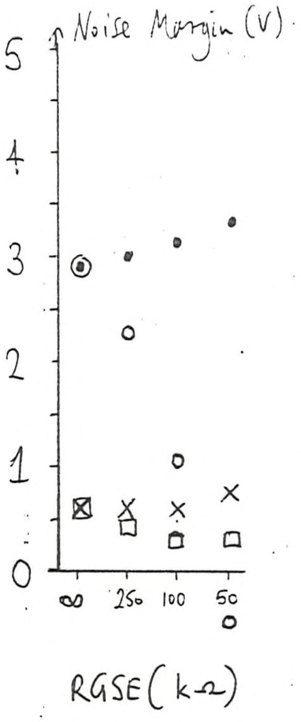
○ - output high
• - input high
□ - output low
× - input low

(b) Drain/source-substrate reverse diode leakage current, JS = .1, 4ØØ, 8ØØ, 16ØØ A/m**2
A B C D

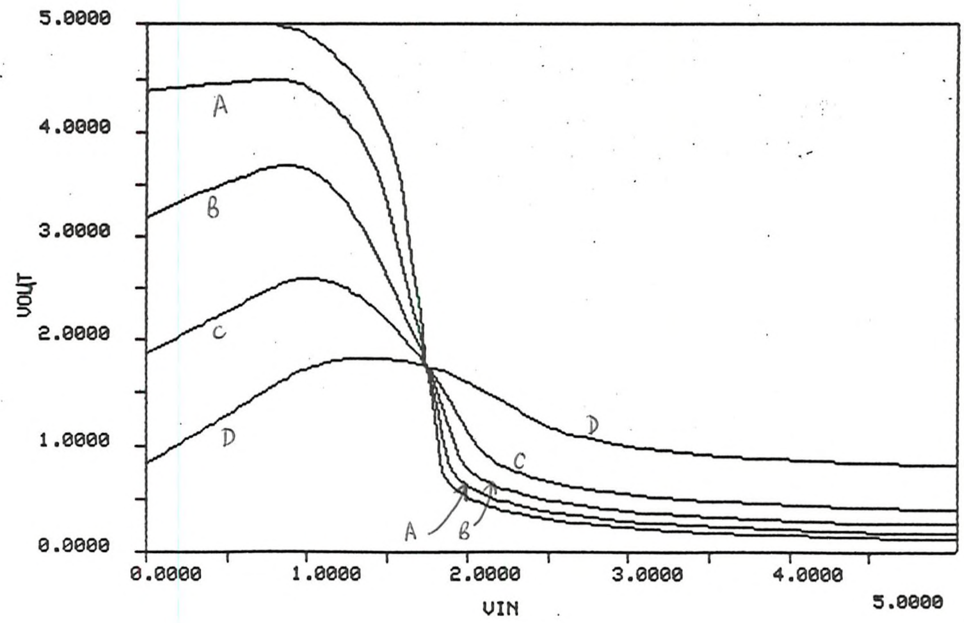
Figure 3.4 SPICE simulations of dc transfer characteristics of faulty inverters, with noise margins

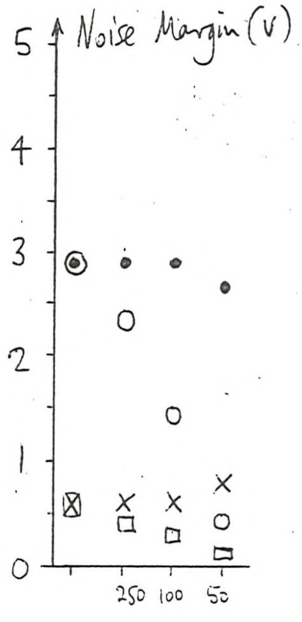
Note: faults which did not produce significant variations in the dc transfer characteristic shapes have been omitted (e.g. length and width variations of the depletion device)



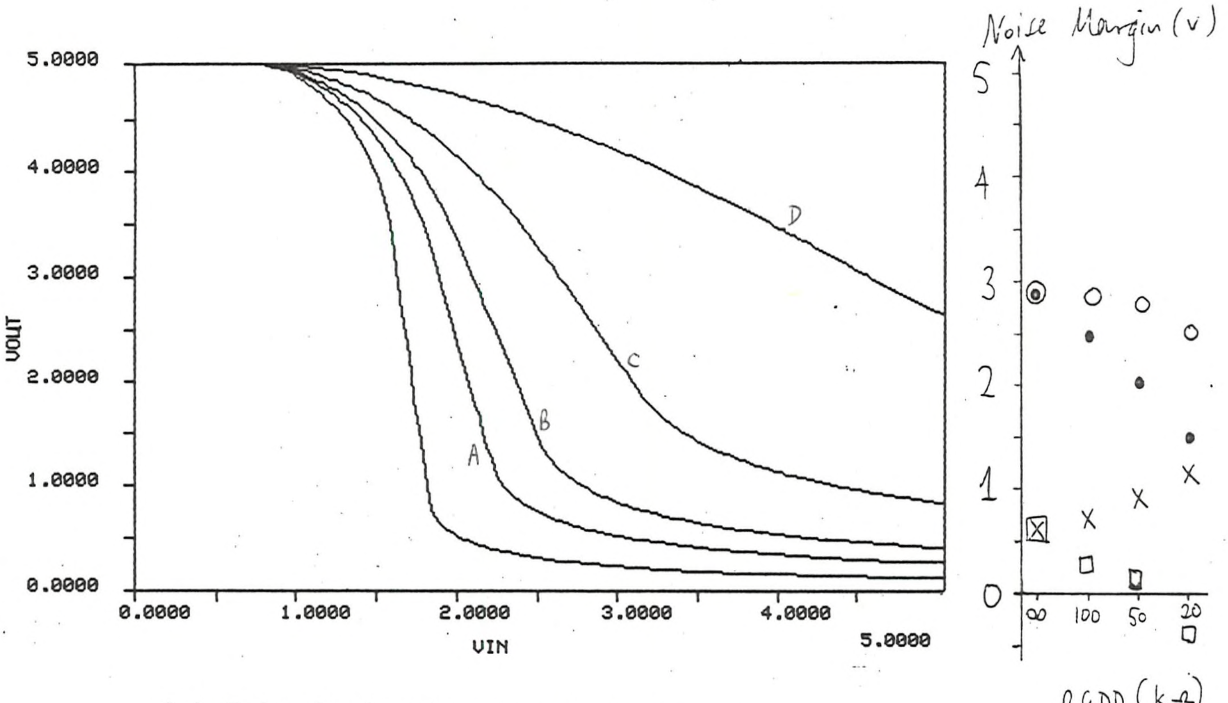


(c) Gate-source pinhole (enhancement device),
RGSE = 25Ø, 1ØØ, 5Ø, 2Ø k-Ohms
A B C D

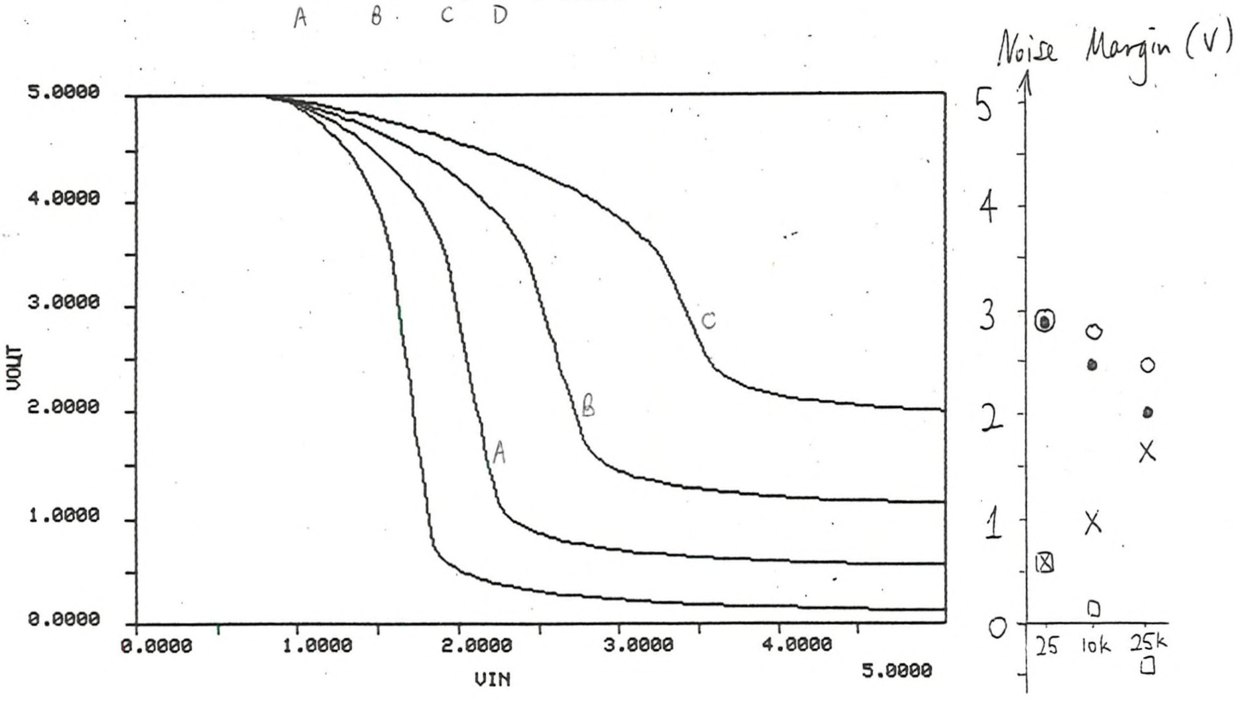




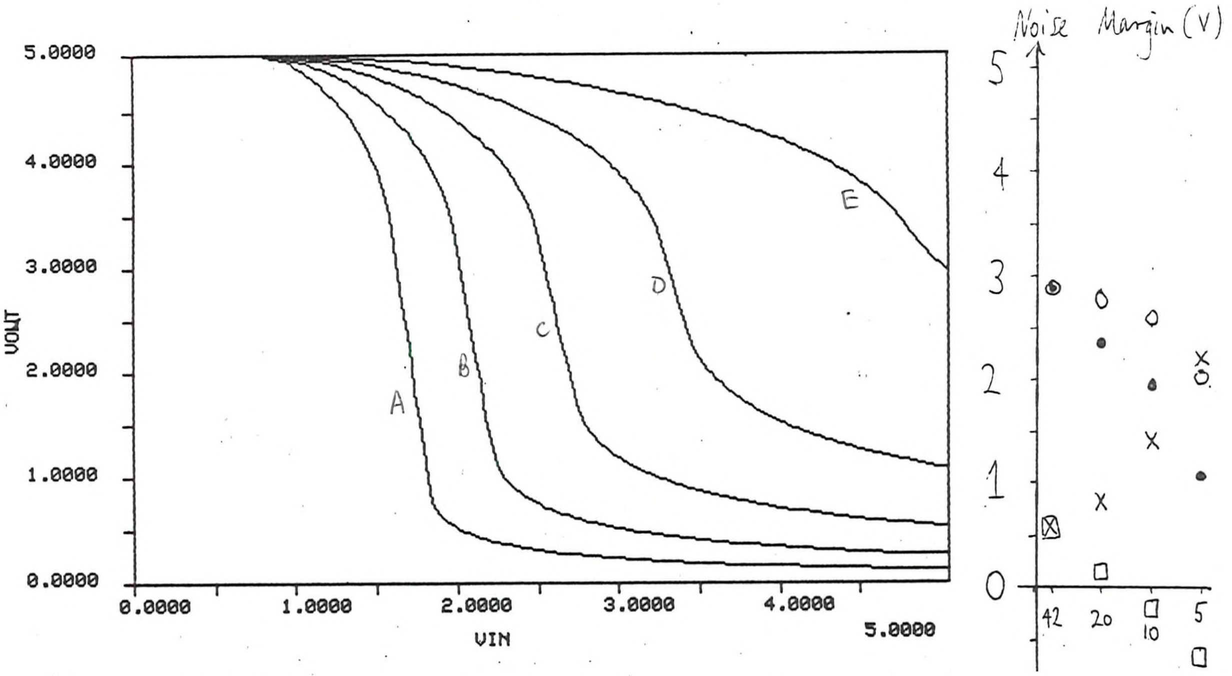
(d) Gate-drain pinhole (enhancement device),
RGDE = 25Ø, 1ØØ, 5Ø, 2Ø k-Ohms
A B C D



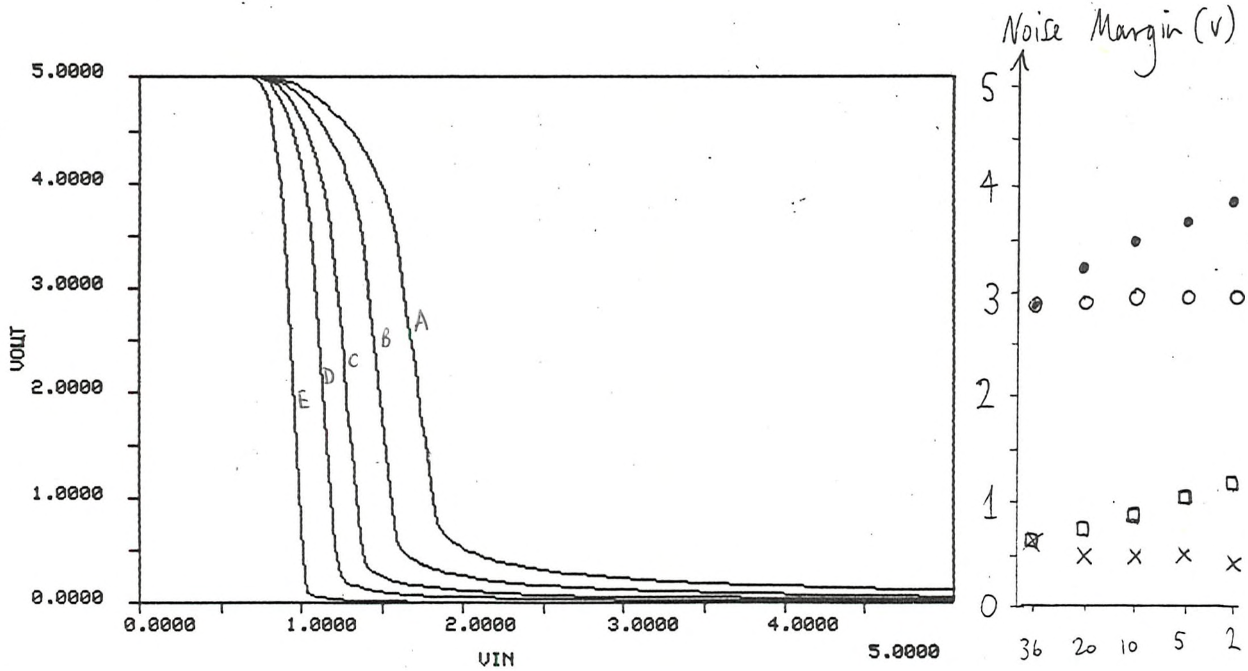
(e) Gate-drain pinhole (depletion device),
RGDD = 1ØØ, 5Ø, 2Ø, 5 k-Ohms
A B C D



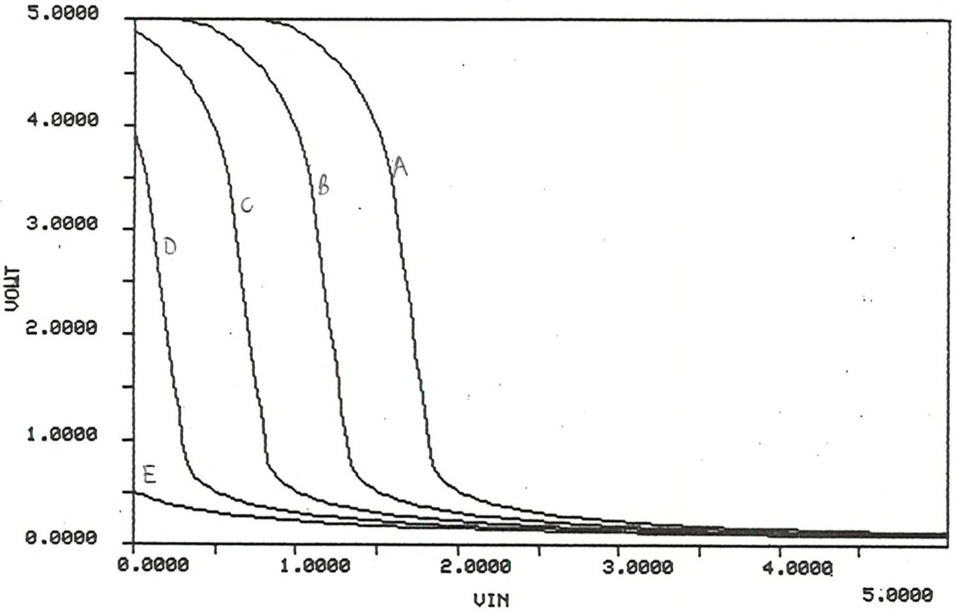
(f) Source contact resistance, RS = 5Ø, 25, 1Ø k-Ohms
A B C

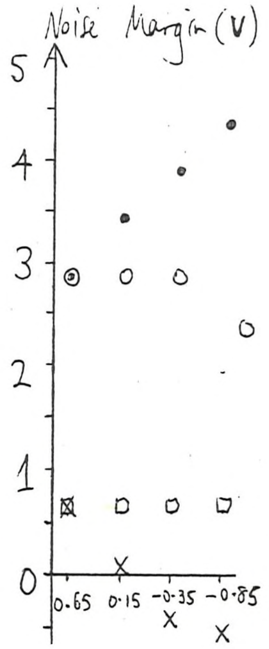


(g) Transconductance (enhancement device),
KPE = 42, 2Ø, 1Ø, 5, 2 uA/V
A B C D E



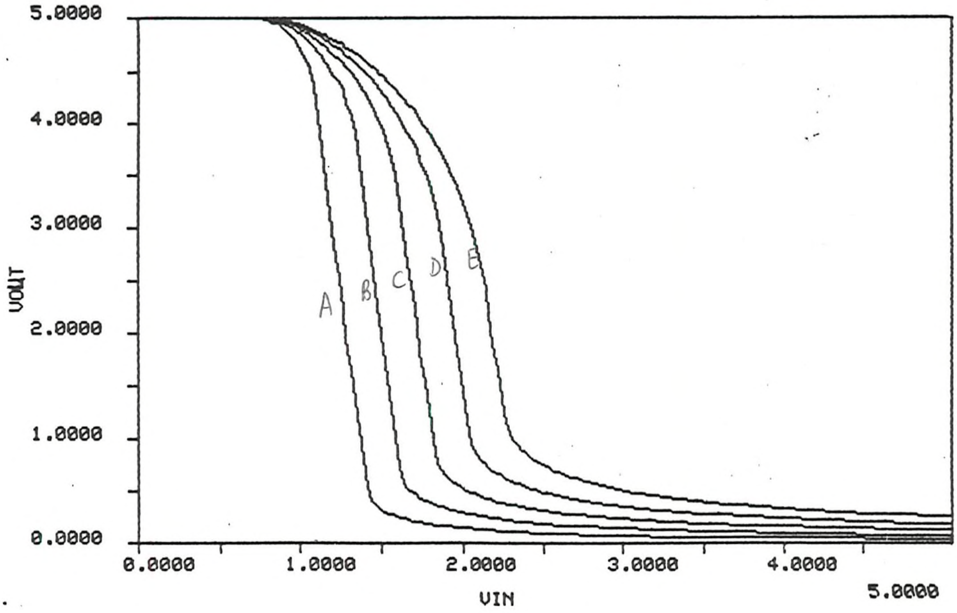
(h) Transconductance (depletion device),
KPD = 36, 2Ø, 1Ø, 5, 2 uA/V
A B C D E

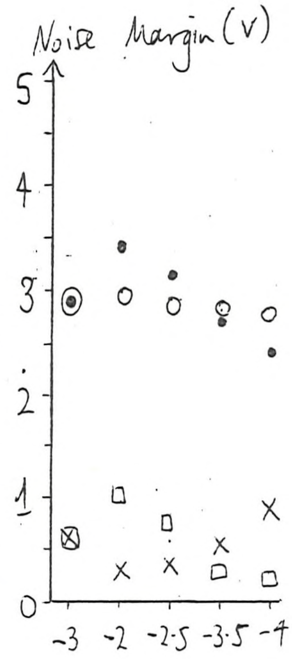




(i) Threshold voltage (enhancement device),
VTE = Ø.65, Ø.15, -Ø.35, -Ø.85, -1.35 V
A B C D E

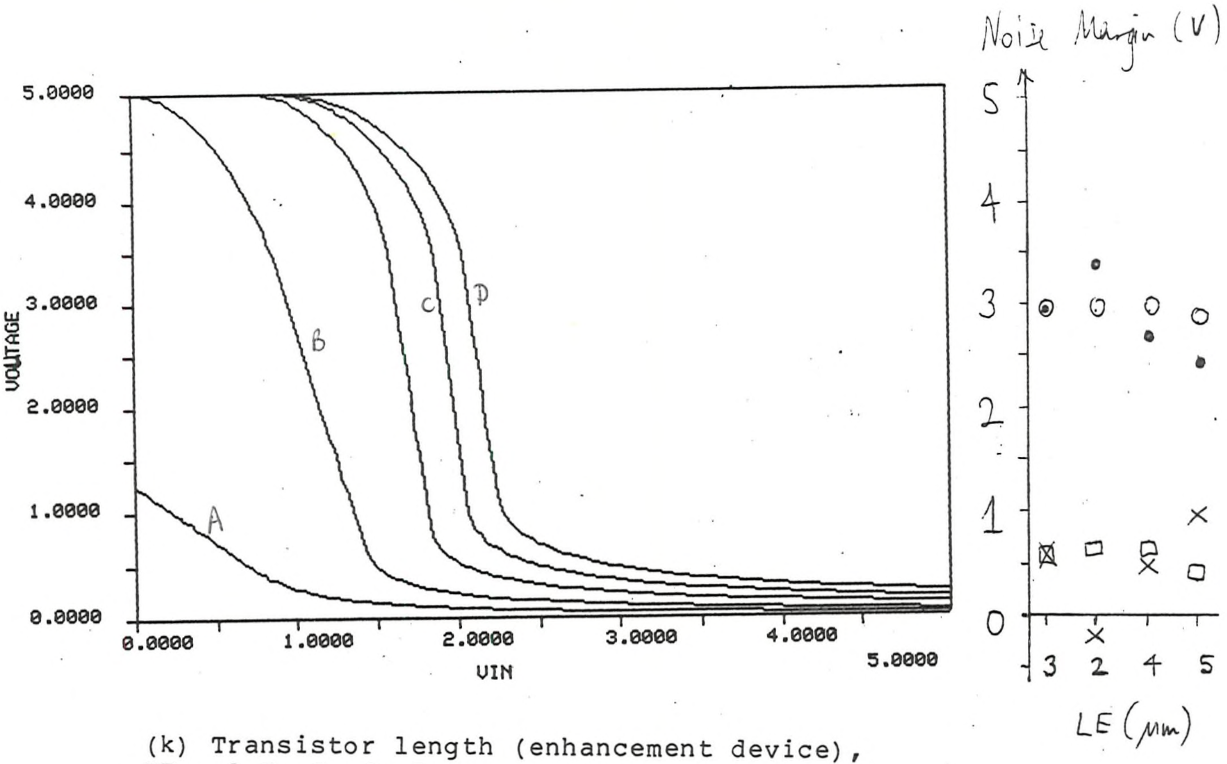
VTE (v)



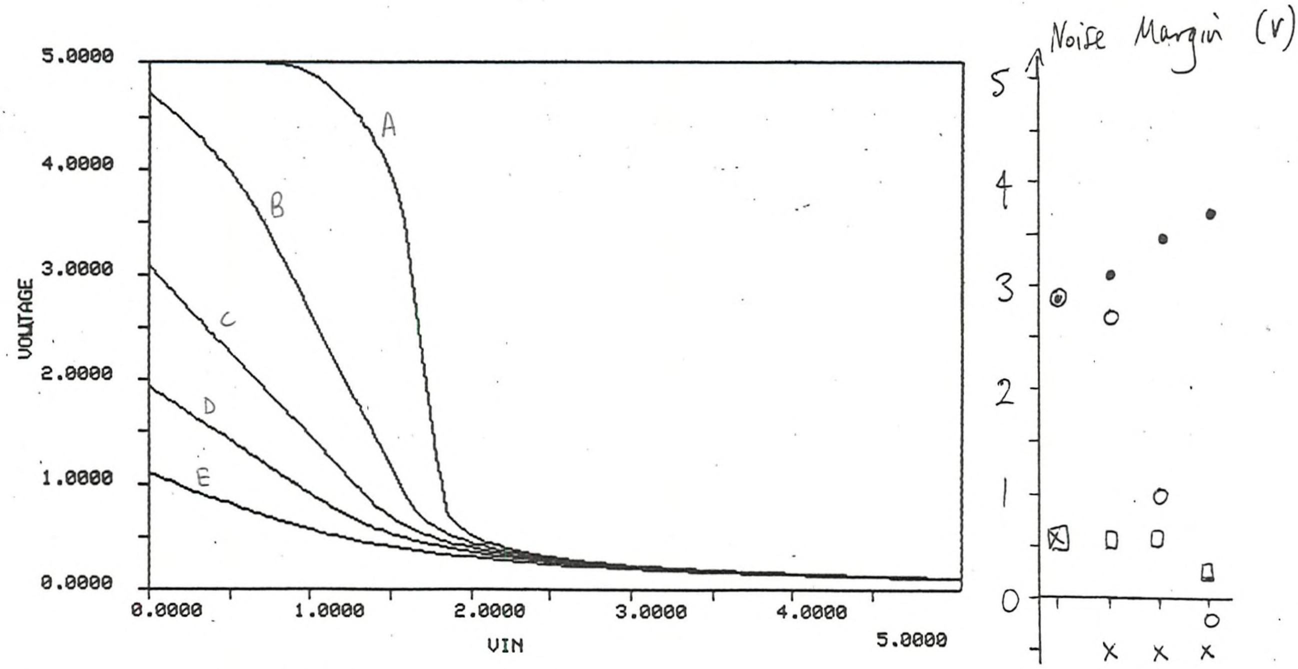


(j) Threshold voltage (depletion device),
VTD = -4, -3.5, -3, -2.5, -2 V
A B C D E

VTD (v)



(k) Transistor length (enhancement device), LE = 1.5, 2, 4, 5 um
A B C D



ETA

(l) Subthreshold leakage current factor, ETA = Ø.3, 3, 6, 1Ø, 18 (scalar quantity)
A B C D E

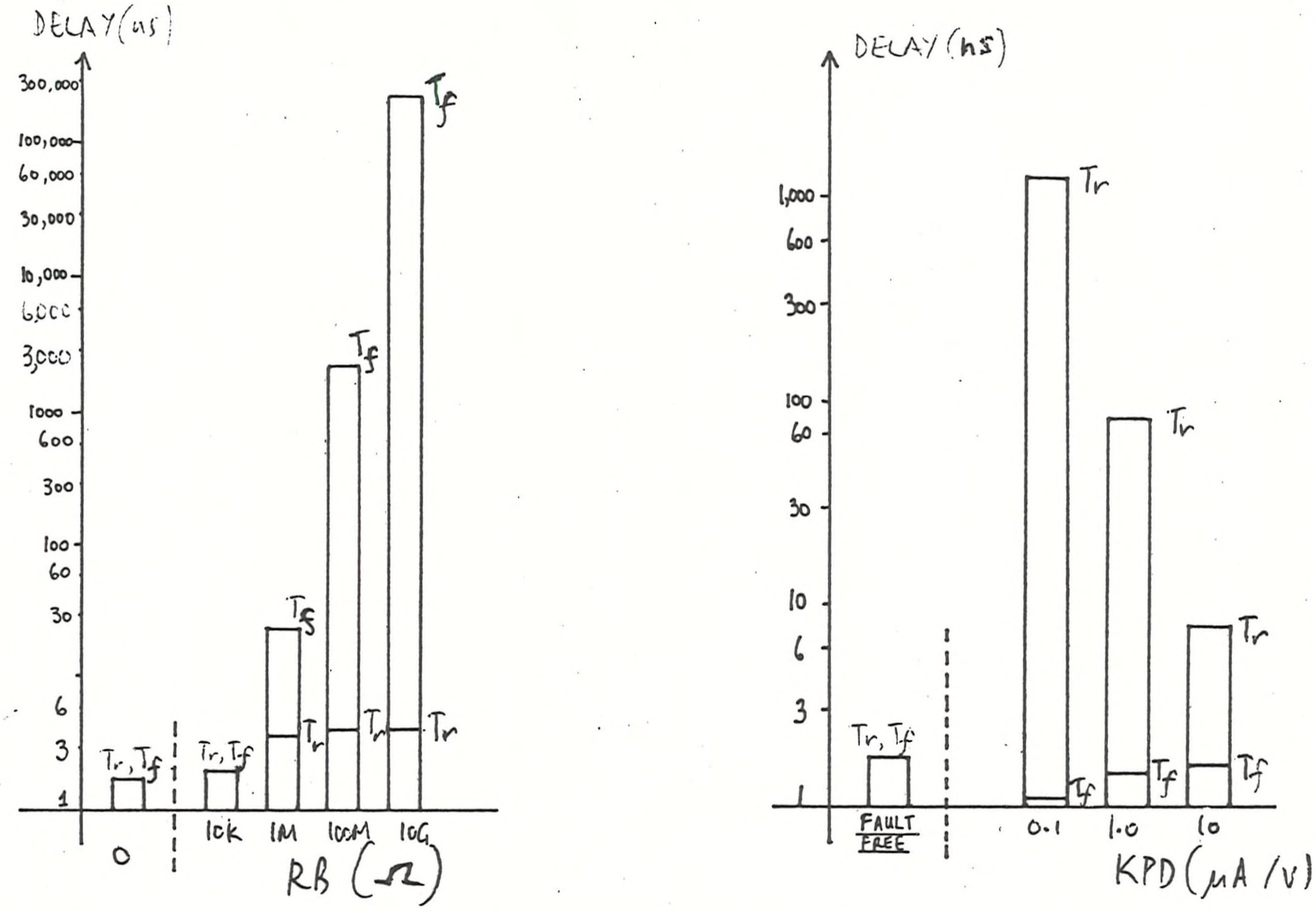
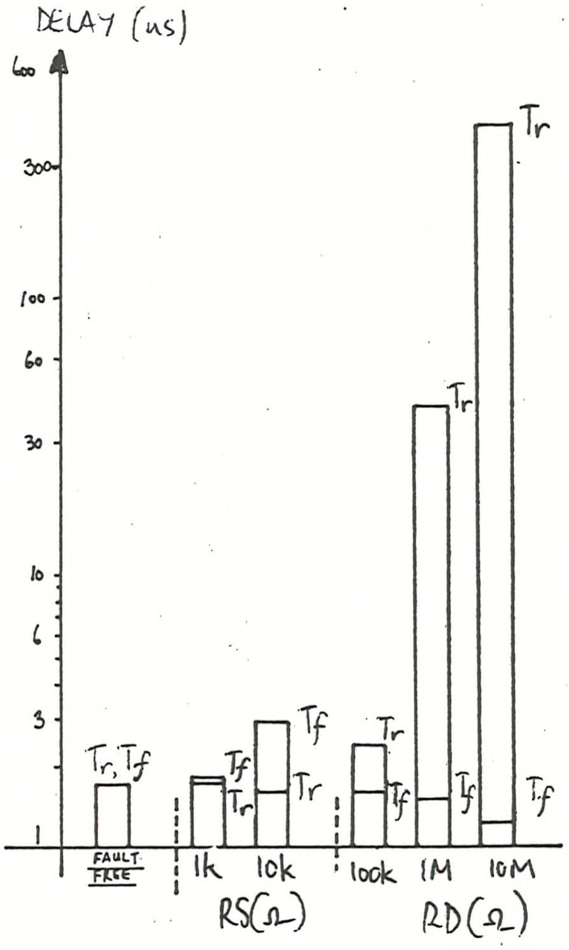


Figure 3.5(a) Buried contact resistance, RB

Figure 3.5(b) Transconductance (depletion device), KPD

Figure 3.5 Propagation delays of faulty inverters as simulated on SPICE ($t_r$: rise time, $t_f$: fall time)

Note: faults which did not produce significant variations in the propagation delays have been omitted (e.g. gate-source and gate-drain enhancement device pinholes)



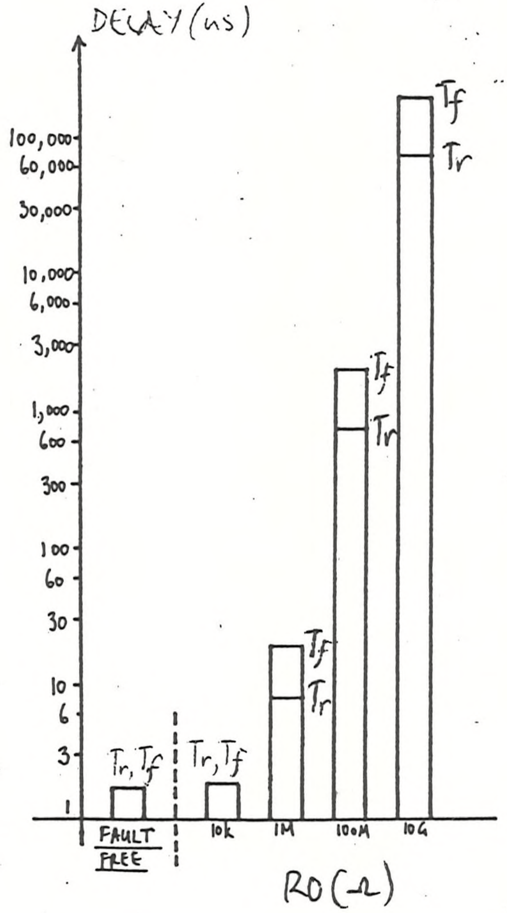


Figure 3.5(c) Contact window resistances, RS and RD

Figure 3.5(d) Output line resistance, RO

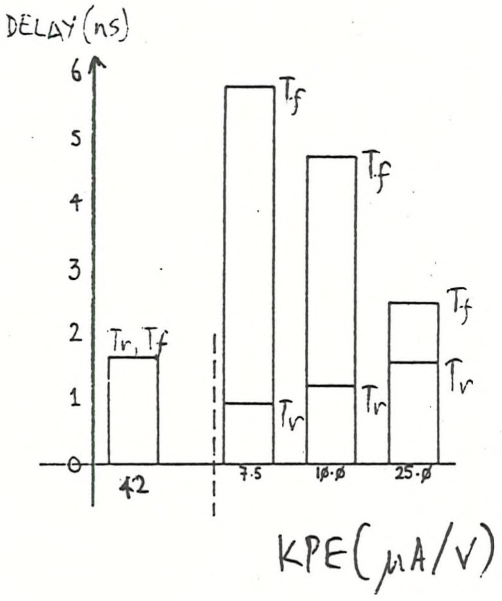


Figure 3.5(e) Transconductance (enhancement device), KPE

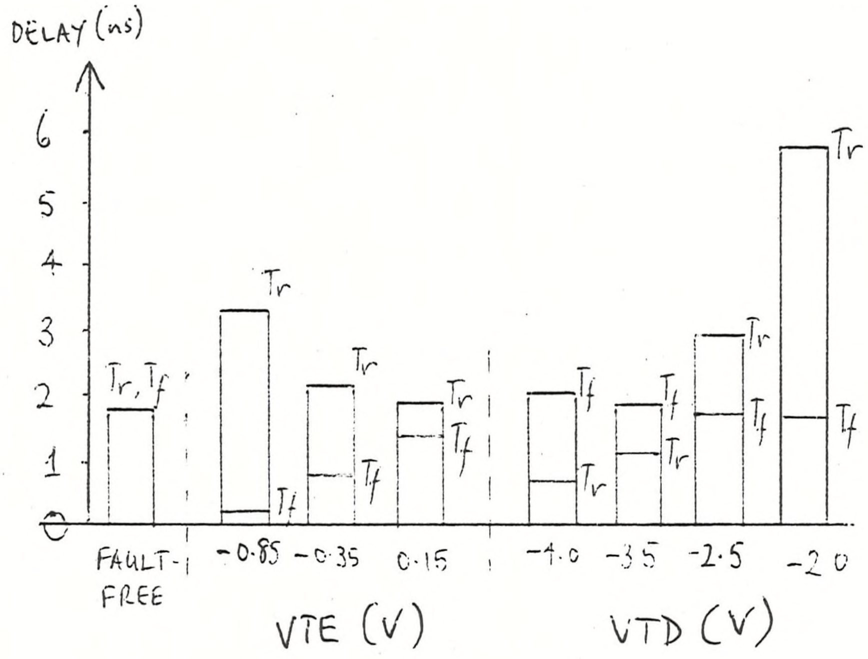


Figure 3.5(f) Threshold voltage (both devices), VTE and VTD

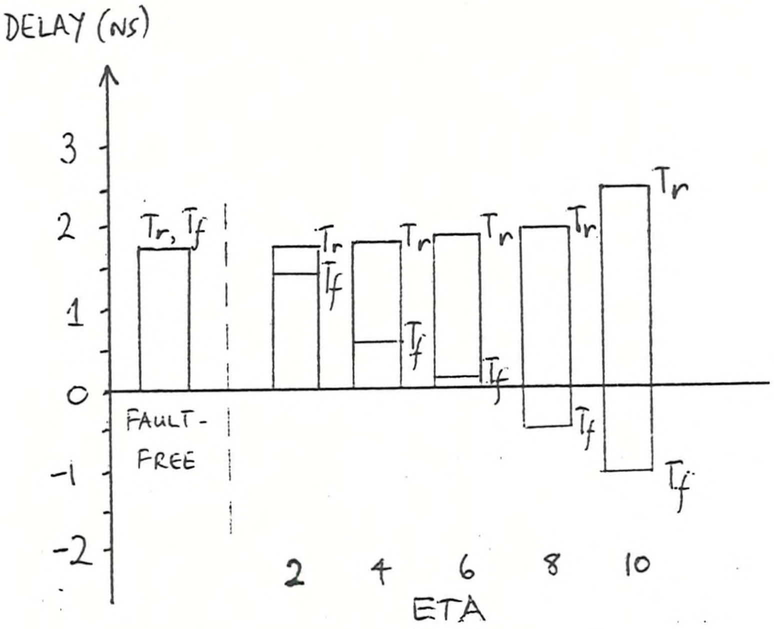
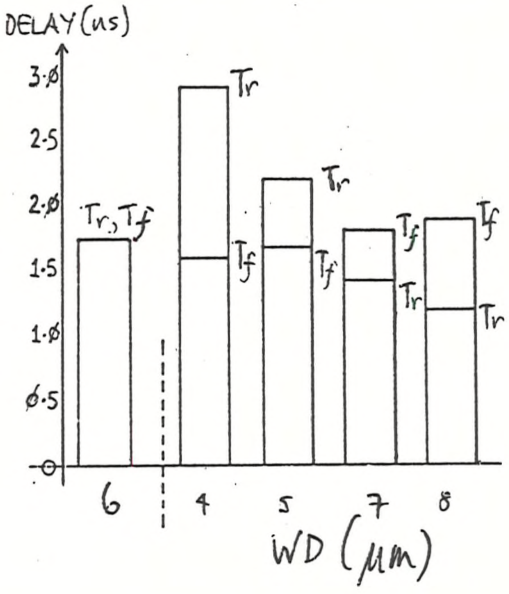


Figure 3.5(g) Subthreshold leakage current factor, ETA



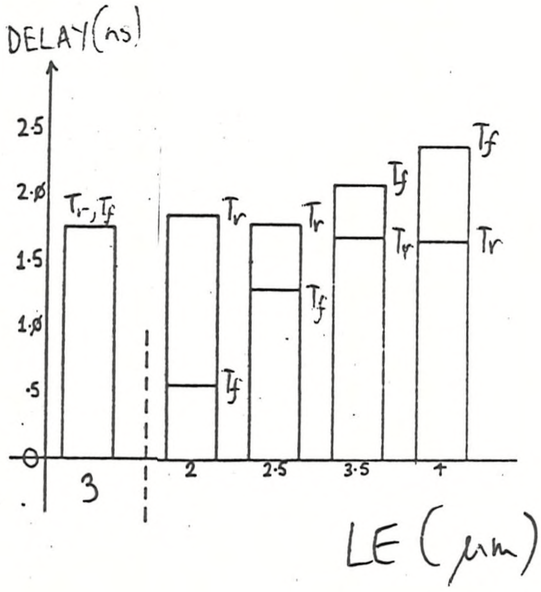


Figure 3.5(h) Transistor width (depletion device), WD

Figure 3.5(i) Transistor length (enhancement device), LE

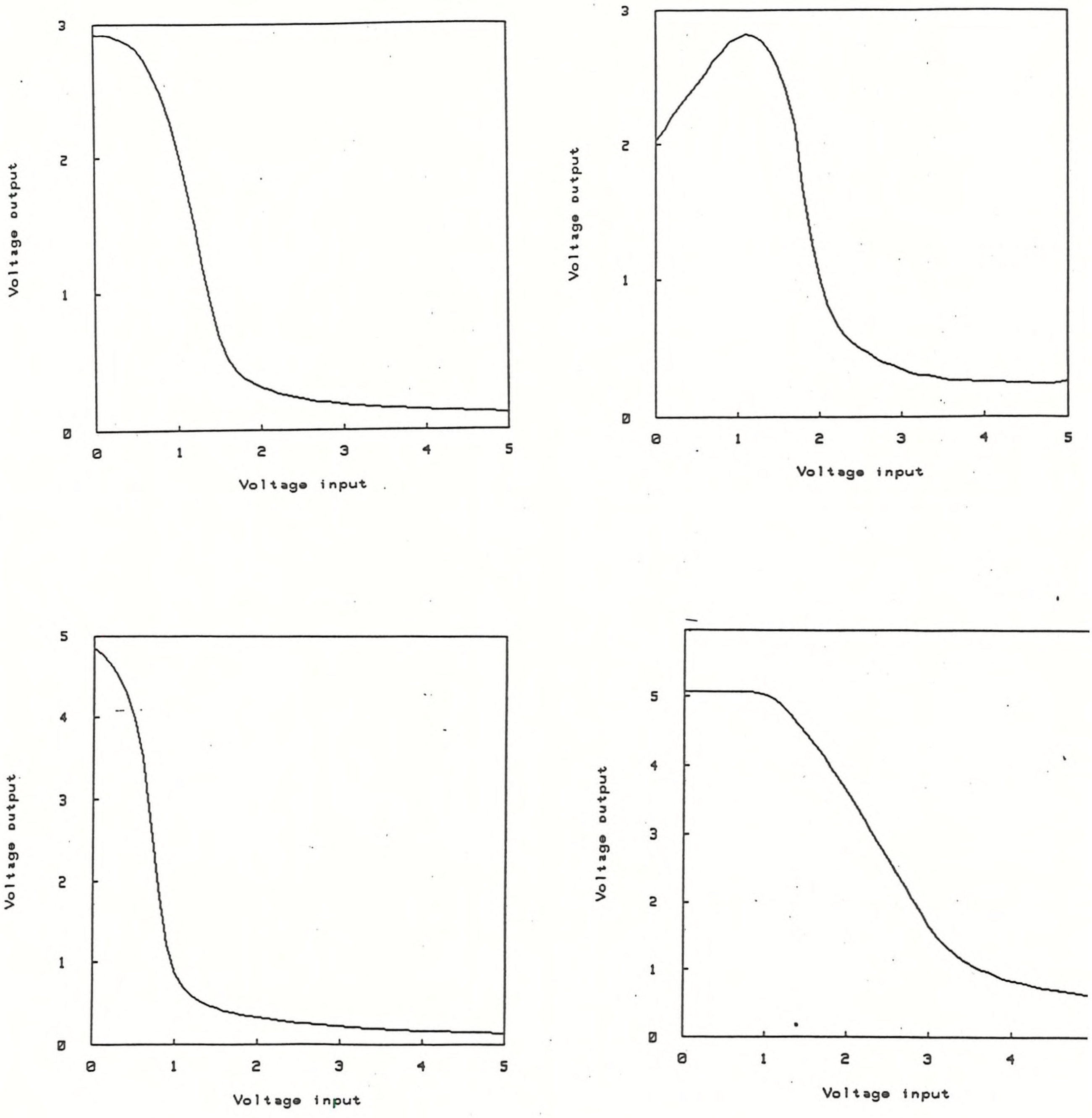
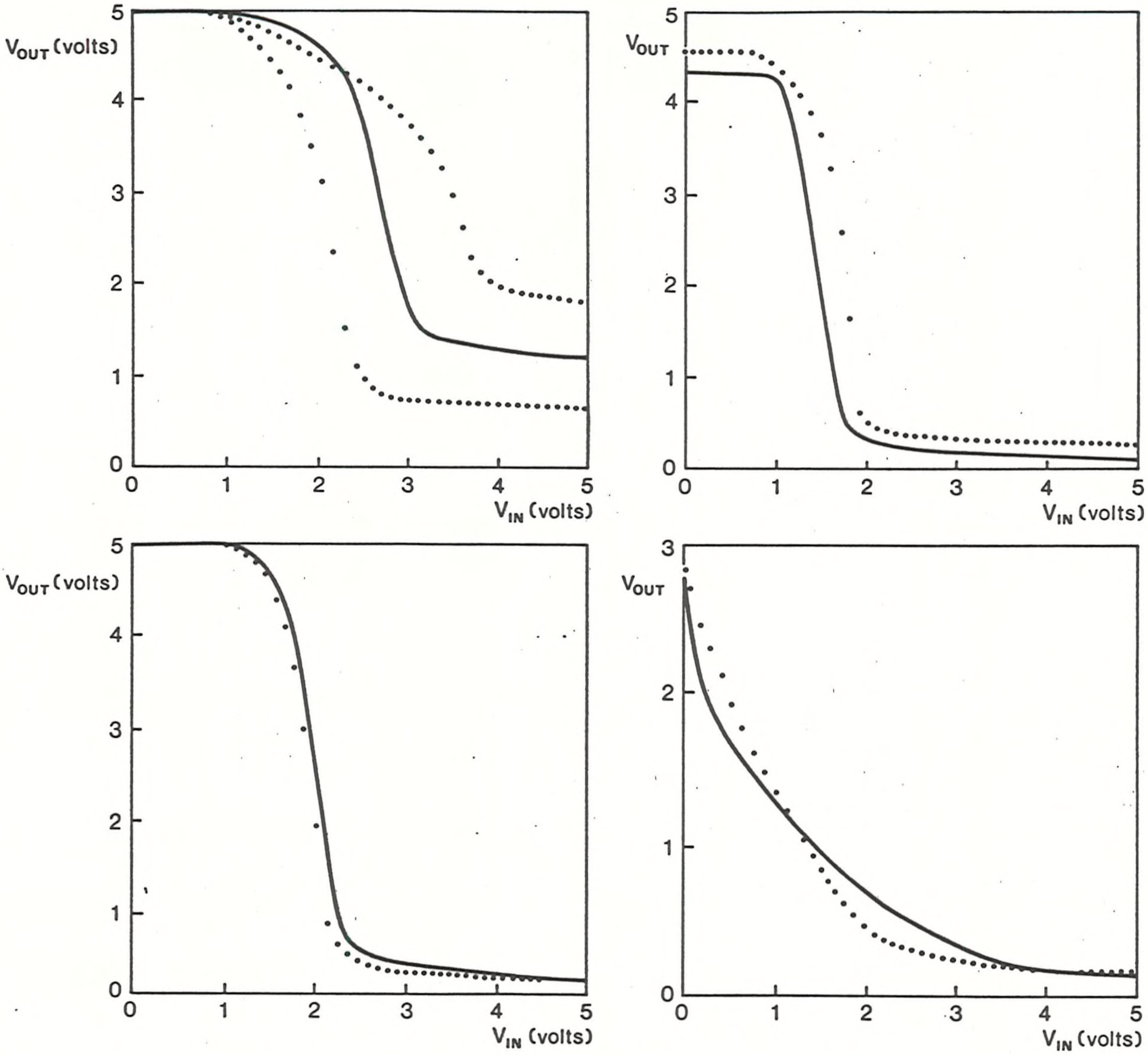


Figure 3.6(a) Transfer characteristics obtained from faulty inverters on drop-ins fabricated at Southampton



——— Drop-in characteristic
•••••• Simulated characteristic

Figure 3.6(b) Comparison of SPICE simulations and results obtained from faulty inverters fabricated at Southampton

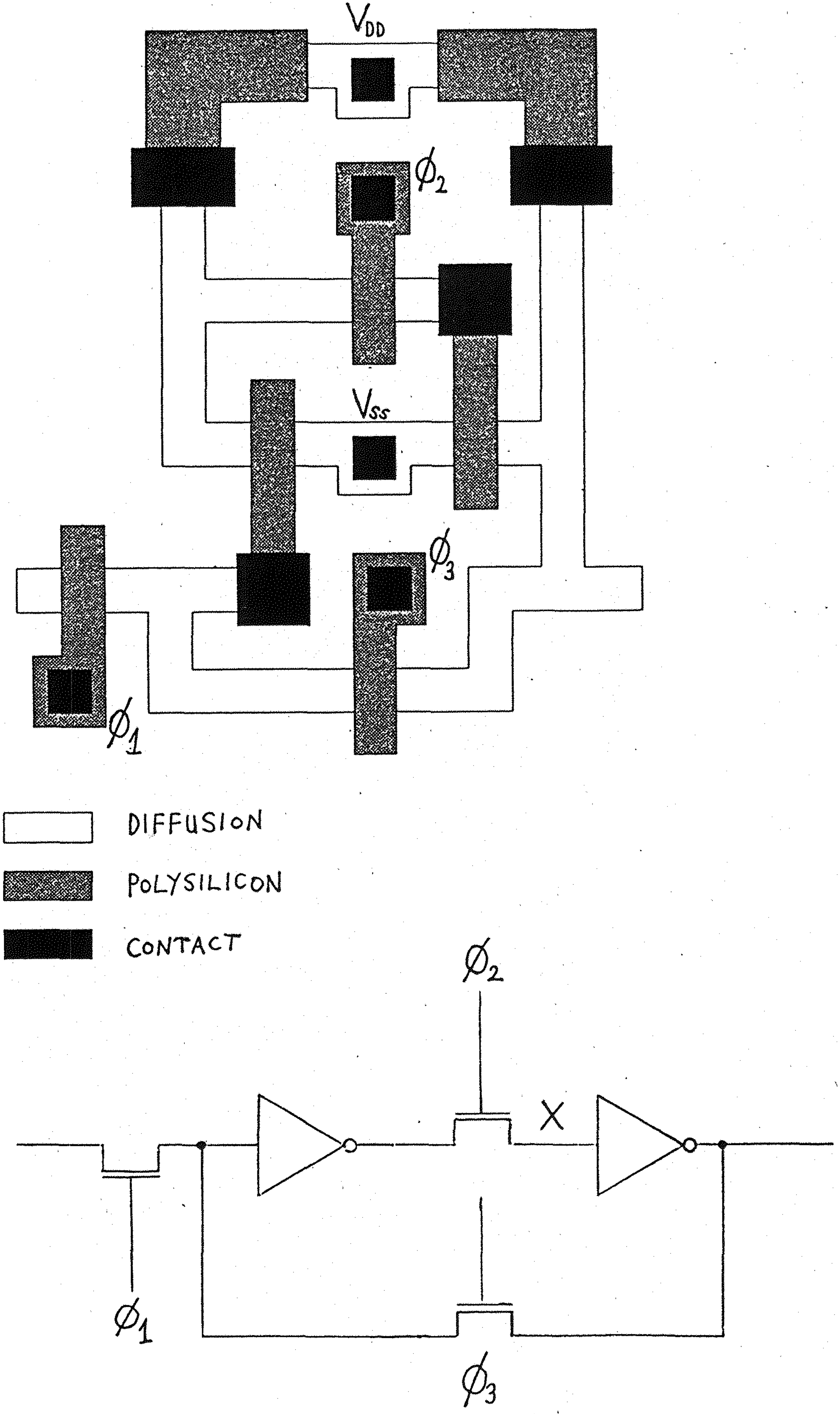


Figure 3.7(a) QSR cell layout and circuit diagram

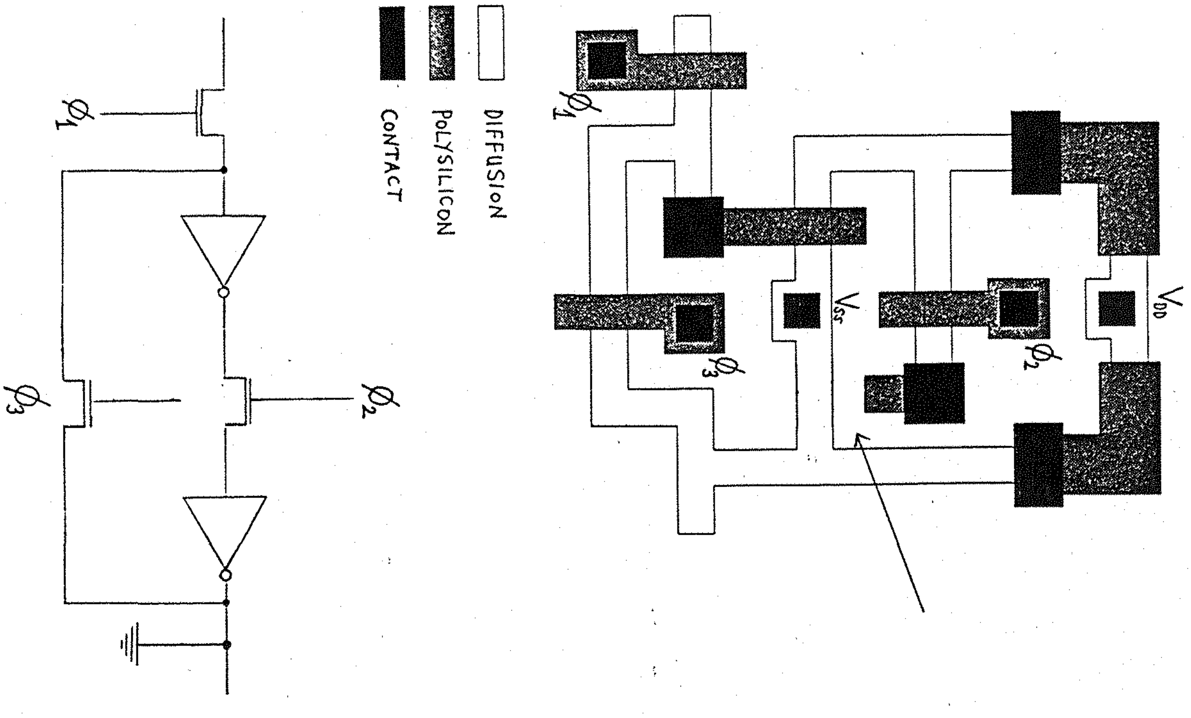


Figure 3.7(b) QSR cell: missing polysilicon

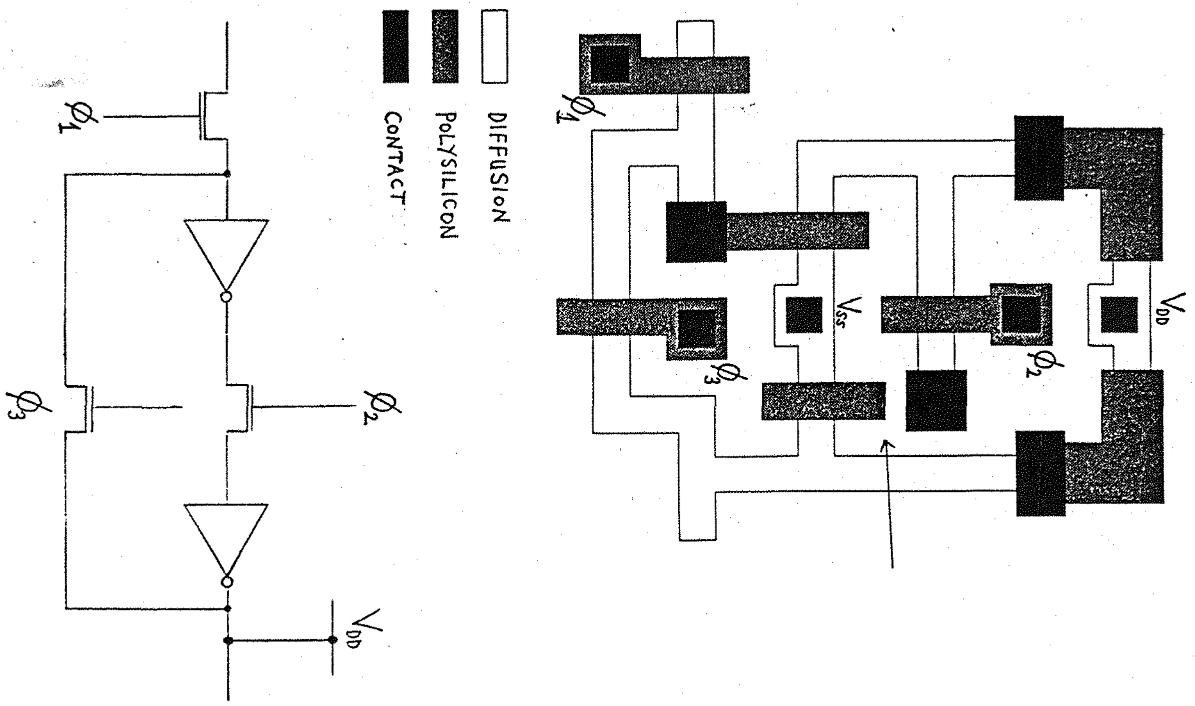


Figure 3.7(c) QSR cell: floating polysilicon gate

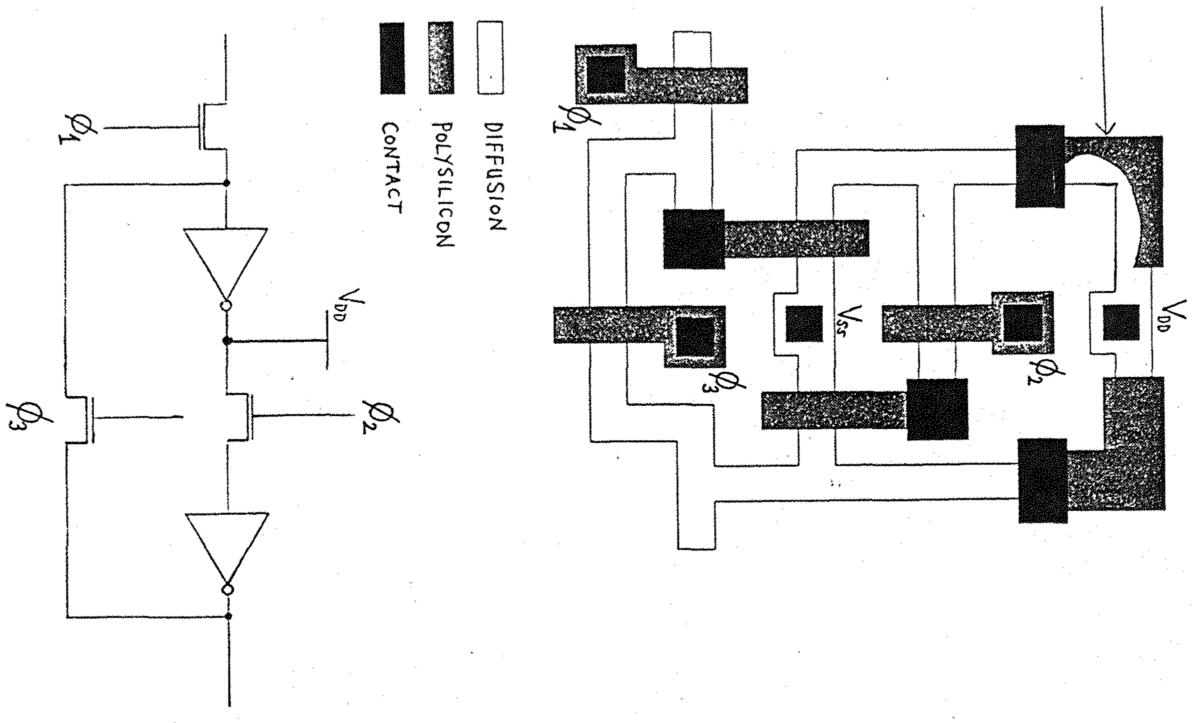


Figure 3.7(d) QSR cell: polysilicon wrongly etched

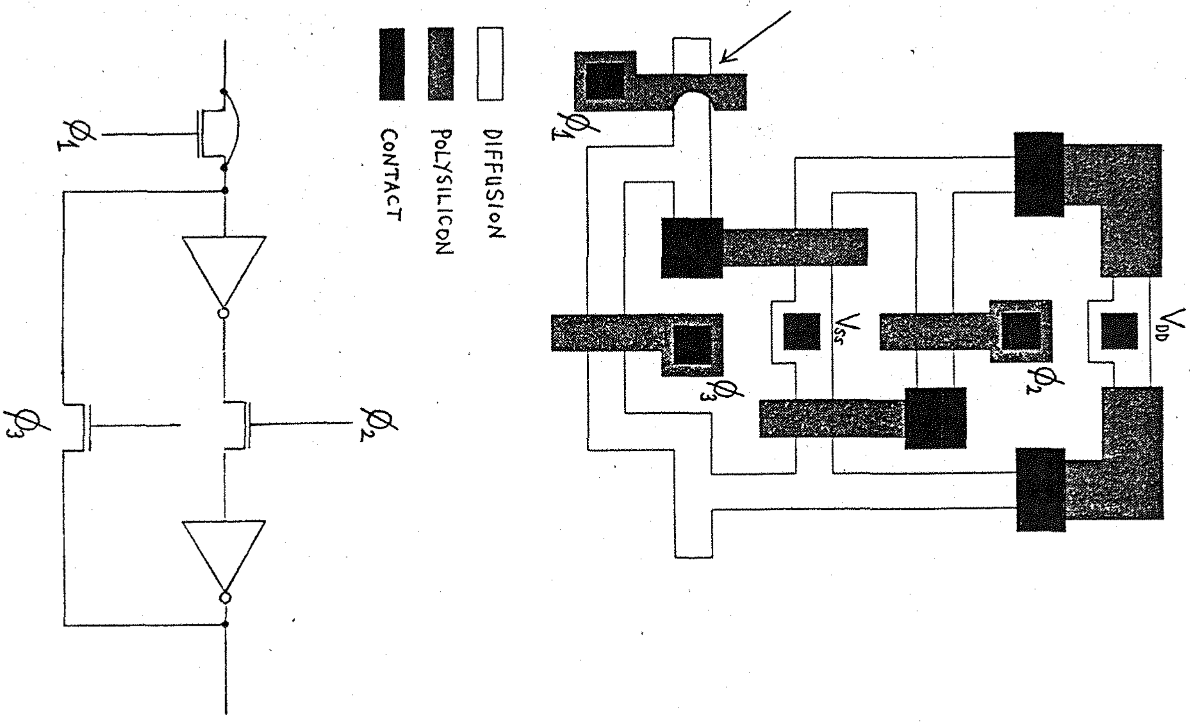


Figure 3.7(e) QSR cell: polysilicon wrongly etched

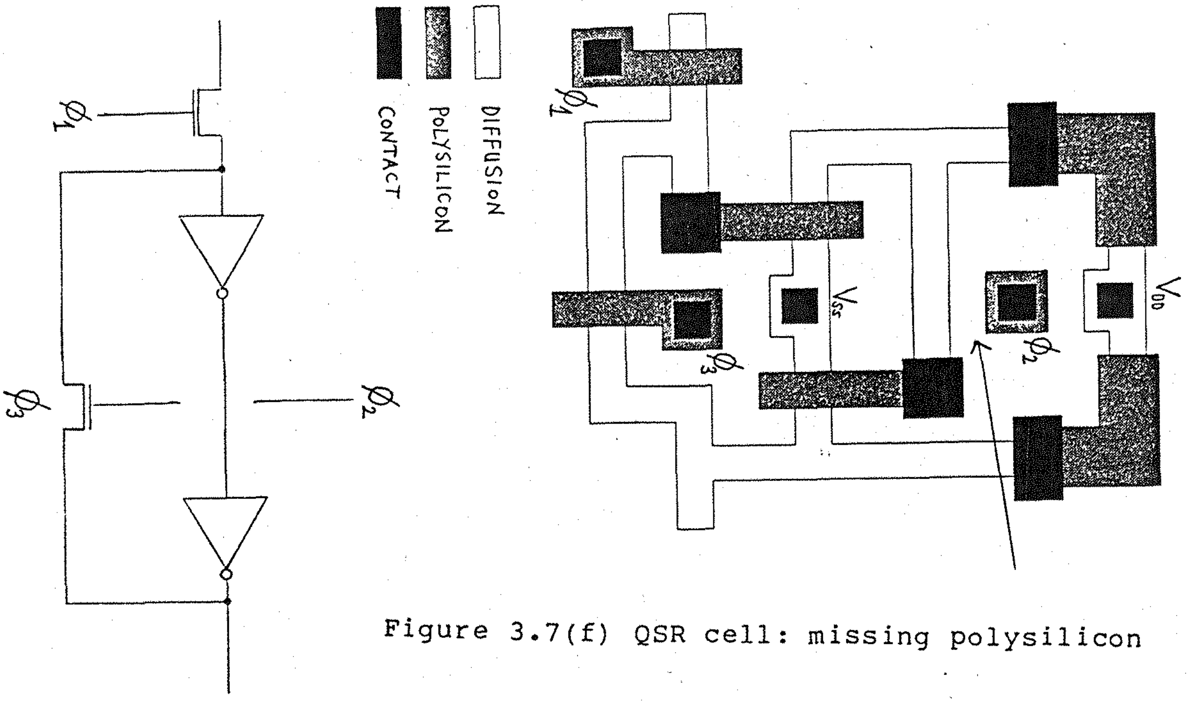


Figure 3.7(f) QSR cell: missing polysilicon

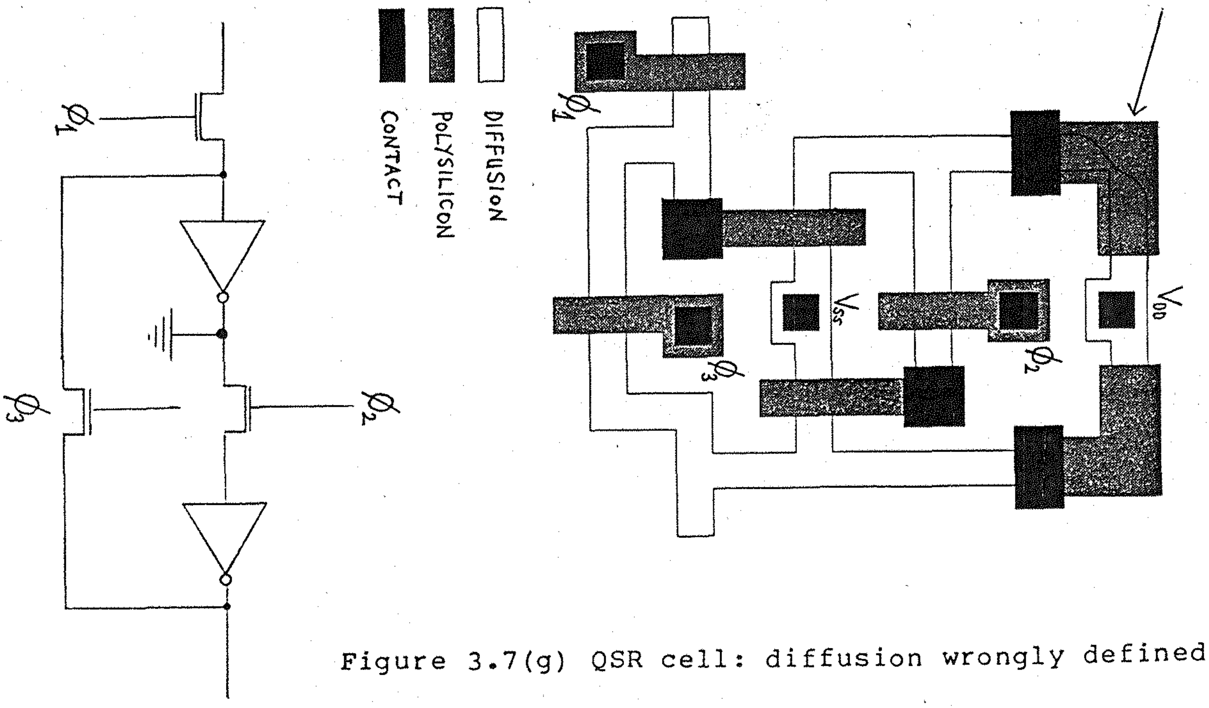


Figure 3.7(g) QSR cell: diffusion wrongly defined

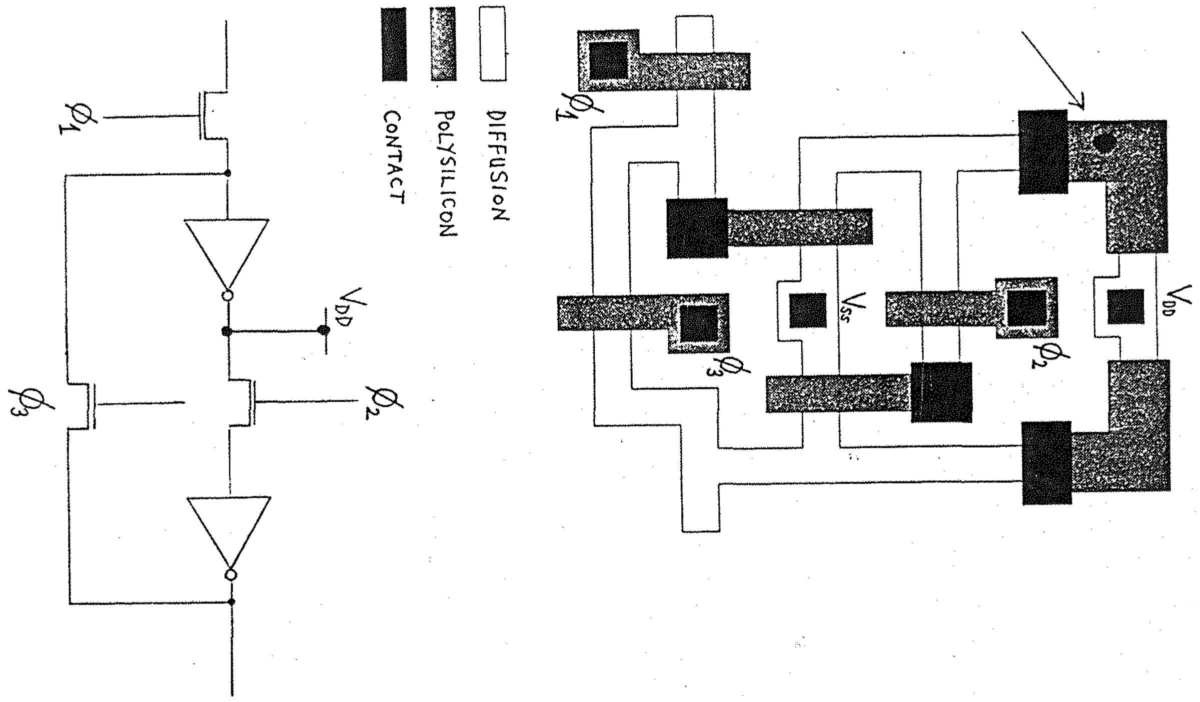


Figure 3.7(h) QSR cell: polysilicon-diffusion pinhole

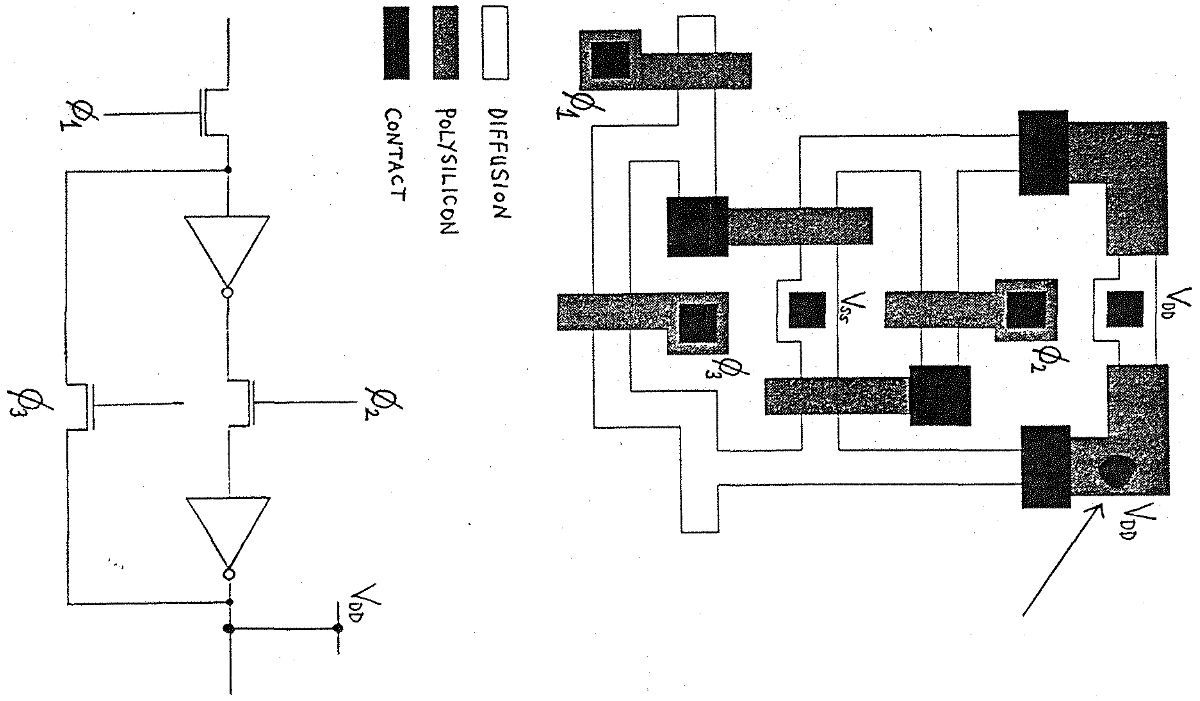


Figure 3.7(i) QSR cell: metal (VDD)-polysilicon short circuit

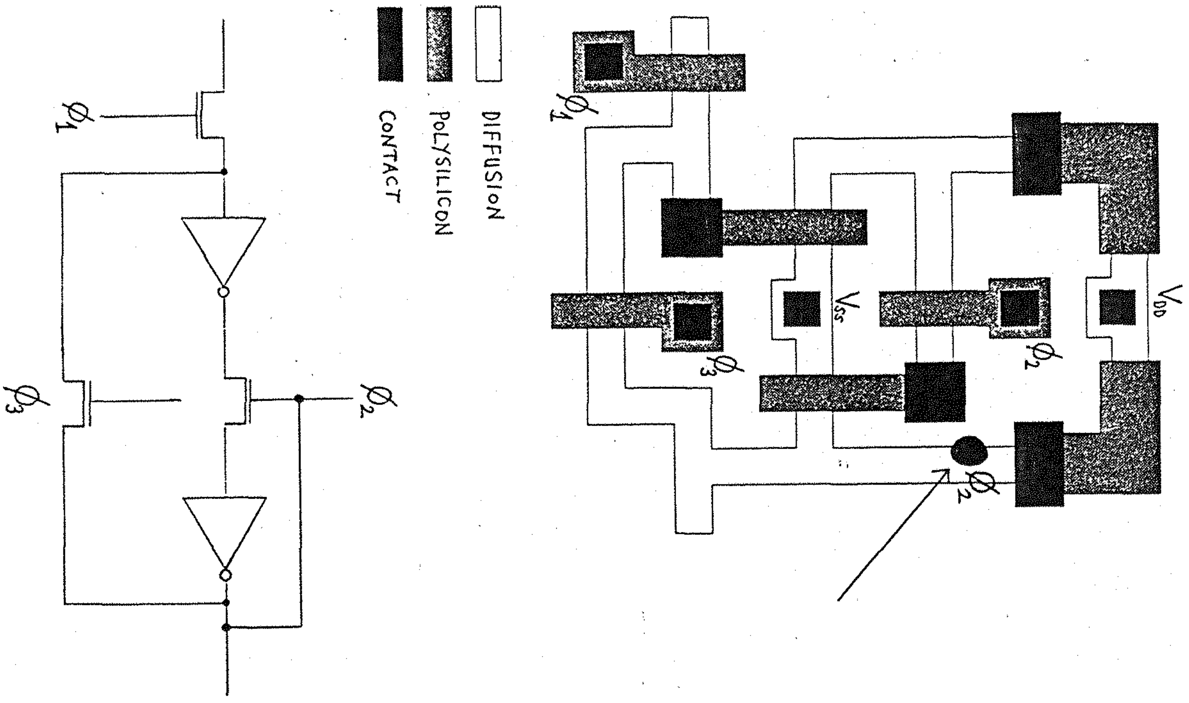


Figure 3.7(j) QSR cell: extra metal-diffusion contact window

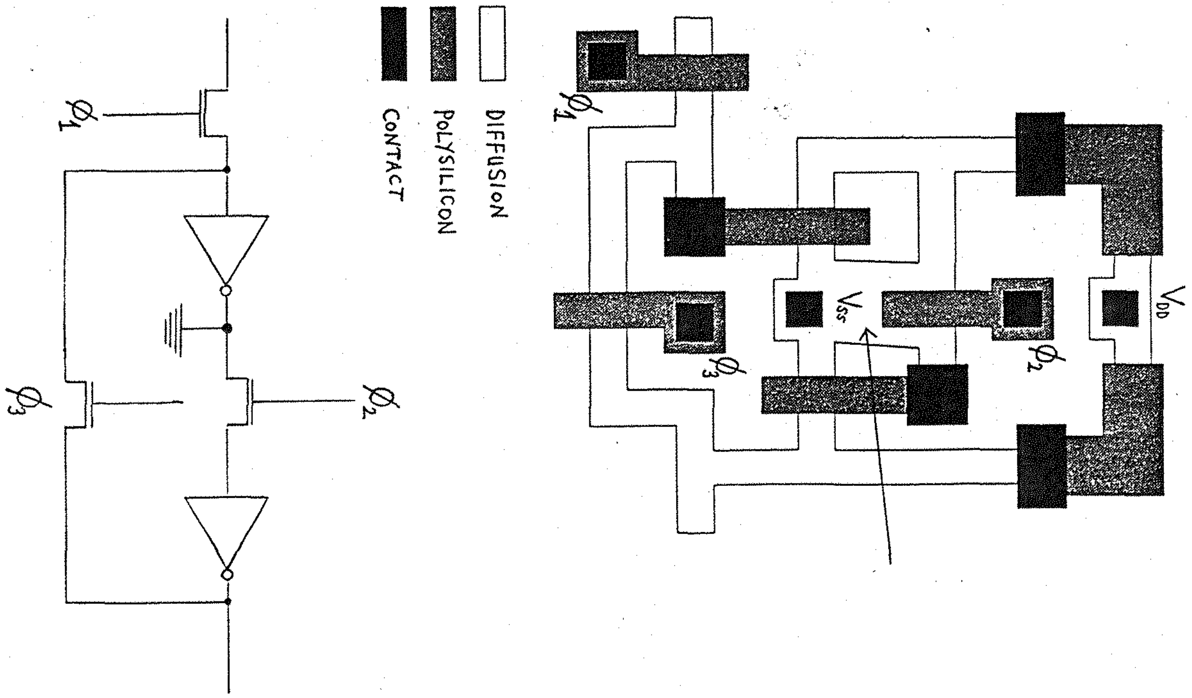


Figure 3.7(k) QSR cell: missing field oxide

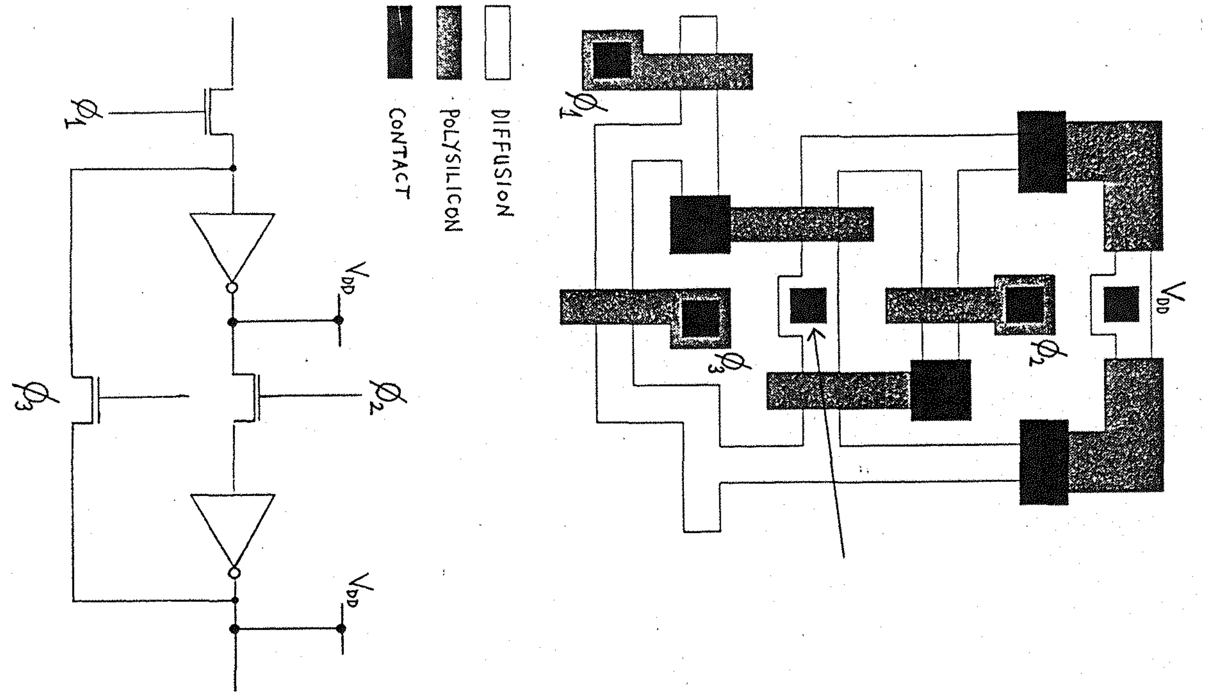


Figure 3.7(l) QSR cell: missing metal - contact window open circuit

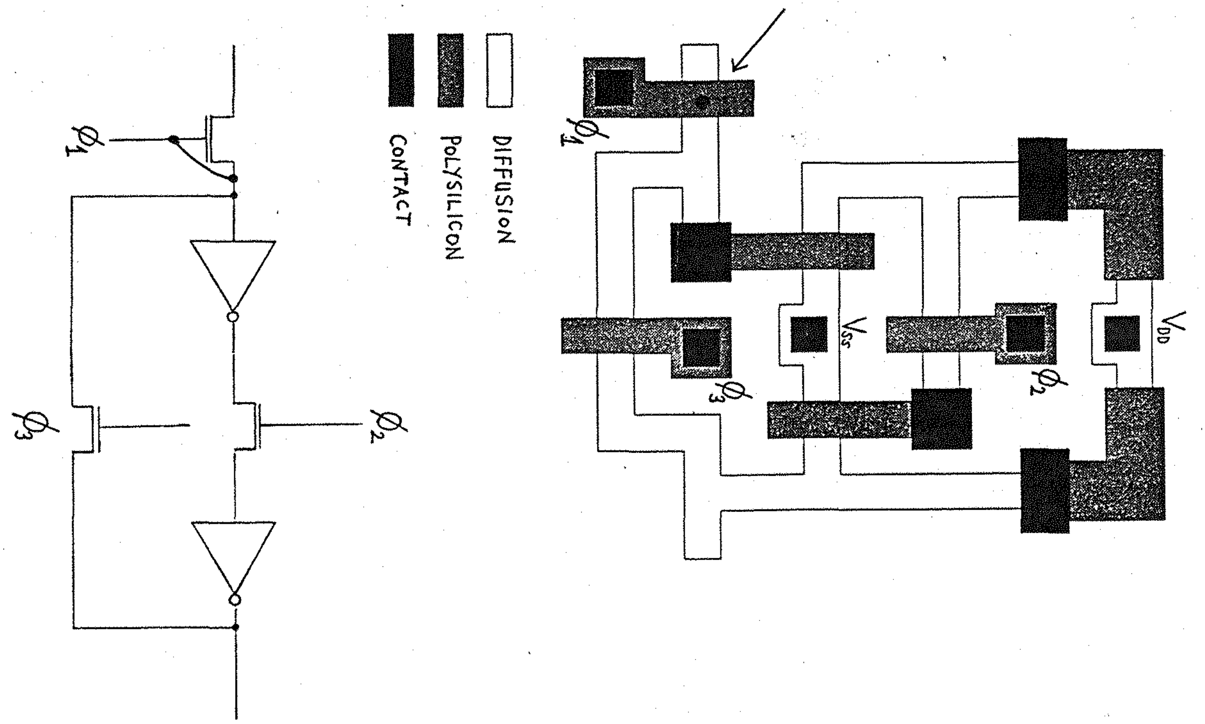


Figure 3.7(m) QSR cell: polysilicon-diffusion pinhole

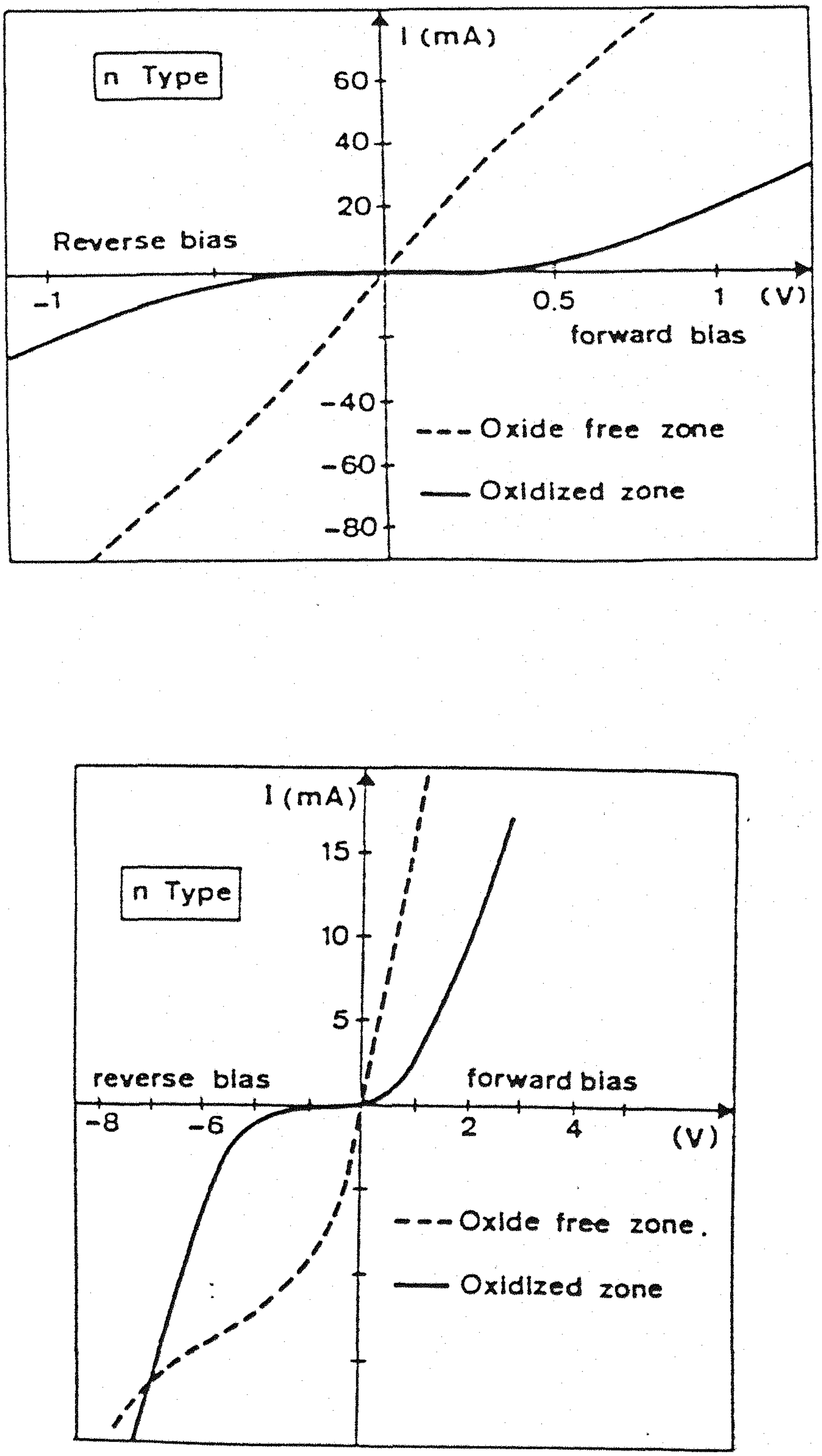


Figure 3.8(a) I-V characteristics of contact windows with a thin layer of oxide at the n-type silicon-metal interface

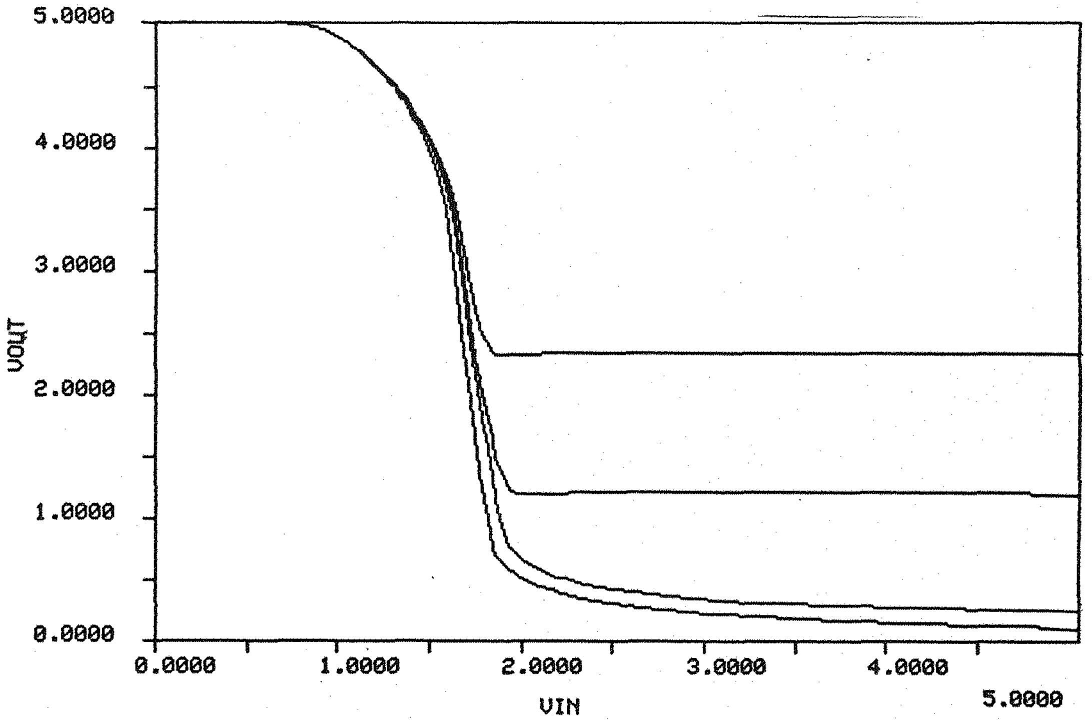


Figure 3.8(b) SPICE simulations of faulty source contact window modelled by a Schottky Barrier Diode

# CHAPTER 4 THE INADEQUACY OF THE STUCK-AT FAULT MODEL

## 4.1 Introduction

The work presented in the previous chapter showed that, in general, NMOS failure modes introduce open circuits, short circuits (bridging faults), and device characteristic degradations. However, not all these failures produce Boolean logic gate input and output "stuck" faults, and thus test routines designed to uncover even all stuck faults may not uncover all the potential physical faults in an NMOS circuit. In particular, the previous chapter identified three main areas of difficulty:

- propagation delay degradations;
- noise sensitivity problems;
- open and short circuits.

In this chapter I will examine the problems presented by these faults, together with methods that may be employed to combat them. Finally, I describe some experiments in which differently-derived test sets were used to test the same circuits, in a practical attempt to evaluate the suitability of different fault models for MOS testing.

## 4.2 Non-stuck faults - timing faults and noise-sensitive faults

Malfunctions arising from timing faults or from noise-sensitive nodes may be called "hard-to-test" faults since they are not modellable as stuck nodes. Instead of producing stuck nodes, they produce nodes that only occasionally appear to be at the wrong logic value. For timing faults, a wrong value is observed if the node, on being "sampled", has not fully charged up (or discharged). This kind of fault may be uncovered by running the test sequence at the chip's specified clock rate if possible, rather than at a lower frequency set by the automatic test equipment. A sufficiently high clock frequency will convert slowly-switching nodes to nodes that do not switch at all in the available clock period - in other words, to "stuck" nodes. If the automatic test equipment cannot exercise the chip at full speed, techniques that slow the whole chip down would need to be used, thus exacerbating any timing problems in the chip. The use of strict synchronous circuit design techniques for VLSI has contributed greatly to making VLSI circuits less prone to "hard-to-test" timing faults (which require two inputs to check a propagation delay), as well as easing timing analysis by permitting the use of zero-delay logic simulation. (The LSSD design philosophy has the reduction of system dependence on ac parameters as one of its aims).

A node that has been made noise-sensitive as a result of a fault in a chip is likely to display an "intermittent" fault-effect - sometimes the node will switch correctly; at other times, it will not. A node that is "floating" as a result of an open circuit will also be susceptible to this kind of fault-effect, as it may be controlled by signal pick-up from an adjacent track (ref 1) or the power rails (ref 2). This type of fault is not modellable as a stuck node, as the node may have to be switched several times to uncover an incorrect transition. Alternatively, techniques that alter the gate input and output voltage levels across the chip might be used to exacerbate the fault at the noise-sensitive node.

The conditions under which a chip is tested can be manipulated in order to increase the probability of uncovering timing faults and noise-sensitive faults. Hunger and Gaertner (ref 3) have shown how faulty behaviour may be immediately uncovered in apparently fault-free chips by using standard 'burn-in' techniques, such as increasing the chip temperature, and/or altering the power supply. Further SPICE simulations were thus performed to investigate how an apparently fault-free device containing a "hard-to-test" fault could be converted into a faulty device by altering the chip temperature or the power supply voltage.

## 4.3 SPICE Simulations

There are three main MOS integrated circuit parameters which are temperature-dependent:

transconductance,
threshold voltage,
p-n junction reverse leakage current.

An empirical formula for the reduction in transconductance (due to the reduced mobility of the charge carriers) is

$$K = K_o \, (T/T_o)^{-3/2}$$

where K is the value of the transconductance $K_0$at room temperature $T_0$ , and T is the operating temperature in degrees Kelvin. For a 100 K increase in temperature, the transconductance reduces to 65% of its room temperature value, thus reducing the circuit's operating speed, because the current available to charge up the capacitative output node of an NMOS logic gate is lower. (Pass transistor networks will also exhibit this reduction in speed because the "on-resistance" of a MOSFET is increased as a result of a decrease in transconductance).

The threshold voltage of an n-channel MOSFET reduces by approximately 1 mV/K due to changes in the bulk silicon Fermi level with temperature. A 100 K increase in temperature reduces the threshold voltage by 0.1 V; however, this shift is not nearly so significant as the change in transconductance on circuit performance, and in any case is typical of the variation in threshold voltage observed in silicon processing.

Finally, the reverse leakage current of p-n junctions doubles for every 8-10 K rise in temperature. A 100 K increase in temperature thus typically increases the leakage current by three orders of magnitude, producing significant reductions in the output high voltage of a logic device. SPICE simulation results showing these temperature-dependent effects are presented in Figure 4.1.

These results show that 'baking' a chip may uncover faulty behaviour not detected at room temperature. In particular, voltage level degradations resulting from reverse diode leakage to the substrate may be converted to output stuck-at-0 faults, and faulty behaviour induced by transconductance degradations in transistors will be exaggerated.

Acceptance test routines often contain a test run at a lowered $V_{DD}$ as experience shows that this uncovers additional failures in chips that have previously passed a test at normal $V_{DD}$. Figure 4.2(a) shows the simulated dc transfer characteristics with $V_{DD}$ = 5 V, 4.5 V, 4 V and 3.5 V of an inverter with a faulty $V_{SS}$ contact (modelled by a Schottky barrier diode with $I_S = 5 \times 10^{-8}$ ). This fault can produce a stuck-at-1 output if severe enough. The reduction in $V_{DD}$ produces a corresponding reduction in $V_{OH}$, without changing $V_{IH}$ or $V_{OL}$, and a small incrase in $V_{IL}$. Thus, reducing $V_{DD}$ does not affect the faulty part of the characteristic - in fact, the curves are identical to the normal $V_{DD}$ curve at $V_{OUT} \leqslant (V_{DD} - 1.85\text{ V})$ for the different $V_{DD}$'s.

Figure 4.2(b) shows a similar set of curves for a fault that, if severe enough, produces a stuck-at-0 output. The specific fault depicted is excessive subthreshold leakage current (modelled by the SPICE parameter ETA).

As before, the portions of the characteristic where $V_{out} \leqslant (V_{DD} - 1.85\ V)$ are identical to the normal $V_{DD}$ curve. In this case, however, the upper portions of the curve ($V_{out} > 1.65\ V$) do not show the same large variation with $V_{DD}$ as seen in Figure 4.2(a). Thus, reducing $V_{DD}$ only marginally alters the faulty part of the characteristic.

The cause of these effects is the saturation of the depletion load transistor. Simple analysis shows that if $V_{DS} \geqslant (V_{GS} - V_{TH})$, an MOS transistor is saturated, and any further increase in $V_{DS}$ produces almost no increase in $I_{DS}$, the current flowing through the device. For the case of the depletion load transistor in an NMOS logic circuit, this equation is rewritten $(V_{DD} - V_{out}) \geqslant (0 - V_{TH})$, which gives $(V_{DD} - 1.85) \geqslant V_{out}$ for saturation to occur (the depletion transistor's threshold voltage, $V_{TH}$, is $- 1.85V$). If $V_{out}$ is less than 1.65 V this equation is satisifed for $V_{DD} = 5$ V and 3.5 V simultaneously, resulting in identical transfer characteristics for the various power supply voltages below this value of $V_{out}$.

The differences between the curves in Figure 4.2(b) are less than those in Figures 4.2(a) because in Figure 4.2(b) the depletion transistor, although not saturated, is in the transition between the saturation region and the linear region of operation. This means that the current it passes is not significantly different from the saturation current, and thus the transfer characteristics do not differ above $V_{out} = 1.65V$ as markedly as in Figure 4.2(a), where the depletion transistor is operating in its linear region for most values of $V_{out}$ above 1.65V.

Lowering $V_{DD}$ does not therefore uncover faults as a consequence of altering the input and output logic high and low voltages. Rather, as Hunger and Gaertner (ref 3) have also shown, reducing $V_{DD}$ reduces the operating speed of a component. This is especially true of a component containing pass transistors, since the "on-resistance" of a MOSFET is a function of its gate-source voltage, and a pass transistor presents an R-C load rather than a purely capacitative load. Thus decreasing $V_{DD}$ significantly alters the time constant associated with a pass transistor load. Table 1 shows results obtained from SPICE simulations of inverters driving differing loads operating at several values of $V_{DD}$. The timing degradations are significantly worse for the pass transistor loads. The rise times for the capacitative loads remain more-or-less constant because the depletion transistor is saturated for most values of $V_{DD}$ while $V_{out}$ charges up to $V_{INV}$. The fall times decrease with $V_{DD}$ because the output logic 1 value decreases with $V_{DD}$, and less charge is stored on the output capacitance as a result.

In conclusion, marginal "hard-to-test" faults may be uncovered by altering the ambient conditions during a test run. Increasing (rather than decreasing) $V_{DD}$ can also be a useful technique in screening out marginally good devices because the stronger electric fields will induce faulty behaviour as a result of any structural weaknesses (crystal

TABLE 1

| | VDD: | 5V | 4.5V | 4V | 3.5V |
|---|---|---|---|---|---|
| Load | | | | | |
| Two inverters | tr | 2.5 ns | 2.4 | 2.3 | 2.2 |
| | tf | 3.3 ns | 3.2 | 3.0 | 2.8 |
| Four inverters | tr | 2.8 | 2.6 | 2.4 | 2.3 |
| | tf | 4.4 | 4.4 | 4.3 | 4.1 |
| One pass transistor | tr | 2.9 | 2.8 | 2.7 | 2.5 |
| | tf | 4.9 | 4.9 | 4.9 | 6.4 |
| Two pass transistors | tr | 3.7 | 3.6 | 3.5 | 3.4 |
| | tf | 7.2 | 7.3 | 7.6 | 9.2 |
| Four pass transistors | tr | 6.2 | 6.3 | 6.4 | 6.5 |
| | tf | 12.3 | 12.8 | 14.4 | 20.6 |

defects, pinholes etc) in the chip. Thus, the current practice of burning-in devices before selling them greatly reduces the chance of a manufacturer shipping a low reliability component. Indeed, burn-in tests have been found to uncover marginal behaviour in chips not detected by tests performed under normal operating conditions (refs 4,5).

## 4.4 Non-stuck faults - open and short circuits

In this section, I examine the fault-effects associated with stuck-open and stuck-on transistors, and with open and short circuit interconnects. The nature of a fault-effect depends on the configuration of the circuit in which a fault occurs, and in the following examples faults giving rise to stuck nodes and non-stuck nodes are presented. Figure 4.3 shows a typical NMOS gate realising the function $Z' = a(b + c)$. In this circuit, any transistor or interconnect fault may be modelled as a stuck-at fault on the equivalent Boolean logic gate diagram. This is because any such fault converts one of the function variables (input or output) to a permanent logic one or zero. The Boolean logic diagram is a graphical representation of the prime implicants in the function being realised, and thus any such fault can readily be inserted as a stuck-at fault on the Boolean logic gate diagram. However, this is by no means always the case.

Figure 4.4, for instance, shows a circuit containing a fault that is not modellable as a stuck fault. The original function is $Z' = (a + c)(b + d)$, but the open circuit removes the minterms abcd = 1001 and 0110 thus changing the logic function to Z' = ab + cd. This alteration cannot be represented as a stuck fault on the Boolean logic diagram, since none of the function variables have been entirely eliminated by the fault. Instead, the fault is on a piece of interconnect that does not appear in the Boolean logic diagram, and thus the faulty circuit may only be modelled by re-drawing the Boolean logic diagram.

Another example of this problem is illustrated in figure 4.5 which shows two different NMOS implementations of the XNOR (logical comparison) function. According to the stuck-at fault model, this single logic gate may be fully tested by applying any three of the four possible input combinations. However, implementation (a) needs all four combinations to be applied to cover all transistor and interconnect faults, and implementation (b) needs all four input combinations to be applied in the correct order to uncover all possible transistor and interconnect faults (see section 1.5). The main conclusion to be drawn from both these examples is that the stuck-at fault model operates at the wrong level of abstraction, since the structural information needed to model accurately faults in NMOS circuits is not available at the Boolean gate level of abstraction.

Finally, figure 4.6 shows a pass transistor implementation of a rotator circuit, and its equivalent Boolean logic diagram. This circuit forms the basis of the barrel shifter circuit often used in digital signal processing chips to provide fast variable shifts (to the left or the right) of a data word (ref 6). The test for line L stuck-at-1 on the gate-level diagram (the equivalent MOSFET fault is transistor T1 stuck-on) is abcd = 1X01, where X is the "don't care" state. This input propagates the fault-effect to output f where a 1 instead of a 0 will appear as a result of the fault. SPICE simulations of this circuit (modelling transistor T1 as stuck-on by decreasing its threshold voltages to a negative value) show that, contrary to predictions based on the gate-level diagram, output f remains at 0 even in the presence of this fault. This is because the current flowing out of the inverter on input c flows through both transistors T1 and T2 whereupon it is sunk by the inverter on input d, thus pulling node f low.

Similarly, the test for line L stuck-at-0 (the equivalent fault in the gate-level diagram is transistor T1 stuck-off) is abcd = OX0X. However, in the pass transistor network this fault produces a memory state, because output f is left floating, retaining its previous logic value by charge storage on the associated output capacitance from the previous input conditions. For example, suppose the two vectors abcd = 1000 and 0000 were applied in that order to the faulty circuit. The gate-level diagram indicates that output f would be set first to logic 1, and then to logic 0.

The transistor diagram, on the other hand, indicates that output f would remain at logic 1 for both inputs, on account of the charge storage at the output node. The real problem is that certain MOSFET characteristics (bi-directionality and the high impedance tri-state condition) are not modelled by the Boolean logic gate diagram. Thus, fault-effects that arise as a result of these characteristics are not predicted and hence will only be covered fortuitously by a test set derived using the stuck-at fault model operating on the gate-level representation of a circuit.

## 4.5 Evaluation of different fault models

The above analysis has shown how the stuck-at fault model does not adequately cover certain common faults in NMOS circuits. To confirm this analysis, a circuit was designed in NMOS, fabricated at the University's Microelectronics Centre fabrication facility, and tested with differently - derived test input sequences. The circuit chosen was a 16-to-1 multiplexer because it is typical of NMOS circuits that do not map onto Boolean logic diagrams, and because it contained less than 200 transistors - the maximum size of circuit necessary to obtain a spectrum of catastrophically faulty, slightly faulty and fault-free circuits from the fabrication facility.

The multiplexor was designed and implemented as a "function block" (ref 6), consisting of an array of enhancement- and depletion-mode transistors (Figure 4.7(a)). In this particular realisation, only one of the input diffusion paths is selected by the polysilicon control lines for any set of control inputs. The depletion-mode transistors are always 'on' and act as simple R-C networks. However, this implementation has no contact windows, and thus there is no opportunity to study the effects of possible contact window defects on the circuit's behaviour. An alternative implementation of the function block exists in which the control lines are put on metal, and polysilicon stubs are used to create enhancement-mode transistors where necessary (Fig 4.7(b)). Therefore, in order to cover as many possible failure modes as possible the multiplexor had two select inputs driving polysilicon lines, and two driving metal lines with polysilicon stubs. In addition, all inputs were inverted in an attempt to emulate conditions deep within an LSI chip, where off-chip driving conditions have no effect on the electrical behaviour of fault within a circuit element (Figure 4.8).

Five different test sets were written for the multiplexor chips, and all five were applied to each chip. The particular concern was to see if any circuits passed some tests but failed others. The test patterns were written using different fault models appropriate to different levels of circuit representation:-

* Functional or "black box" level, testing for primary input and output stuck-at faults;

* assumed Boolean gate-level (using a TTL equivalent logic description of the circuit consisting of 16 5-input AND gates and 1 16-input OR-gate), testing for all gate input and output stuck-at faults;

* transistor-level (using tri-state elements to model the individual enhancement-mode transistors), testing for all transistor stuck-on and stuck-open faults.

The transistor-level test set generation was performed twice: firstly, using the HITEST automatic test pattern generation system (ref 7); and secondly, by hand. For the latter test set, SPICE simulations of the multiplexor were performed to determine the fault-effects of a range of "known likely" faults (interconnect open and short circuits, pinholes, stuck-on and stuck-off transistors). Figure 4.9 is a diagram of a 2-to-1 multiplexor which will be used to desribe the fault-effects found.

If transistor 3 is stuck-off, a parasitic latch is formed at the output, because it is possible for transistors 3 and 4 to be off simultaneously (if control, c = 1). If transistor 3 is stuck-on, and the input abc = 010 is applied, no fault is observed (a logic 1 correctly appears on the output node) because transistor 2 is able to sink the current from both depletion transistors. Conversely, if the input abc = 100 is applied, a wrong output is produced as transistor 1 sinks all the current. This behaviour could not be deduced from a Boolean logic diagram because the fact that transistors 3 and 4 can act bidirectionally does not show up on the equivalent logic diagram (ref 8). In this case, current flowing through transistor 3 towards the output node then flows "backwards" and away from the output node through transistor 4. The output node acts as a wired-AND gate, with conflicts between a "1" and "0" at the output being resolved in favour of the "0". This occurs because the output low resistance of an NMOS gate is much lower than its output high resistance, thus producing a wired-AND effect (the on-resistance of a depletion load transistor is at least an order of magnitude higher than that of an enhancement driver transistor). SPICE simulations to determine the fault-effects introduced by pinholes which lead to shorts between the gate and source or drain regions of a MOSFET showed that similar wired-AND behaviour is observed for these faults as well. The test pattern sequence for the multiplexor must therefore uncover the parasitic latch behaviour (which may be achieved by ensuring that the output node switches between 1 and 0 for every change of input). It must also detect possible stuck-on transistors and pinholes (which may be achieved only if the wired-AND nature of the output node is correctly modelled).

The following algorithms were used to generate the different test vector sets (described using a pseudo-high level language):

## a. Functional tests

Test 1

```
All inputs := 0

For n = 0 to 15
    input n := 1
    select input n
    check output = 1

For n = 0 to 15
    input n := 0
    select input n
    check output = 0
END
```

Test 2

```
All inputs := 0

For n = 0 to 15
    select input n

    check output = 0
    input n := 1
    check output = 1
END
```

Test 2 should uncover any parasitic latch behaviour since the output node toggles with each test input; test 1 may not, even though it will uncover primary input and output stuck faults.

## b. Gate-level test

The circuit was modelled as 16 5-input AND gates driving a 16-input OR gate. The following test algorithm covers all stuck faults on these gates. (Any input not assigned in the following description is set to 0).

Test 3

Select input 0, input 0 := 1, check output = 1;
inputs 1, 2, 4, 8 := 1 (input 0 := 0),
check output = 0.

Select input 1, i/p 1 := 1, check o/p = 1;
i/p's 0,3,5,9 := 1, check o/p = 0

Select input 2, i/p 2:= 1, check 1; i/p's 0,3,6,10
:= 1, check 0.
3 i/p 3: = 1, check 1; i/p's 1,2,7,11
:= 1, check 0.
4 4 check 1; 0,5,6,12
check 0.
5 5 ; 1,4,7,13
6 6 ; 2,4,7,14
7 7 ; 3,5,6,15
8 8 ; 0,9,10,12
9 9 ; 1,8,11,13
10 10 ; 2,8,11,14
11 11 ; 3,9,10,15
12 12 ; 4,8,13,14
13 13 ; 5,9,12,15
14 14 ; 6,10,12,15

Select input 15, i/p 15: = 1, check 1; i/p's 7,11,13,14
:= 1, check 0.

This test algorithm will detect parasitic latches provided the output toggles with every test input. However, for the experiment, the tests were ordered such that the output was at 1 for the first 16 test inputs, and at 0 for the second 16 test inputs.

### c. Switch-level test generated by HITEST

By modelling each pass transistor as a uni-directional tri-state buffer element, HITEST generated a test set described by the following algorithm (taking into account the presence of the inverters on all the inputs and the output).

```
Test 4
All inputs: = 0
For n = 0 to 15
        input n: = 1
        select input n
        check output = 1
        input n: = 0
        check output = 0
    Next n
END.
```

### d. Switch-level test generated by hand

The following test algorithm was generated to cover all the fault-effects predicted by SPICE. It is the inverse of the automatically generated test in that each input is set in turn to 0 (instead of 1 in the HITEST test) whilst all the other inputs are held at 1 (0 in the HITEST test).

Test 5

```
All inputs: = 1
For n: = 0 to 15
    input n: = 0
    select input n
    check output = 0
    input n:= 1
    check output = 1
Next n
END.
```

These five test sets were each applied to all the chips using a Datatron LSI-400 automatic test system, and the following results were obtained:

| | | | |
|---|---|---|---|
| Total chips tested | 529 | | |
| Fault-free chips<br>(passing all texts) | 168* | 31.7% | |
| Faulty chips<br>(failing all tests) | 305 | 57.7% | |
| - stuck-at output | 139 | | 26.3% |
| - other | 166 | | 31.4% |
| Faulty chips erroneously<br>passed by test 1 | 47 | 8.9% | |
| Faulty chips erroneously<br>passed by test 4 | 9 | 1.7% | |
| | | 100.0% | |

*Note: Due to a photolithography error on the mask, the inverter on input 16 had a stuck-at-0 output fault on every chip. Test vectors relating to input 16 were thus omitted from the five test sets.

Chips with gross defects observable under an optical microscope (scratched, breaks in power supply lines etc) were not tested.

These results show that the ordering of the test inputs for NMOS pass transistor networks is very important. Test sets 1 and 2 consist of the same test vectors but in different orders. As a consequence, nearly 9% of the chips tested passed test 1 erroneously but correctly failed test 2. It was most interesting to find that the Boolean gate-level test (set 3), which was not ordered to uncover the parasitic latch problem, nonetheless failed all the chips that had passed test 1 and failed test 2. The most likely reason for this is the chips that erroneously passed test 1 had pinholes in an inverter on the control lines, so that both control lines (X and X') were always the same. Under these conditions, either two inputs or no inputs would be selected by the multiplexor. These chips would pass test 1 when two inputs at opposite logic values were being selected if the output should have been 0, due to the wired-AND output. In test 3, however, should two opposite inputs be selected, the 16-input OR-gate assumed in the gate-level representation exactly models the wired-AND output with inverted inputs and outputs. Thus logic conflicts introduced by faults were fortuitously modelled correctly with the conflict being resolved in favour of a 0 at the output.

To confirm this point, the gate-level test set was then re-written by inverting all the input vectors (thus effectively re-modelling the output node as a wired - OR node), and re-applied to the chips on one of the wafers. As expected, the re-written test erroneously passed all the chips that had erroneously passed test 1 and failed test 2, as well as passing 5 chips that had failed all the other tests. It also passed chips that had been erroneously passed by test 4. Thus, the re-written stuck-at test proved to be the least effective of the tests, and showed that NMOS circuit behaviour in the presence of faults cannot in general be modelled accurately at the Boolean primitive level of extraction. In this case, an NMOS circuit was described by two logically identical Boolean gate-level diagram; however, whilst the test set generated from one of these diagrams successfully identified all the faulty chips, the other test set identified 12% of the faulty chips as being fault-free. Thus, stuck-at motivated testing is likely to have only limited success in correctly identifying faulty NMOS circuits.

It was not surprising that the automatically generated sequence passed a few chips failed by the other tests because the fault model used by HITEST tacitly assumed a wired-OR output node. Thus, HITEST would detect a fault arising from a logic conflict if the output should have been a 0, instead of detecting a logic conflict fault if the output should have been a 1, as predicted by SPICE. However, HITEST did correctly sequence the tests to uncover parasitic latches.

In addition to testing the chips with the five sequences as decribed above, the chips on one wafer were all tested at different power supply voltages. No chip worked at $V_{DD}$ = 3.5 V, and no chip which failed at $V_{DD}$ = 5 V worked at a lower voltage. Most of the remaining chips worked at $V_{DD}$ = 5 V, 4.5 V and 4 V, although 4 chips only worked at $V_{DD}$ = 5 V, and 16 chips only worked at $V_{DD}$ = 5 V and 4.5 V. These chips had a lower than average $V_{OH}$, suggesting a "marginal" fault in the output pad driver network. This suggests that lowering $V_{DD}$ would be a particularly useful technique for testing the output driver stage of a chip.

Overall, this chapter has examined the relationship between physical faults in NMOS circuits and their related logical fault-effects. The stuck-at model has been shown to be inadequate for testing NMOS circuits for the following reasons:

* the gate-level description(s) of an NMOS logic circuit and the layout of the circuit are not topologically equivalent, resulting, in the worst case, in the generation of tests for nodes that do not exist or the non-generation of tests for nodes that do;

* the faults that occur in NMOS integrated circuits tend to produce short or open circuits rather than stuck nodes;

* MOS transistors are bi-directional - a characteristic that is not modelled on an equivalent gate-level representation. This property can produce fault-effects which are not predicted (and therefore not covered) by the stuck-at model;

* MOS logic gate inputs present purely capacitative loads. In the presence of certain faults, output nodes can act as dynamic (stored-charge), parasitic latches. This phenomenon is usually associated with CMOS circuits

(under stuck-open conditions), but can equally well occur in pass transistor networks implemented in NMOS.

Having thus found that the stuck-at fault model is inadequate for testing NMOS circuits, a more realistic fault model is required as a replacement. In the next chapter a new fault model for NMOS circuits is described which better reflects the fault commonly found in NMOS circuits. In common with the stuck-at fault model, the new fault model is mathematically simple and lends itself to efficient and convenient implementation in software.

## REFERENCES

1 I Masuda, M Ueno and K Tashiro (1983).
A fault-tolerant MOS-LSI for train controller applications.
IEEE ISSCC digest of technical papers, pp 138-139.

2 R Y Li, S C Diehl and S Harrison (1983).
Power supply noise testing of VLSI chips.
Proc 1983 IEEE Test Conference, pp 366-369.

3 A Hunger and A Gaertner (1984).
Functional characterisation of microprocessors.
Proc 1984 IEEE Test Conference, pp 794-803.

4 M Campbell (1984).
Monitored burn-in (a case-study for in-situ testing and reliability studies).
Proc 1984 IEEE Test Conference, pp 518-523.

5 E R Hnatek (1984).
Thoughts on VLSI burn-in.
Proc 1984 IEEE Test Conference, pp 531-534.

6 C Mead and L Conway (1980).
Introduction to VLSI systems.
Addison-Wesley, Reading, Mass.

7 D J Wharton (1983).
The HITEST test generation system - overview.
Proc 1983 IEEE Test Conference, pp 302-310.

8 H H Chen (1984).

Test Generation for MOS Circuits.

Proc 1984 IEEE Test Conference, pp 70-78.

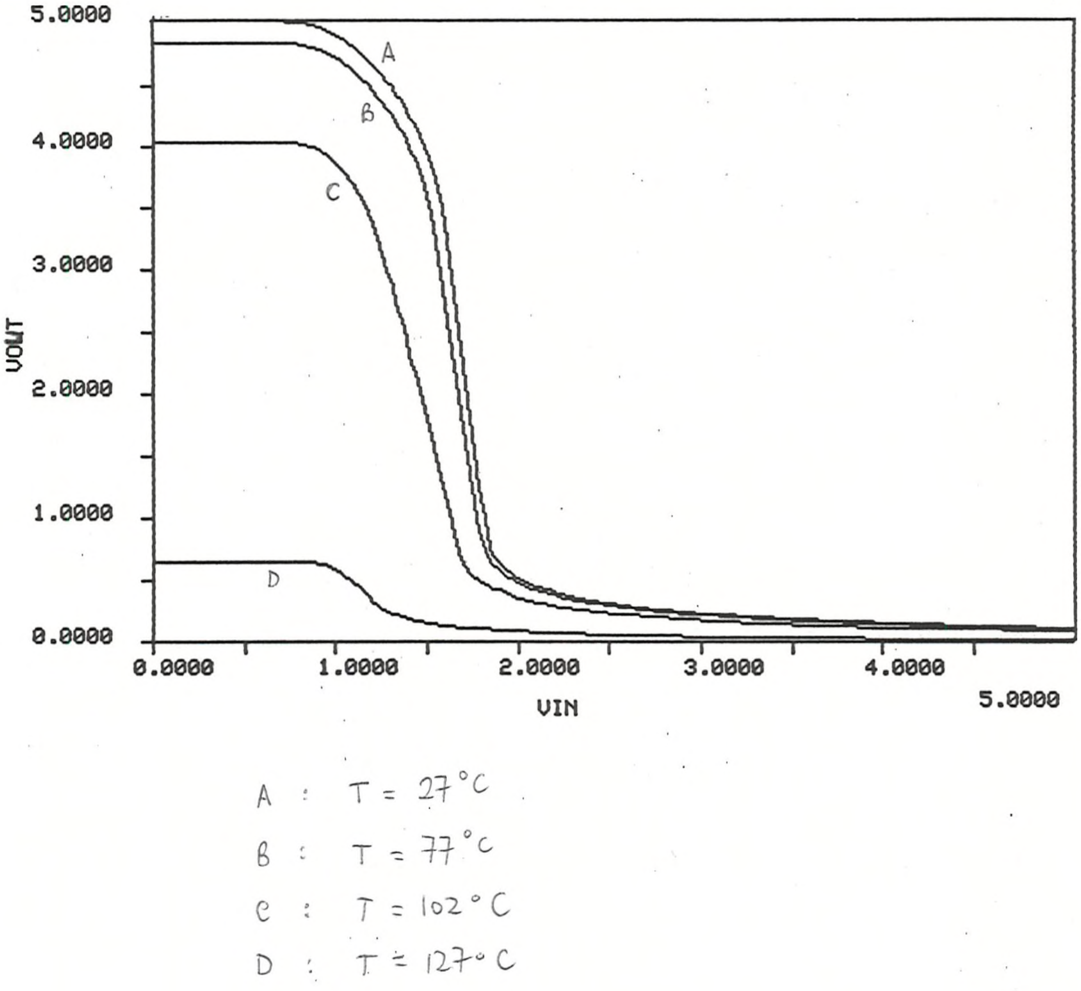
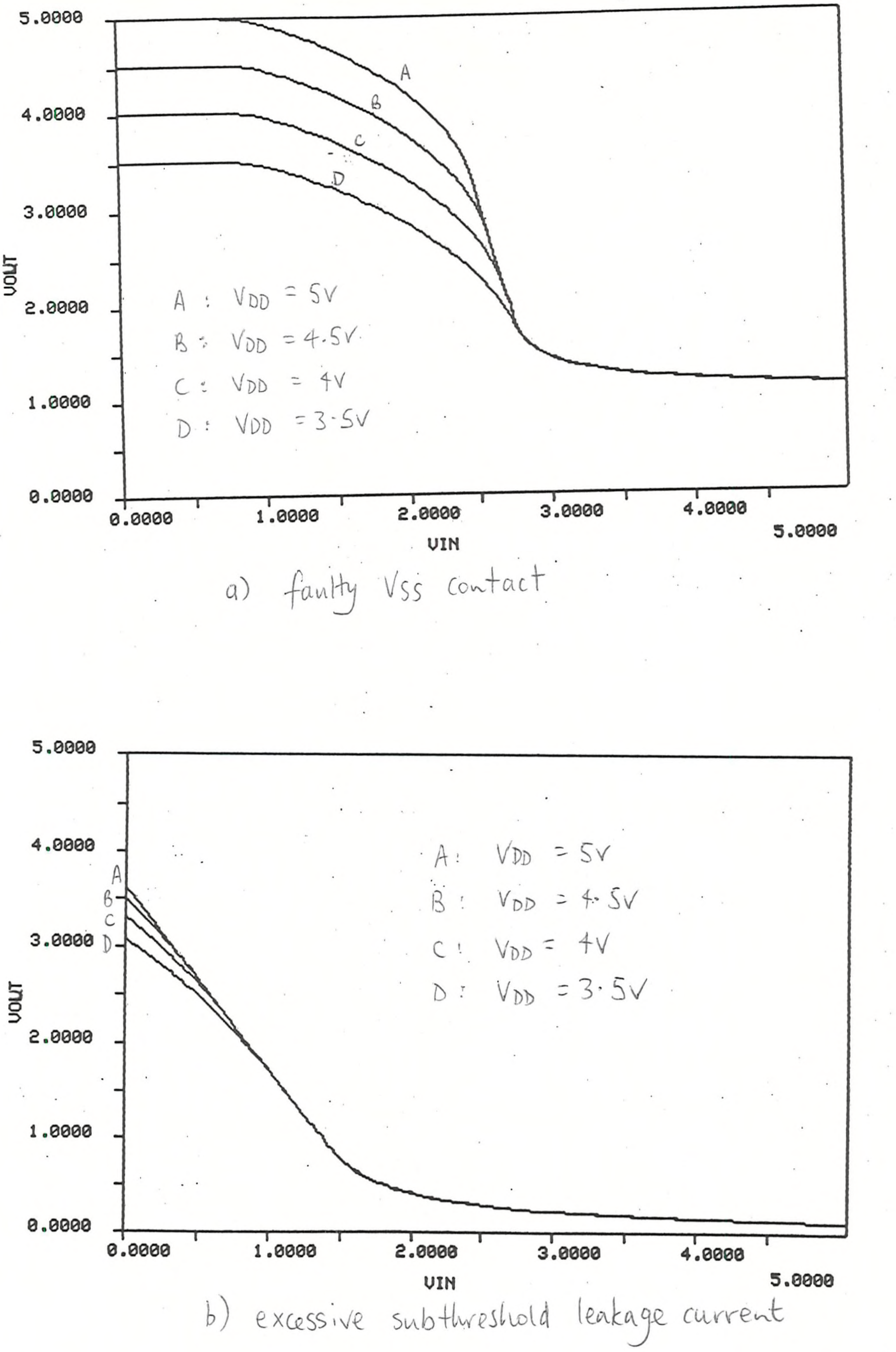


Figure 4.1 SPICE simulations of inverter with marginal fault at different temperatures



a) faulty Vss contact

b) excessive subthreshold leakage current

Figure 4.2 SPICE simulations of inverters with marginal faults at different supply voltages

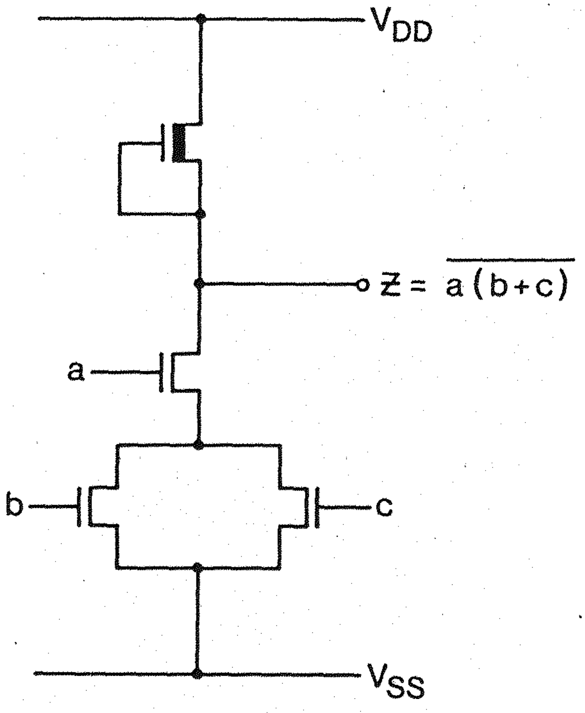


Figure 4.3 Example of an NMOS circuit whose faults may all be modelled as stuck nodes

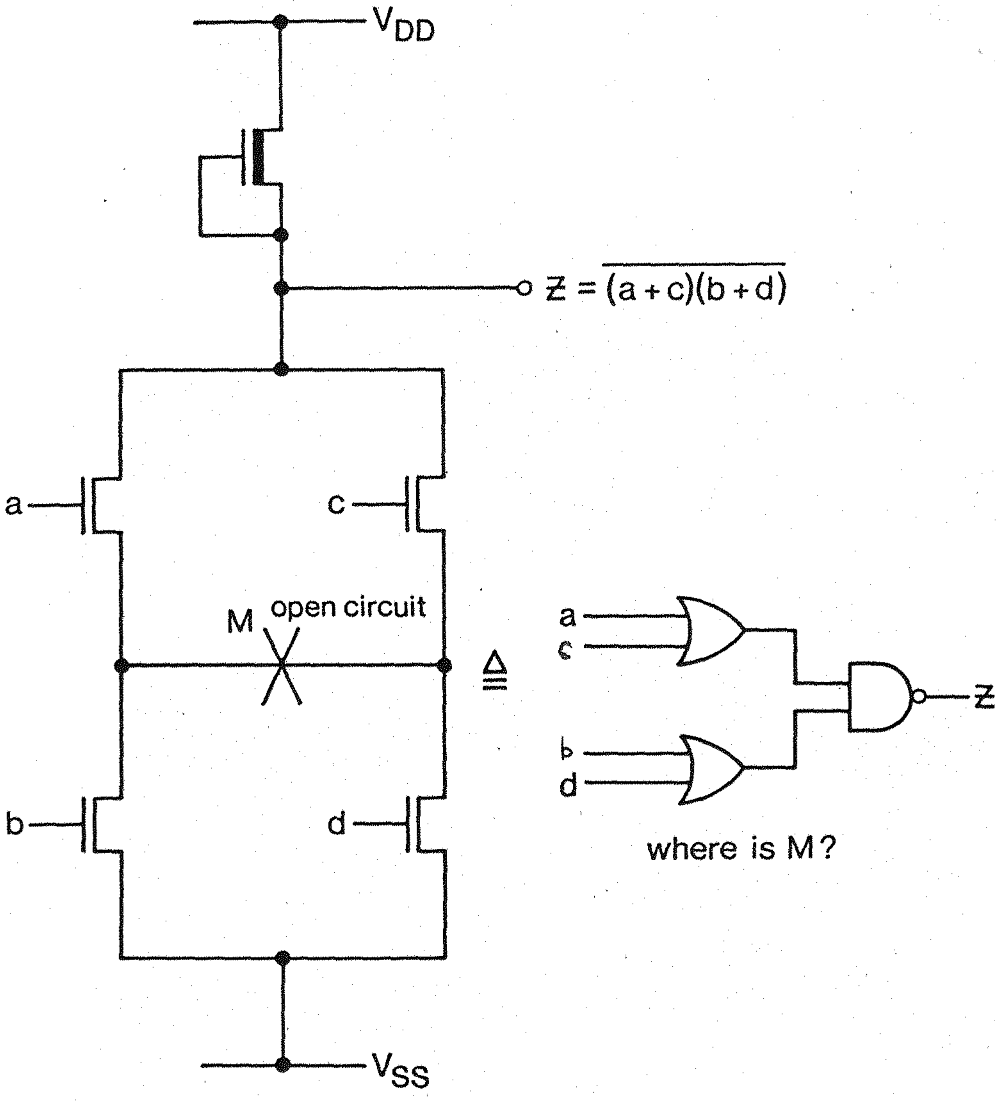
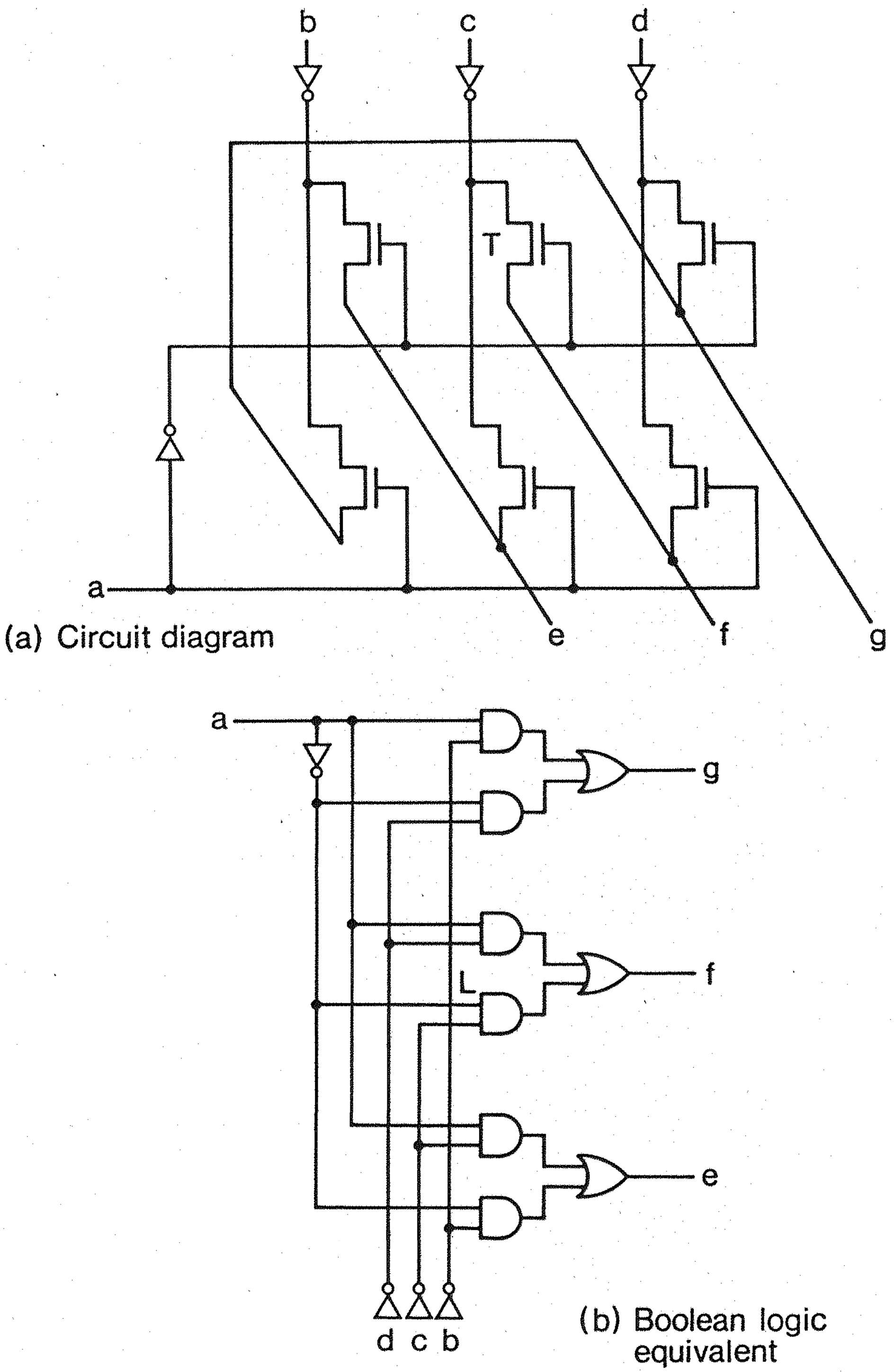


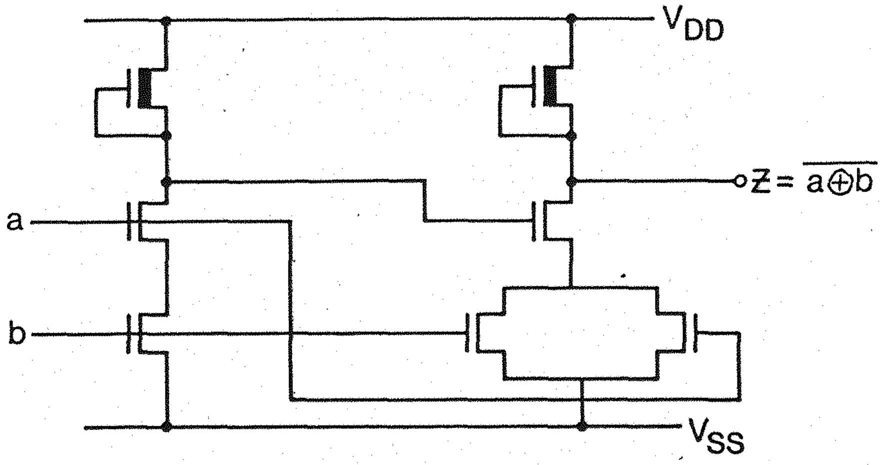
Figure 4.4 Example of an NMOS circuit containing a fault that does not produce a stuck-node fault effect



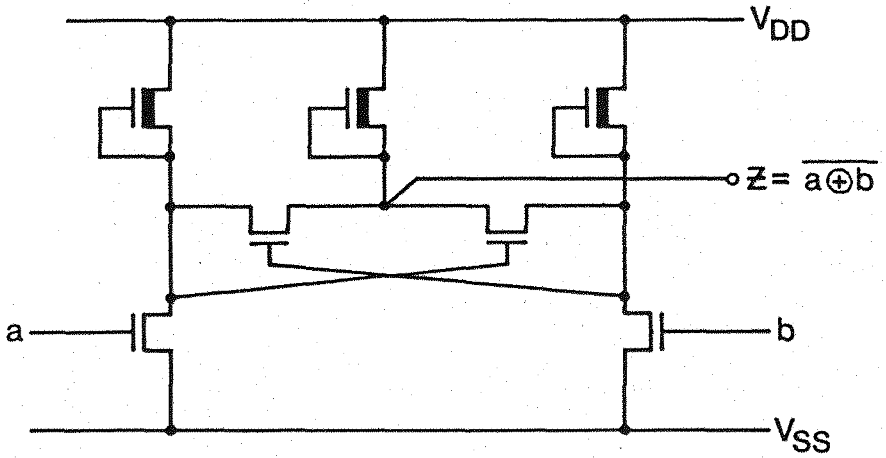
(a) Circuit diagram

(b) Boolean logic equivalent

Figure 4.6 NMOS rotator circuit

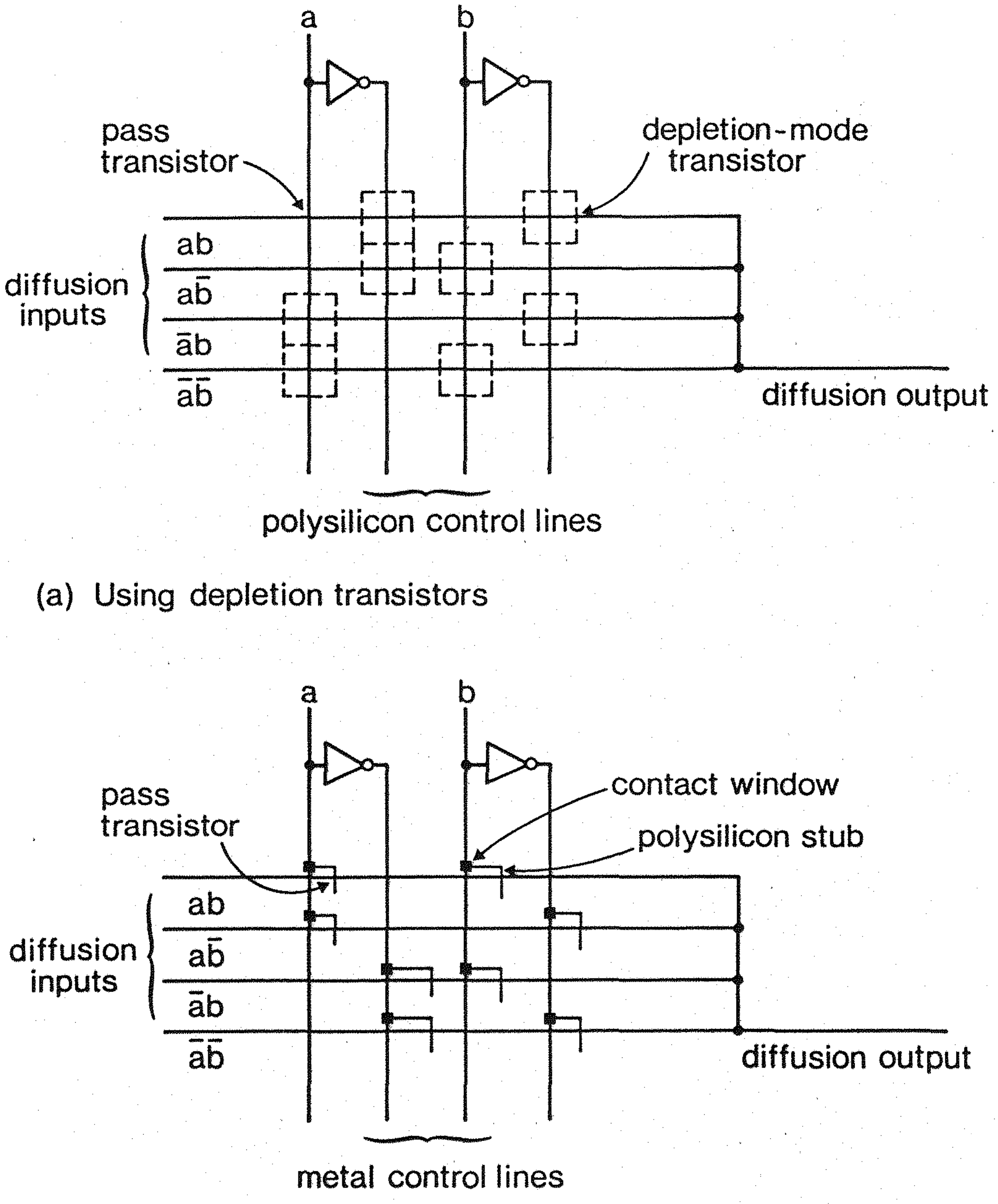


(a)



(b)

Figure 4.5 NMOS implementations of the exclusive-nor function



(a) Using depletion transistors

(b) Using polysilicon stubs

Figure 4.7 Function block implementations of a 4-to-1 multiplexor

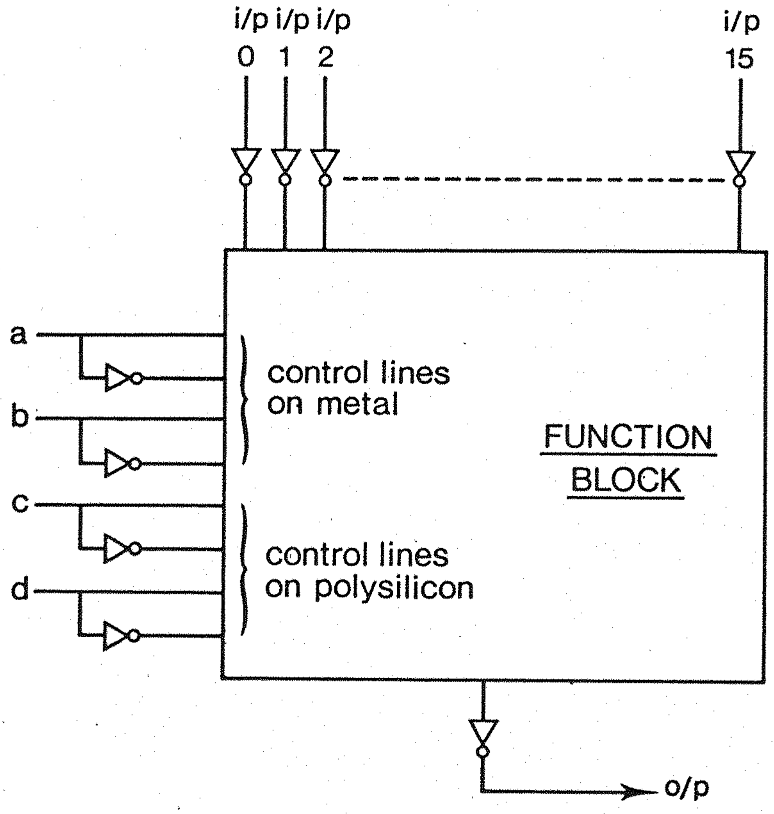


Figure 4.8 Block diagram of the 16-to-1 multiplexor test circuit

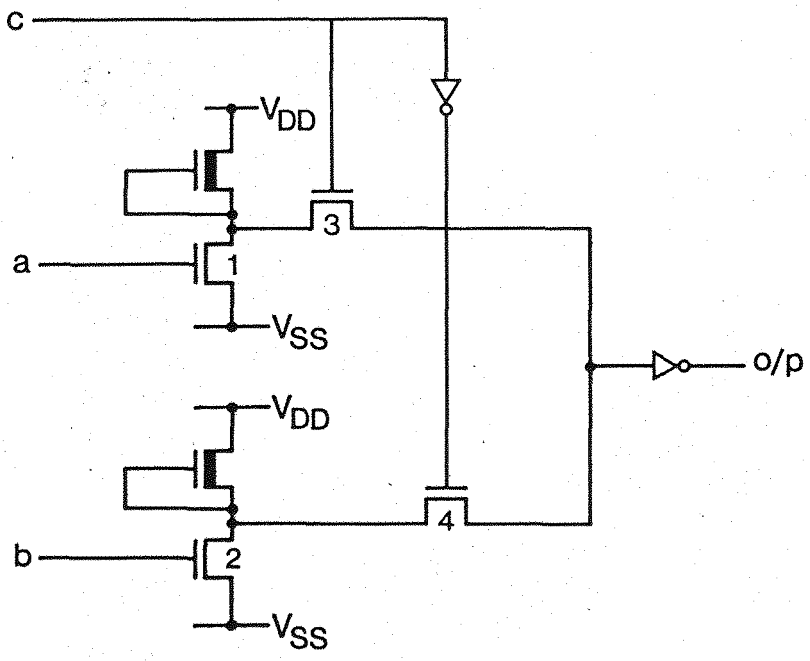


Figure 4.9   A 2-to-1 multiplexor implemented in NMOS

# CHAPTER 5 A NEW FAULT MODEL FOR NMOS CIRCUITS

## 5.1 Introduction

In the previous chapter the stuck-at fault model was shown to be inadequate for testing NMOS circuits for a number of reasons. A better approach was found to be to consider faults at the transistor (or "switch") level, as this would retain the structural information about the circuit that is lost by using the gate-level representation (ref 1). However, it would be impractical to consider stuck-at node faults at the lower level of abstraction, because of the large increase in the number of circuit nodes to be analysed, with a correspondingly large increase in computation time needed to generate and evaluate the test input vectors. In view of this difficulty, it seemed preferable to discard the stuck-at model entirely and work with more realistic fault assumptions.

Thus, a better fault model for digital MOS circuits would operate at the switch level, but would consider transistor stuck-open and stuck-on faults as well as open and short circuits in interconnections (refs 2, 3). These faults lead either to conducting paths (eg between $V_{DD}$ or $V_{SS}$ and a gate output node, or between input and output nodes of a pass transistor network) which should not exist under fualt-free conditions, or to the absence of paths which should exist. Test patterns ought, therefore, to be selected to detect the fault-induced presence or absence of such paths. In practice, these test patterns should be derived algorithmically, and so a mathematical framework for this activity is required.

There is a well-developed branch of algebra, concerned with the extraction of path information from digraphs, which is relevant to this problem. By describing the MOS circuit in graph form, with each transistor or interconnection of interest forming one arc of the graph, the algebraic methods - so-called path algebras (ref 4) - can be directly applied. A path algebra is defined as a set (having a zero and a unit element) equipped with two operators, dot (.) and join (v), where the exact nature of the dot and join operators is determined by the path problem that the algebra is designed to solve. Two algebras are employed here to yield sets of arcs describing paths or cuts in the circuit under specified input conditions. As a property of the algebras, each set of arcs corresponds directly to a test vector, with high physical fault coverage resulting from the use of a realistic fault model.

## 5.2 Path Algebras and MOS Circuits

MOS logic circuit operation is conveniently understood by considering current flow from the positive power supply rail along current paths in the circuit, either to the negative rail or onto the input of another logic circuit. Transistors are envisaged as switches which conduct or not as a function of their gate-source potential difference, thus determining which current paths exist under given input conditions.

These paths can be readily identified by applying the first of two path algebras, where the resulting arc-set has the property that if any one arc is removed, the path becomes an open circuit (so-called elementary paths - see below). Thus each path may be used to form an input test vector which tests for all open circuits along that current path. The algebra generates all such paths.

In a similar fashion, the second path algebra is used to determine sets of arcs that minimally cut all the current paths, where a minimal cut-set is a set of arcs that cuts all the paths between two specified nodes such that if any one arc is removed from the cut-set, the remaining arcs do not form a cut-set. Each cut-set may be used to form an input test vector which tests for all specified short circuits and stuck-on transistors in the cut-set, and the algebra generates all such cut-sets. These sets of paths and cut-sets define the "singular cover" of an MOS gate (with load transistor), or the "minimum" conditions for charge storage at, or signal propagation to, an output of a pass transistor network. Faults may be propagated through a gate if the faulty input appears in both a path set and a cut-set within an input vector, and through a pass transistor network simply by setting up a path through it. The notations and methods used in this path algebraic approach are now developed, with extensive use of examples to facilitate understanding.

A network of transistors connected together to form a circuit may be modelled as a graph whose nodes are numbered from 1 to n and whose arcs, which form the interconnections between the nodes, are identified by labels which are typically single letters. An (n x n) matrix, A, called the adjacency matrix describes the graph in a convenient form. It consists of elements $a_{ij}$ that take the label of the arc that connects node i to node j if one exists, or 0, the null element, otherwise. If an arc is bi-directional, then $a_{ij} = a_{ji}$. If the arc is uni-directional, then one of $a_{ij}$ or $a_{ji} = 0$. An example of a transistor network, its graph, and the adjacency matrix of the graph is shown in Figure 5.1.

If the adjacency matrix is "multiplied" (see below for definition of path-algebraic operators) by itself m times, the resulting matrix, $A^m$, contains elements $a_{ij}{}^m$, that describe the paths m arcs long connecting node i to node j. These paths are said to be "of order m", and all possible paths of order m or less in a graph may be found by forming the union of the successive powers of the adjacency matrix ($A$, $A^2$, $A^3$, ...$A^m$). Figure 5.2 shows $A^2$ and $A^3$ for the graph of Figure 5.1. In this particular example, where two matrix elements would be multiplied under conventional algebraic rules, they are concatenated; and where they would normally be added, their union is formed.

In path algebra, two operators only are defined: DOT (denoted ".") and JOIN ("v"). In the above example, conventional multiplication is replaced by the DOT operation, and conventional addition by JOIN, where DOT is defined as concatenation, and JOIN as union. However, other useful path algebras exist for which DOT and JOIN are differently interpreted. Furthermore, for all path algebras, there exist both a zero element, 0, defined such that $a_{ij} . 0 = 0$, and $a_{ij} \vee 0 = a_{ij}$, and a unit element, U, defined such that $U . a_{ij} = a_{ij}$, and $U \vee a_{ij} = U$. The unit element is thought of as a connection between two nodes that may not form part of any arc-set.

An elementary path is defined as a path (or set of arcs) which visits no node more than once, and is analogous to the concept of a tie-set used in impedance network analysis to determine chains of impedances in the network. The DOT and JOIN operators are defined for the algebra that determines all elementary paths as follows:

DOT: $A . B = \{a \text{ concatenate } b\}$ for all a and b, where a is a member of the set of arc-sets A, and b a member of the set of arc-sets B.

JOIN: $A \vee B = r\{A \text{ union } B\}$ where r means reduce the set such that if a path exists whose arcs form a subset of another path in the set, the longer path is removed from the set.

If this algebra is used on a graph, every elementary cycle, $a_{i=j}$, is set to 0, as no path may visit a node more than once. This means no path may be more than (n-1) arcs long, where n is the number of nodes in the graph, which in turn implies that in order to find all the elementary path arc-sets in a graph the adjacency matrix only need to "multiplied" by itself (n-1) times.

A somewhat different algebra is required to determine all possible cut-sets: in this case, DOT and JOIN operators are conversely defined as:

DOT: $A \; . \; B = r\left\{A \text{ union } B\right\}$ .

JOIN: $A \vee B = r\left\{a \text{ concatenate } b\right\}$ for all a and b.

Figure 5.3 shows these two algebras being used to determine the elementary path arc-sets and the cut-sets for the graph shown in Figure 5.1. The resulting matrix in both cases has been formed by JOIN-ing A, $A^2$, and $A^3$ where the successive powers of the adjacency matrix have been determined using the different definitions of DOT and JOIN for each algebra. Note that in the elementary path arc-set matrix, all elements on the leading diagonal have been set to 0, because no elementary path can start and finish at the same node, by definition. Similarly, any path containing a repeated arc label (eg aab) has been set to 0 for the same reason.

A most important property, contributing to the attraction of this approach, is that the adjacency matrix can be manipulated in such a way that its successive powers need not be computed. In essence, the matrix is transformed into upper triangular form using Gaussian elimination (see reference 4 for details), so that all elements $a_{i \leqslant j}$ are set to 0 and all the remaining elements are operated on to retain all the path information. A backward substitution method is then performed on this upper triangular form of the adjacency matrix to determine the elementary path arc-sets and the cut-sets. This substitution method finds an arc-set between node i and node j by successively finding "partial" paths from node (j-n) to node j. Initially, n is set equal to 1 to find all paths from node (j-1) to node j. Next, n is set equal to 2 and all paths from (j-2) to both (j-1) and to j are found. The paths from (j-2) to (j-1) are DOT-ted with those already found from (j-1) to j, and thus all paths from node (j-2) to j are found. This substitution process is repeated until (j-n) = i, at which point all paths from node i to node j have been found.

An alternative to the method based on Gaussian elimination is the Yen double-sweep technique, again described by Carré (ref 4). In this technique, backward and forward substitutions are made alternately without first transforming the adjacency matrix, where a forward substitution is defined as the opposite of a backward transformation. For an n x n matrix, (n-1) substitutions are needed at most (counting forward and backward substitutions separately). The Gaussian elimination method is preferred when dealing with circuits having both multiple inputs and multiple outputs. On the other hand, the double-sweep method is preferred for analysing circuits with either a single input or a single output.

## 5.3 Test Pattern Generation for NMOS Circuits

The path algebraic techniques described above generate the singular cover for a single MOS gate, which may be used to generate input test vectors and fault propagating input vectors. A framework is needed to enable test patterns to be generated for a circuit made up of many MOS gates and pass transistor networks connected together in a combinational block. The D-Algorithm (see Section 1.4 and ref 5) performs precisely this operation in the context of stuck-at, gate-level testing by determining and activating "sensitive paths" in a combinational logic circuit, along which a fault-effect may then be propagated. The reasons for using the D-Algorithm are that it is already widely-used, and is acknowledged as being the best available (if not optimum) method for generating test vectors for a given fault in a circuit, and as such is regarded as being unlikely to be radically improved on. I shall show how the D-Algorithm may be used with the path algebras rather than the stuck-at fault model to utilise the more realistic yet equally simple fault model for MOS circuits.

In section 1.4, the D-Algorithm was described as having three distinct parts: fault insertion, fault propagation (the "D-drive"), and the consistency check. The consistency check may be taken without alteration, using the singular cover derived by the path algebras. The fault insertion may also be done using the singular cover. A path-set (cut-set) is chosen as an input vector to an NMOS gate, and by setting the designated gate inputs to a 1 (0), a path (cut) is defined in the gate which under fault-free condition would force the gate output to a 0 (1). This covers any stuck-open (stuck-on) transistor or open (short) circuit in the path-set (cut-set). In the presence of any such fault the output is forced to an incorrect value, as represented by D'(D). The remaining gate inputs should be set to 0 (1) to ensure an erroneous output in the presence of a fault. In this way, the stuck-at fault model is replaced by the far more realistic "short circuit/open circuit" fault model for MOS transistors and interconnects.

In the fault propagation phase, the D-Algorithm uses pre-defined propagating and non-propagating D-cubes for the different Boolean gate-types to drive a fault-effect to a primary output. Faults are propagated through a gate by setting unassigned inputs to 1 or 0 as appropriate, such that all inputs at D (or D') are included in both a path-set and a cut-set across the gate inputs. For example, figure 5.4 shows an NMOS gate and its singular cover (paths and cuts). Suppose that in the course of the D-drive, inputs a and c had been set to D. By setting one of the "don't care" inputs (b or d) to 1, both inputs already at D become arcs on paths activated in the gate (ab and cb, or ad and cd). Inputs a and c together form a cut-set and so no input need be set to 0 to make up a cut-set containing these inputs. This gives the propagating D-cube abcd = D1DX or DXD1 with the output set to D'. Thus if inputs a and c are at 1 (fault-free condition), the paths are activated and the output pulled down to 0; if inputs a and c are at 0 (fault-induced condition), the cut-set is defined in the gate and the output is forced to 1. If no such set of input conditions can be established without setting any input simultaneously to 1 and 0, then no propagating D-cube exists for the given input conditions, and the algorithm must back-track to its last decision point and alter one of the gate inputs, or try propagating the fault through a different gate. If a gate has a D and a D' simultaneously on its inputs, again the algorithm must back-track and choose a different option. As stated before, the D-Algorithm's handling of such decision points and of back-tracking contributes substantially to its wide-spread acceptance and success.

Non-propagating D-cubes are taken directly from the singular cover, since each arc-set specifies the gate inputs that need to be set to a 1(0) to force a 0(1) on the gate output. The non-propagating cube is formed by noting which gate inputs have already been set to D (or D') and picking an arc-set that does not conflict with any other inputs in the gate (most of which will be in a "don't care" state in any case). For example, returning to the circuit of figure 5.4, suppose that as a result of the D-drive the state abcd = DXOX had been reached. The only available non-propagating D-cube is DOOO; all the other arc-sets either contain input a, or else require input c to be a 1.

The piece of interconnect, w, is also explicitly tested for in this approach since it is treated as a transistor whose input is always at 1. In the presence of a fault, the "interconnect transistor" is effectively switched off. In a similar manner, potential short circuits are treated as transistors, which under fault-free conditions are switched off, and which under fault conditions are turned on. Thus, pieces of interconnect (susceptible to open circuits) only appear in path-sets, and conversely, potential short circuits, which are easily inserted into the adjacency matrix, only appear in cut-sets. These interconnect faults are often impossible to specify on a gate-level description of a circuit (section 4.3).

Before the paths and cut-sets can be determined, certain transformations on the adjacency matrix are necessary. For the path-set generation, any potential short circuit is replaced by the null element, 0, to ensure that no tests are generated using a path that does not exist in the fault-free circuit. Similarly, for the cut-set generation, any interconnect must be replaced by the unit element, U, to ensure that no test is generated that requires the interconnect to be switched off, as this is clearly an impossibility. In addition, all arcs that are connected to either power rail (via a load transistor where relevant), or to an output node, are made uni-directional to avoid determining current paths that traverse these nodes. These ensures that any paths that do visit any of these nodes will start or finish there. (In static NMOS, the output node and the node connected to VDD via the load depletion transistor are one and the same).

If two nodes are connected by more than one arc in parallel, those arc labels are JOIN-ed with each other, using the appropriate algebra, before proceeding. Similarly, if two nodes are connected by more than one arc in series, those arc labels are DOT-ted with each other using the appropriate algebra. This ensures that multiple arcs between any two nodes are correctly described under the algebra being used. For example, if two parallel paths exist between node i and node j, then under both algebras, the correct arc-set (elementary path or cut-set) between those two nodes is obtained simply by JOIN-ing the two arc labels. Figure 5.5 shows an example of these manipulations for a simple representative circuit.

Sometimes, an MOS gate will have internal fan-out, whereby separate transistors are controlled by the same input. Circuits like this are tested by entering each transistor label separately in the adjacency matrix but with a numbered subscript for transistors with the same input. Test vectors can then be generated in the usual way. If a test for a specific transistor cannot be generated, the transistor is redundant, and produces no observable fault-effect if it is faulty. To prevent redundant transistors from affecting fault detection, they should be switched off during test. Figure 5.6 gives an example of test generation for a circuit with internal fan-out (inputs a, c and d) and redundancy. The procedure is as follows:

i. If more than one transistor is controlled by a logic input X, re-label those transistors $X_1$, $X_2$, .... $X_n$;

ii. Generate arc-sets (elementary path sets and cut-sets) in the usual way using the new labels;

iii. Reduce the arc-sets treating $X_1$, $X_2$, .... $X_n$ as the same label.

As stated above, a transistor not present in the final set of test inputs is redundant: in the example, transistor e is logically redundant.

The existence of pass transistor networks in MOS circuits can lead to parasitic latch behaviour under fault conditions. Such behaviour can only be detected if test inputs are carefully ordered in view of the sequential nature of this fault. The path-algebraic method, however, gives a powerful way to effect the required ordering and thus achieve fault insertion. Figure 5.7 illustrates how this may be done, in essence, by applying input vectors from the path-sets and cut-sets alternately. First, a test input is applied to the driver network to force node X to a specified value. This value is propagated through the pass transistor network and onto the charge storage node, Z, by simultaneously applying an open circuit test input to the pass transistor network (which defines an elementary path through the network). Next, the inputs to both the driver network and the pass transistor network are changed by applying test inputs that are complementary in type to the first applied inputs. In this example, the new input to the pass transistor network is a short circuit test input, and if the charge storage node now changes value, there must be a short circuit in the pass transistor network. After this, a second open circuit test input is applied to the pass transistor network, and if the output node does not now change, there is an open circuit present in the pass transistor network, or a fault in the driver network. This process is repeated until all faults in the driver and pass transistor networks have been covered, and the method may be readily extended to test pass transistor networks with multiple inputs or outputs.

## 5.4 Previous Work

Papers describing methods for testing MOS logic circuits at the switch-level have appeared before, some even using graph theory, but they have shortcomings which are avoided by the present method. For example, El-Ziq and Su (ref 6) use a matrix-manipulation method to derive tests, but do not model the MOSFET's bi-directionality and do not guarantee to generate tests for interconnect faults. Chiang and Vranesic (ref 7) use a path-oriented approach which explicitly models interconnection faults and MOSFET bi-directionality. However, they do not use path algebras: the method followed is capable of finding cut-sets only. Thus, to generate open-circuit tests, they are forced to use the cut-set of the dual of the digraph representation. This approach is restricted to circuits having planar (two-dimensional) graphs, since only such graphs possess duals. Multi-level interconnect technology, however, makes it perfectly possible to implement circuits having non-planar digraphs. Even more seriously, it is extremely difficult to determine computationally the dual of a graph by manipulation of its matrix description.

More recently, Roth et al (ref 8) have described a "structural mapping" of an MOS logic circuit onto a (logically equivalent) gate-level description of the circuit designed to retain important switch-fault information. They assert that tests generated to cover stuck-at faults in the "image" logic circuit also constitute tests for transistor stuck-on and stuck-open faults in the physical circuit. The attraction of the method is that well-developed automatic test pattern generation procedures for stuck-at testing (eg the D-algorithm) can continue to be used. Unfortunately, however, interconnection faults are not considered during test generation. Section 4.3 of this thesis shows that use of Boolean logic primitives to model an MOS logic circuit necessarily leads to loss of information about the topology of the layout and about MOSFET characteristics that cause the stuck-at fault model to fail.

Jain and Agrawal (ref 9) suggest the use of special functional blocks which are sequential rather than combinational in nature, to model explicitly faulty MOS behaviour (logic clashes, charge storage phenomena). In this way, the MOS circuit may be modelled at the gate level thus again permitting well-developed automatic test pattern generation procedures to be used. This gives rise to a large increase in the number of circuit elements to be analysed and an increase in simulation complexity as a result of the introduction of the special sequential functional blocks. For example, a circuit consisting of a NOR gate, an inverter and a pass transistor is modelled by four Boolean logic gates and three sequential functional blocks, Thus, accurate fault modelling is achieved but at the cost of significantly longer run-times and increased effort in the circuit description.

## 5.5 An example of path algebra-based test pattern generation

In this section, an example of path-orientated test pattern generation is given, together with some discussion on the algorithm's complexity and software implementation.

Figure 5.8 shows a possible NMOS implementation of a 3-to-1 multiplexer using two NOR gates, one inverter and a 5-input complex gate. The singular covers of the different gates are also shown. As discussed earlier, the procedure for test generation using the D-Algorithm is

i. insert a fault;

ii. drive the fault to a primary output;

iii. check the fixed values introduced for consistency, and hence derive a test input.

Thus, to test node 7 stuck-at-0 (corresponding to transistor b in the upper NOR gate open circuit) a path set including this node is selected from the singular cover. The input(s) in the selected path set are all set to D, and any other inputs to 0. The output node is set to D'. Since the D in this instance refers to a fault within the gate-under-test, the fault may be regarded legitimately as being confined to that particular fan-out branch, without propagating back to the fan-out trunk. Thus, in the example, node 7 is a D but node 2 (and therefore node 9) is a logical 1. The resulting test cube is:

$$(1_2 \quad 0_3 \quad D_7 \; 1_9 \; D'_{10})$$

To propagate the D' through G4, node 10 must be part of both a path set and a cut set across the complex gate's inputs. Node 10 appears in the path set $\{10, 11\}$ and in the cut-sets $\{9, 10, 12\}$ and $\{6, 10, 12\}$. The fault may be propagated by setting node 11 to a 1, thus defining a path set including node 10, and setting nodes 6 and 12 to 0, thus defining a cut-set simultaneously. (Node 9 has already been set to 1 and thus may not be used to form a cut-set). The output of the complex gate is now a D and the new test cube is:

$$(1_2 \; 0_3 \; 0_6 \; D_7 \; 1_9 \; D'_{10} \quad 1_{11} \; 0_{12} \; D_{13})$$

The primary output has been reached and the consistency check now begins. To force node 12 to a 0, node 5 or 8 must be at 1 to define a path through the NOR gate G3. Node 8 may not be set to 0 since node 11 is already at 1; thus node 5 is set to 1. Similar reasoning leads to node 1 being set to 1, and the final test cube is:

$$(1_1 \; 1_2 \; 0_3 \; 1_4 \; 1_5 \; 0_6 \; D_7 \; 1_8 \; 1_9 \; D'_{10} \quad 1_{11} \quad 0_{12} \quad D_{13})$$

The input $1_1\ 1_2\ 0_3\ 1_4\ 1_5$ covers nodes 7 and 13 stuck-at-0, and node 10 stuck-at-1. This process of fault insertion, D-drive, and consistency check is repeated until all the faults have been covered. The above example illustrates well some of the differences between path-orientated testing at the switch level of representation, and stuck node testing at gate level. In the path testing method, faults are not associated with circuit nodes that connect logic primitives, but with the primitives themselves. This permits more realistic fault modelling than is possible with the stuck node approach because physical faults within each primitive may be directly modelled as a result of retaining the primitives' structural information at the switch level. Furthermore, the path testing primitives are somewhat more complex than their corresponding Boolean primitives - hence, less primitives are needed to describe the circuit-under-test. In the example, the complex gate G4 implements the function $Z' = e + d.(c + (a . b))$, which would need to be modelled by four gates on a Boolean gate-level diagram. By using a single primitive to describe the complex gate, the number of nodes in the circuit has been reduced, but at the expense of having to determine the singular cover of the complex gate.

A program to determine the singular cover of an NMOS complex gate has been written and implemented in PASCAL. The program is only 600 lines in length and is based on the Yen double-sweep method described earlier. Each arc-set (elementary path or cut-set) is represented as a PASCAL set of integers, and each set of sets is constructed as a linked list of sets. The adjacency matrix is thus implemented as an array of pointers. By using this data structure, the DOT and JOIN path algebra operators map readily onto existing PASCAL set operations (union, addition, comparison) and linked list manipulations (add to list, delete from list). Unfortunately, there was insufficient time to extend this work to include implementation of the modified D-Algorithm itself.

Goel (ref 10) has shown that test pattern generation and evaluation costs are proportional to the square of the number of logic primitives in the circuit-under-test. In the path testing method the number of logic primitives in a circuit may be reduced, with a corresponding reduction in test pattern generation and effort. However, path testing requires a larger number of primitives to be used, each of whose singular covers must be evaluated. Carré (ref 4) has shown that the determination of a path-set (or set of cut-sets) in a graph using the Yen method has a computational complexity O (number of nodes x number of arcs). NMOS logic primitives will typically consist of 4-6 arcs and a similar number of nodes, while LSI/VLSI circuits consist of several thousand logic primitives. Thus, the expense of calculating a greater number of singular covers to attain a significant improvement in real fault coverage should be offset by the saving in test pattern generation effort gained by reducing the total number of primitives and nodes needed to describe the circuit.

## REFERENCES

1 J Galiay, Y Crouzet and M Vergniault (1980) Op Cit.

2 Y El-Ziq (1983).
Classifying, testing and eliminating VLSI MOS failure modes
VLSI Design, September 1983, pp 30-35.

3 B Courtois (1981). Op Cit.

4 B A Carré (1979).
Graphs and Networks.
Oxford University Press, Oxford.

5 J P Roth (1966).
Diagnosis of automata failures: a calculus and a method.
IBM J Res Dev, 10, pp 278-281.

6 Y El-Ziq and S Y H Su (1982).
Fault diagnosis of MOS combinational networks
IEEE Trans, TC-31, pp 129-139.

7 K-W Chiang and Z G Vranisec (1982).
Test generation for MOS complex gate networks.
Proc 12th Fault-Tolerant Computing Symposium, pp 149-157.

8 P J Roth, V G Oklobdzija and J F Beetem (1984).
Test generation for FET switching circuits.
Proc 1984 Int Test Conference, pp 59-62.

9 S K Jain and V D Agrawal (1985).

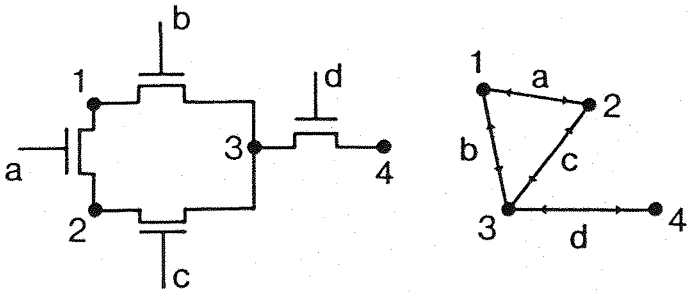
Modelling and test generation algorithms for MOS circuits

IEEE Trans, TC-34, pp 426-433.

10 P Goel (1980).

Test generation costs analysis and projections.

Proc 17th IEEE Design Automation Conference, pp 77-84.



transistor network graph representation

$$\begin{array}{rc} & \text{to:} \\ & \begin{array}{cccc}1&2&3&4\end{array} \\ A = \begin{array}{r}1\\2\\\text{from: }3\\4\end{array} & \begin{pmatrix}0&a&b&0\\a&0&c&0\\b&c&0&d\\0&0&d&0\end{pmatrix}\end{array}$$

adjacency matrix

Figure 5.1 Graph representation and adjacency matrix of a simple transistor network

$$A^2 = \begin{pmatrix} aa,bb & bc & ac & bd \\ cb & aa,cc & ab & cd \\ ca & ba & bb,cc,dd & 0 \\ db & dc & 0 & dd \end{pmatrix}$$

$$A^3 = \begin{pmatrix} acb,bca & \begin{array}{c}aaa,acc\\bba\end{array} & \begin{array}{c}aab,bbb\\bcc,bdd\end{array} & acd \\ \begin{array}{c}aaa,abb\\cca\end{array} & abc,cba & \begin{array}{c}aac,cbb\\ccc,cdd\end{array} & abd \\ \begin{array}{c}baa,bbb\\ccb,ddb\end{array} & \begin{array}{c}caa,bbc\\ccc,ddc\end{array} & bac,cab & \begin{array}{c}bbd,ccd\\ddd\end{array} \\ dca & dba & \begin{array}{c}ddd,dcc,\\dbb\end{array} & 0 \end{pmatrix}$$

Figure 5.2 Successive powers of the adjacency matrix of figure 1

Elementary path set

$$A^2 = \begin{pmatrix} 0 & bc & ac & bd \\ cb & 0 & ab & cd \\ ca & ba & 0 & 0 \\ db & dc & 0 & 0 \end{pmatrix}$$

$$A^3 = \begin{pmatrix} 0 & a & b & acd \\ a & 0 & c & abd \\ b & c & 0 & d \\ dca & dba & d & 0 \end{pmatrix}$$

$$r\{A \vee A^2 \vee A^3\} =$$

$$\begin{pmatrix} 0 & a,bc & b,ac & bd,acd \\ a,cb & 0 & c,ab & cd,abd \\ b,ca & ba,c & 0 & d \\ db,dca & dc,dba & d & 0 \end{pmatrix}$$

Cut-set

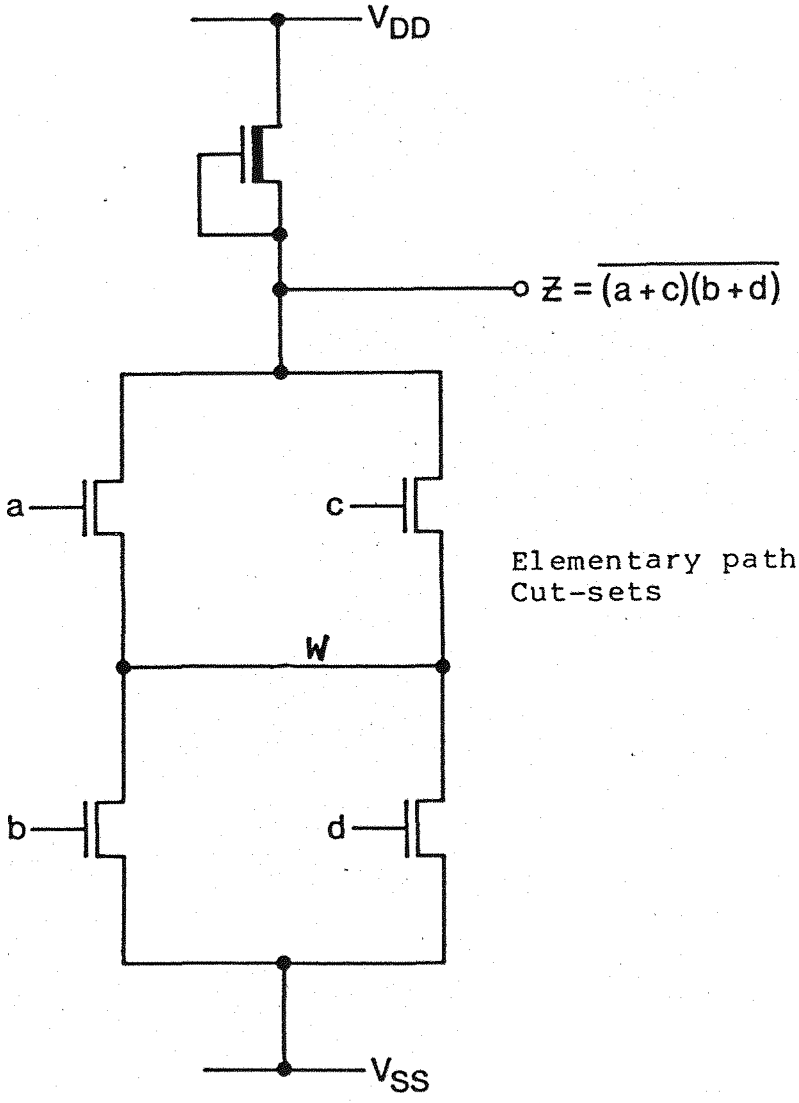
$$A^2 = \begin{pmatrix} ab & b,c & a,c & b,d \\ c,b & ac & a,b & c,d \\ c,a & b,a & bcd & 0 \\ d,b & d,c & 0 & d \end{pmatrix}$$

$$A^3 = \begin{pmatrix} a,b,c & a & b & a,c,d \\ a & a,b,c & c & a,b,d \\ b & c & a,b,c & d \\ a,c,d & a,b,d & d & 0 \end{pmatrix}$$

$$r\{A \vee A^2 \vee A^3\} =$$

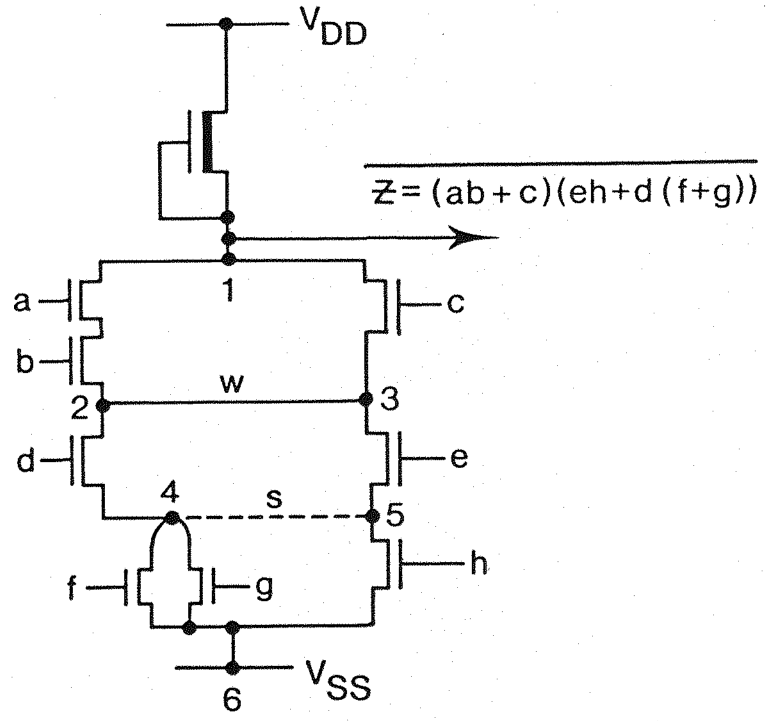
$$\begin{pmatrix} ab & ac,ba & ab,bc & d,cb,ab \\ ba,ac & ac & cb,ca & d,ca,cb \\ ab,bc & cb,ca & bcd & d \\ d,ab,bc & d,ca,cb & d & d \end{pmatrix}$$

Figure 5.3 Derivation of the elementary path set and set of cut-sets for the network of figure 1



Elementary paths: ab, awd, cd, cwb
Cut-sets : ac, bd

Figure 5.4 Circuit to show example of D-propagation



$$A = \begin{pmatrix} 0 & a.b & c & 0 & 0 & 0 \\ a.b & 0 & w & d & 0 & 0 \\ c & w & 0 & 0 & e & 0 \\ 0 & d & 0 & 0 & s & fvg \\ 0 & 0 & e & s & 0 & h \\ 0 & 0 & 0 & fvg & h & 0 \end{pmatrix}$$

$$A_{\text{path-set}} = \begin{pmatrix} 0 & ab & c & 0 & 0 & 0 \\ 0 & 0 & w & d & 0 & 0 \\ 0 & w & 0 & 0 & e & 0 \\ 0 & d & 0 & 0 & 0 & f,g \\ 0 & 0 & e & 0 & 0 & h \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad A_{\text{cut-set}} = \begin{pmatrix} 0 & a,b & c & 0 & 0 & 0 \\ 0 & 0 & U & d & 0 & 0 \\ 0 & U & 0 & 0 & e & 0 \\ 0 & d & 0 & 0 & s & fg \\ 0 & 0 & e & s & 0 & h \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

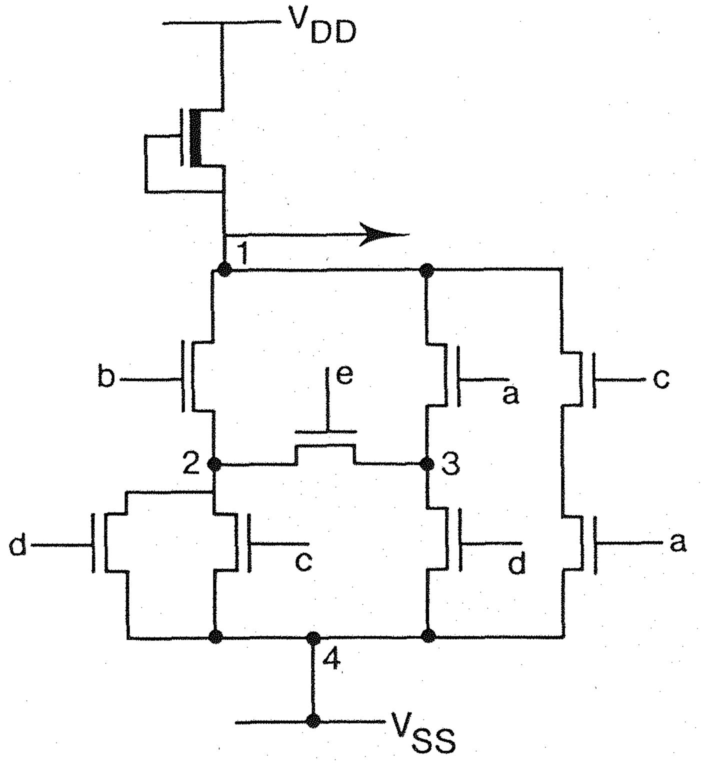
Elementary paths (open circuit tests) = {abdf, abdg, abweh, cwdf, cwdg, ceh}

Cut-sets ( short circuit tests ) = { ac, bc, de, dsh, fgh, efgs }

Test set : abcdefgh = 11001001, 11010010, 00110100, (from elementary path se

01011111, 10011111, 11101110, 11110001. (from cut-set)

Figure 5.5 Transformations of the adjacency matrix for open and short circuit test pattern generation



$$A = \begin{pmatrix} 0 & b & a_1 & c_2.a_2 \\ 0 & 0 & e & d_1vc_1 \\ 0 & e & 0 & d_2 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Elementary path set = $\{a_2c_2, bd_1, bc_1, bed_2, a_1ed_1, a_1ec_1, a_1d_2\}$

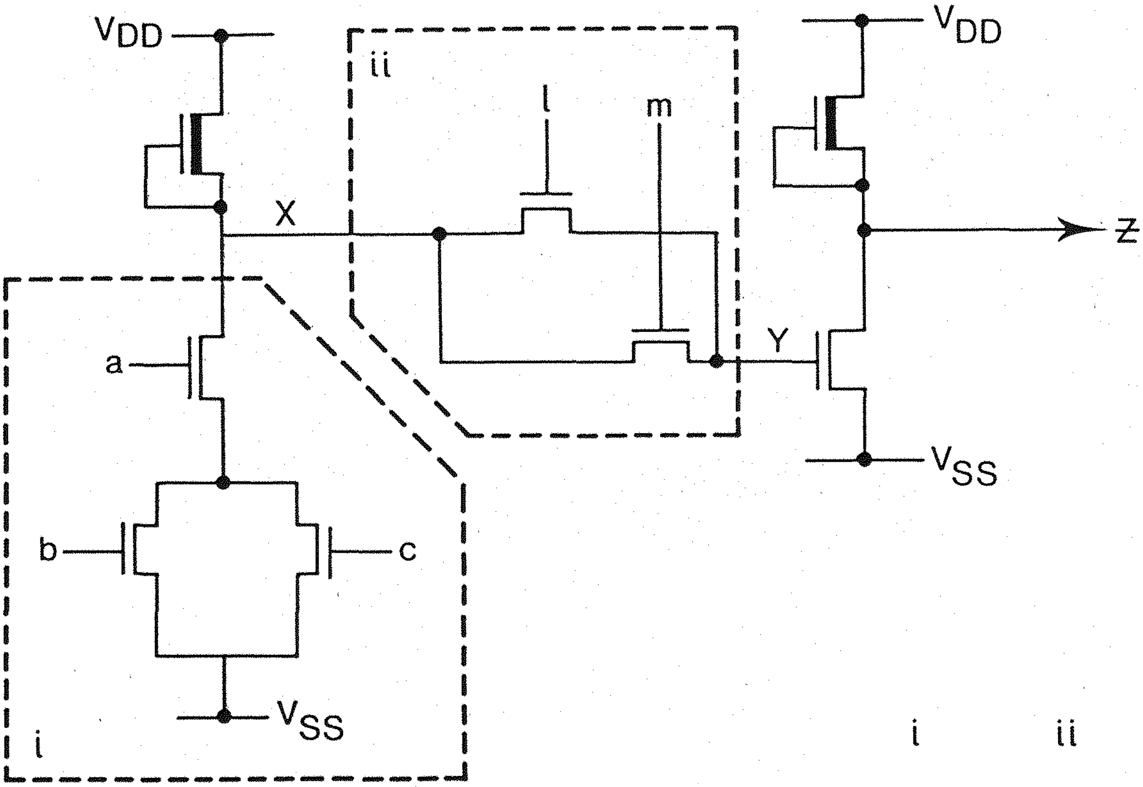
reduces to $\{bd_1, bc_1, a_1d_2, a_2c_2\}$

Set of cut-sets = $\{ba_1c_2, ba_1a_2, bed_2c_2, bed_2a_2, d_1c_1d_2c_2, d_1c_1ea_1c_2, d_1c_1ea_1a_2, d_1c_1d_2a_2\}$

reduces to $\{ba_1a_2, d_1c_1d_2c_2\}$

Test set : abcde = $\{01010, 01100, 10010, 10100, 00110, 11000\}$

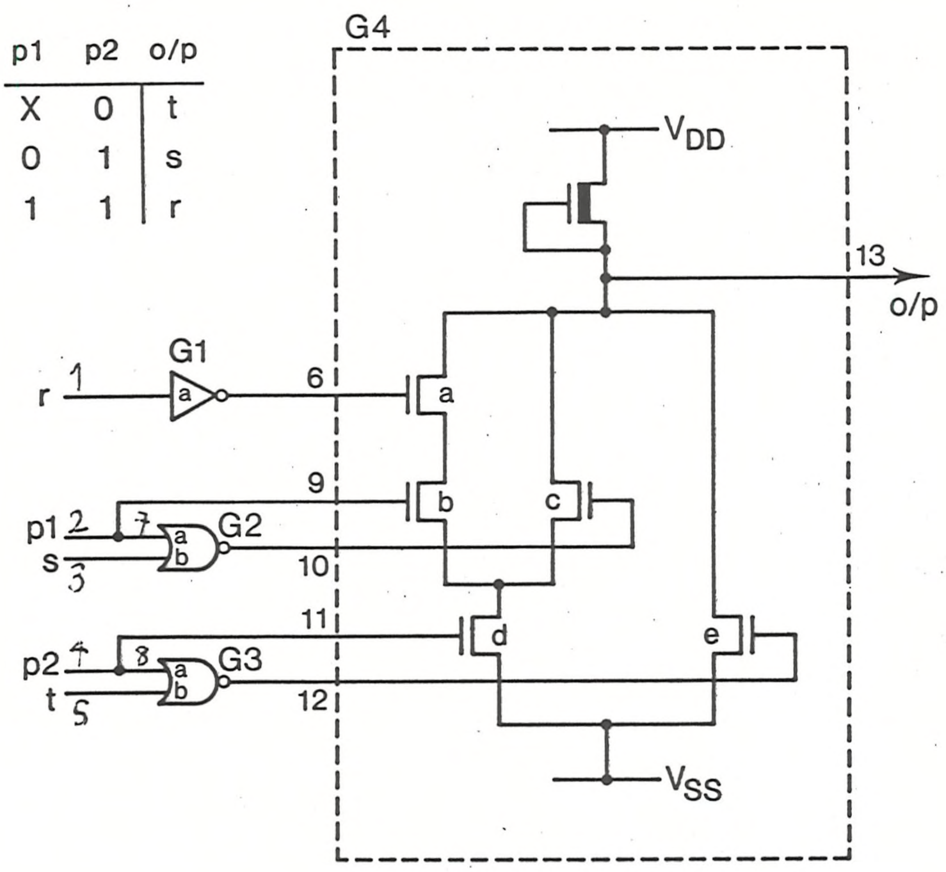
Figure 5.6 Test pattern generation for a circuit containing internal fan-out and redundancy



| | i | ii |
|---|---|---|
| elementary paths: | ab, ac | l, m |
| cut-sets: | a, bc | lm |

| | | |
|---|---|---|
| Test set: abclm = | 01101 | Initialise by forcing node Y to 1 |
| | 11000 | Covers l or m stuck-on |
| | 11010 | Covers a, b, or l stuck-off<br>(l and m stuck-on have already been tested for, so there is no need to apply the network ii cut-set again.) |
| | 10001 | Covers b or c stuck-on, and m stuck-off |
| | 10101 | Covers c stuck-off |
| | 01101 | Covers a stuck-on |

Figure 5.7 Test pattern generation for a simple pass transistor network



| p1 | p2 | o/p |
|---|---|---|
| X | 0 | t |
| 0 | 1 | s |
| 1 | 1 | r |

singular covers:

| | path sets | cut sets |
|---|---|---|
| G1 | (a) | (a) |
| G2,G3 | (a) (b) | (a,b) |
| G4 | (a,b,d) (e) | (a,c,e) (d,e) |
| | (c,d) | (b,c,e) |

Figure 5.8 3-to-1 multiplexor circuit to illustrate path testing technique

# CHAPTER 6 RELATED ISSUES IN VLSI CIRCUIT TESTING

## 6.1 Introduction

In this chapter, I examine issues related to NMOS circuit fault modelling. Firstly, some techniques to reduce the probability of "hard-to-test" faults from occurring are proposed. These may be thought of as "layout for testability" ideas, to distinguish them from the "architectural" design for testability features (scan path, chip partitioning etc) that are nowadays often found in VLSI circuits. Secondly, I discuss some of the fault modelling problems introduced by CMOS technology which has emerged as a rival technology to NMOS during the course of the project. I show how the path algebras-based test pattern generation may be extended to deal with these problems. Finally, there is a note on the use of self-test techniques in the light of the fault modelling problems associated with MOS circuits.

## 6.2 Layout for Testability

In this section I discuss some techniques for making MOS VLSI circuits more testable. These techniques aim to increase the testability of a circuit either by attempting to eliminate the possibility of a "hard-to-test" fault occurring, or by trying to ensure that potentially intermittent faults only manifest themselves as permanent stuck node faults.

In section 4.2 I made the point that circuit nodes can display intermittent fault effects if they have been made noise-sensitive as a result of a fault. In particular, a node that has been left "floating" as a result of an open circuit is susceptible to this kind of fault-effect as it may be controlled by signal pick-up from an adjacent track or the power rails. This problem has been studied for a fault-tolerant train controller chip where the designers wanted to be sure that only stuck node faults could occur (ref 1). Simulations of parallel wires - one floating and capacitatively coupled to a second signal-carrying wire were performed, and the designers found that if parallel wires were at least two track pitches apart, then for all reasonable lengths of track, a transistion on one would not cause gates driven by the floating node to switch erronously. In addition, to avoid noise injection from the power rails, the separation between ordinary signal carrying tracks and the power rails ought to be larger than currently-accepted design rules allow, particularly if the tracks are parallel for any significant distance.

Timing analysis has long been accepted as being exceedingly difficult for LSI/VLSI circuits (ref 2), even without taking into account the timing degradations introduced by faults in a chip. For this reason there has been a trend towards the use of synchronous circuit design techniques for VLSI circuits. Such design techniques will also aid testing (ref 3), as slow-to-switch nodes should manifest themselves as stuck nodes, because the well-defined clock phases limit the time available for switching to occur.

The problems caused by the open circuit shown in figure 4.4 may be removed by a simple alteration in the circuit layout (figure 6.1). Open circuits a-d may be modelled as transistor stuck-open faults, since they permanently disconnect the relevant transistor from ground or from the output node, Z. Similarly, open circuit e may be modelled as node Z stuck-at-1, because the output node can never be pulled to ground. Although, figures 4.4 and 6.1 are electrically identical in fault-free conditions, under potentially faulty conditions a piece of interconnect effectively becomes a component, since the topology of the circuit has gained critical importance. The technique of laying out a circuit to avoid faults that are "hard-to-test" because they do not produce stuck nodes has been called "physical design of testability" (ref 4). A set of rules needs to be developed to prevent these hard-to-test faults from occurring, thus making MOS circuits more testable.
One such set of rules has been developed for PLA's to ensure that totally self-checking two-rail checkers implemented as PLA's remain self-checking in the presence of common MOS faults (ref 5). These rules include increasing the spacing between adjacent lines, and increasing the width of pull-down transistors to ensure that bridging faults and short circuit faults are guaranteed to produce wired-AND effects, thus avoiding intermittent faults arising from indeterminate gate output voltages.

Finally, there is a class of pass transistor-based MOS circuits that do not have cut-sets. For example, the multiplexor circuit considered in section 4.4 is implemented such that every set of select inputs activates a single path through the pass transistor array; thus under fault-free conditions no cut-set can ever be defined. This poses testing problems for detecting stuck-on transistors, since in these circuits stuck-on faults may only be detected if two paths are selected and the output node is forced to an erroneous value. The problem is that simple 3-valued logic simulators are unable to model this behaviour, and hence more sophisticated 5 (or more) valued logic simulators are required (refs 6, 7). The more complicated mathematical manipulations used by these more sophisticated simulators result in longer run-times and, ultimately, higher chip production costs. If cut-sets were available, these networks would be testable by the method described in section 5: namely, charge up the output node by applying a path set through the array, apply a cut-set to the array, invert the inputs to the array, and check the output node remains at its previous logic state - if it does not, a transistor stuck-on fault must be present and is readily detected. Thus, if a simple design method can be found whereby cut-sets involving all the transistors in the pass transistor array may be applied, the simpler logic simulators may be used, with a net reduction in chip manufacturing costs.

Figure 6.2 shows just such a design method by which the inverters on the select lines has been replaced by exclusive-OR gates (exclusive-NOR gates can be used instead), with an additional "test" input. If the "test" input is 1, the multiplexor functions as normal since each exclusive-OR gate is functioning as an inverter. If the "test" input is 0, the exclusive-OR gates become "invisible", and putting a 0 on any select line input applies a cut-set across all the transistors controlled by it. Setting all the other select inputs to 1 will uncover any stuck-on transistors as explained above. Thus, a multiplexor circuit with n select lines requires an extra 5n transistors for this enhancement. Similar design methods may be used in bus driver circuits to make potential bus conflicts testable, and amenable to analysis by a simple 3-valued logic simulator.

## 6.3 CMOS

During the course of this project CMOS has started to emerge as the dominant VLSI technology. Its lower power dissipation and better noise margins have made it a most attractive VLSI technology, albeit a more difficult one than NMOS to process. A CMOS gate consists of a network of n-channel transistors ("n-transistors") between the gate output node and ground, and a complementary network of p-channel transistors ("p-transistors") between the gate output and $V_{DD}$ which share the same inputs. Applying an input vector to a CMOS gate thus simultaneously defines a cut-set in one network and a path-set in the other, and the output node is pulled up or down as a function of which network has a path activated through it. Wadsack (ref 8) showed how transistor stuck-open faults could produce parasitic latch behaviour in a CMOS gate, and that transistor stuck-on faults could produce "indeterminate" logic outputs. More recently, it has been shown that CMOS stuck-open faults may only be uncovered by so-called "robust" pairs of test vectors which satisfy certain strict conditions (ref 9). Path algebraic analysis provides a clear insight into these problems and their solutions.

In the presence of a stuck-open fault, certain gate input vectors will leave the output node floating because cut-sets have been applied to the p- and n-transistor networks simultaneously. Suppose n-transistor e in the circuit in figure 6.3 was stuck-open. On applying the input vector abcde = 01101, a cut-set is applied to the p-transistor network since transistors b, c and e are switched off; the corresponding path-set in the n-transistor network, however, does not pull the output node down to $V_{SS}$ because of the stuck-open fault, and the output retains its previous logic value by virtue of the associated output capacitance. Such faults may only be detected by charging the output node up and then attempting to discharge it through the transistor-under-test.

In the example, if the test vectors 10100, 10011, and 01010 were applied in that order to test for all input s-a-0 faults, the output would correctly remain at 0 for all three inputs and n-transistor e stuck-open would not be detected.

Care must be taken though to ensure that no other discharge paths are activated temporarily during the transition from the first input vector to the second as a result of differences in transistor propagation delays. For example, if the two vectors were 11000 (to force the output to 1) followed by 10011, n-transistor d may turn on before n-transistor b turns off, thus temporarily activating the path bd and discharging the output node but not through the transistor-under-test. "Robust" tests are guaranteed if the transistors in any one cut-set containing the transistor-under-test remain off for both input vectors. In other words, the output node must first be charged up by application of a cut-set containing transistor e and then discharged by applying a path-set containing transistor e and none of the other transistors in the applied cut-set. For example, the test vector pair 01100, 01101 satisfies these conditions since no temporary paths are set up that do not include the transistor-under-test (n-transistor e). Propagating D-cubes automatically fulfil these conditions if the input(s) labelled D or D' are first set to 0 (to charge up the output), then to 1 (to discharge the output via the n-transistors-under-test). P-transistors may be tested for stuck open faults if the tests are applied in the reverse order. Before discussing how this method may be expanded to testing CMOS circuits made up of many gates connected together, stuck-on faults in CMOS circuits will be considered.

In the presence of a stuck-on fault a CMOS gate acts as a voltage divider with the output voltage level a function of the on-resistances of the transistors in the gate and their topology. This means it is virtually impossible to determine a suitable gate input vector to uncover a stuck-on fault using a logic simulator because the logic simulator cannot predict the gate output voltage, and hence the output logic value. Furthermore, some stuck-on faults are guaranteed not to produce an error because of the circuit topology. Figure 6.4 shows a good example of this: even if the stuck-on fault is activated the output voltage will almost certainly still be taken as a 0 by a following gate. (However, the steady state current flowing in the gate may well exceed electromigration limits and induce an open circuit in the gate before long, thus producing a low reliability component). It is now widely acknowledged that the best way to detect stuck-on faults is to monitor the power supply current, since activation of a stuck-on fault will produce a temporary increase of several orders of magnitude in the supply current (ref 10). Thus stuck-on faults need not be propagated to a primary output to be detected. Stuck-on faults are tested for by applying a cut-set including the transistor-under-test. In the presence of a stuck-on fault, any such input vector simultaneously activates a path-set in each network: one fault-free as a result of the complementary nature of the networks, and the other containing the transistor-under-test. A test strategy for CMOS circuits that uses the transistor/interconnect open/short circuit fault model as described in the previous chapter will now be described.

The CMOS test strategy is similar to the NMOS strategy in that it uses path algebras and the D-Algorithm as a framework for test pattern generation and evaluation. The mechanisms involved in fault insertion, the D-drive, and the consistency check remain the same: the complications arise in checking pairs of vectors for stuck-open test "robustness".

Stuck-open faults are inserted (and propagated) by applying a pair of test vectors such that a cut-set containing the transistor-under-test is followed by a path-set containing the transistor-under-test and none of the other transistors in the cut-set. A simple way to achieve this is to apply a cut-set including the transistor-under-test (with all transistors not in the cut-set forced to the opposite logic value), and then to invert the input to the transistor-under-test only. The output node (for an n-transistor stuck-open test) is thus D followed by D' (vice versa for a p-transistor stuck-open test) and all transistors in the cut-set in the same network as the transistor-under-test are also tested for stuck-on faults by the first test input vector.

The main problem is to propagate the fault such that stuck-open faults along the "sensitive path" derived by the D-drive are also uncovered. Propagating D-cubes are derived as before; however, changing the input of the transistor-under-test in the fault insertion phase may affect gates on the sensitive path for the propagation of the second test input. Thus, a method for determining if two successive inputs constitute "robust" propagating D-cubes is required. The simplest way to accomplish this is to check that the input(s) of the n-transistor(s) at D or D' and any n-transistor inputs at 0 in both inputs make up a cut-set. If they do, the stuck-open fault on the propagating transistor(s) is (are) covered, and previous stuck-open faults are propagated; if not, the stuck-open fault(s) on the propagating transistor(s) is (are) not covered, but the faults in other gates along the sensitive path are still propagated. The propagating D-cubes also test for stuck-on faults irrespective of the "robustness" property on any n-transistor labelled with a D' or any p-transistor labelled with a D. The consistency check remains unaltered. This method obviously represents a substantial increase in computing effort with all the extra work involved in determining "robustness", but the method does explicitly test for stuck-open faults. This is not possible using a Boolean logic equivalent diagram to represent the CMOS circuit-under-test, unless extraneous latches are included at the output of every gate to model the faulty behaviour.

The CMOS equivalent of NMOS pass transistors are transmission gates, which consist of pairs of complementary transistors whose sources and drains are shared (figure 6.5). Stuck-on faults in this structure are tested for in the same way as pass transistors; stuck-open faults do not produce hard faults because the two transistors in parallel mean that if one transistor is stuck-open the other transistor can still propagate signals. However, the propagation delays through the transmission gate are increased because of the increased on-resistance of the transmission gate and the p- (n-) transistor's threshold voltage is added to (subtracted from) the logic zero (one) voltage level. The transmission gate single transistor stuck-open fault may thus only be tested for by running the test sequence at full operating speed and hoping that the increased delay is severe enough to register as a fault; otherwise, a potentially low reliability component may be sold.

Recently, 2-phase and 4-phase clocked CMOS design methodologies have been proposed which replace the p-transistor network by a single clocked p-transistor and an additional n-transistor controlled by the same clock input (refs 11, 12). A schematic of such a methodology is shown in figure 6.6. Stuck-open and stuck-on faults in the clocked transistors produce catastrophic fault-effects (output stuck-at faults) and the remaining testing of the n-transistor network exactly matches that described in the previous chapter. These new dynamic CMOS techniques thus remove many of the difficulties associated with CMOS testing (ref 13), as well as introducing improved performance (faster switching, denser logic implementation etc) and simpler layouts.

## 6.4 Random pattern testing of MOS VLSI circuits

A radical alternative to fault-orientated testing is random (or pseudo-random) pattern testing (ref 14), which by-passes the need for computationally expensive fault-orientated automatic test pattern generation. Pseudo-random test pattern sequences may be readily generated using a linear feedback shift register with suitable feedback taps. They lend themselves to built-in test methods, in which a chip tests itself at its specified clock frequency, thus uncovering timing problems not necessarily detectable by high-speed automatic test equipment. In addition, pseudo-random test patterns have been found to uncover some "hard-to-test" faults not detected by a stuck-at fault test set (ref 15), on account of the much larger number of applied test inputs compared with fault-orientated test sets. For these reasons, pseudo-random test patterns with signature analysis methods that also employ linear feedback shift registers have been accepted as an efficient method of (self-) testing VLSI circuits.

Problems arise, however, when attempting to decide how long such sequences should be to test a VLSI chip adequately. Exhaustive test pattern sequences are not a practical option once the number of circuit inputs exceeds 24 or so, since the test time (with, for example, a 10 MHz tester) would rapidly exceed the maximum of a few seconds per chip required by production volume testing. Thus, random pattern sequences need to be truncated, and evaluated probabilistically against a random sample of stuck-at faults (ref 16). Unfortunately, as shown earlier, stuck-at faults do not accurately reflect the possible failure mechanisms and associated fault-effects commonly found in MOS VLSI circuits. Therefore, such probabilistic evaluations will not give reliable results - the solution to this problem is, of course, to use an appropriate and accurate MOS fault model to determine the "real fault" coverage, such as described in this thesis.

## REFERENCES

1 I Masuda, M Ueno and K Tashiro (1983).
A fault-tolerant MOS-LSI for train controller applications.
IEEE ISSCC Digest of Technical Papers, pp 138-139.

2 L C Bening, T A Lane and J E Smith (1982).
Developments in logic network path delay analysis.
Proc 19th IEEE Design Automation Conference, pp 605-615.

3 J R Grierson (1984).
Gate Arrays - computer aids, automation and the UK5000.
Proc IEE Electronic Design Automation Conference, pp 1-4.

4 Y El-Ziq (1983).
Classifying, testing and elminating VLSI MOS failure modes.
VLSI Design, September 1983, pp 30-35.

5 Y Tamil and C H Sequin (1984).
Design and application of self-testing comparators implemented with MOS PLA's.
IEEE Trans, TC-33, pp 493-506.

6 P Banerjee and J A Abraham (1985).
A multivalued algebra for modelling physical failures in MOS VLSI circuits.
IEEE Trans, TCAD-4, pp 312-321.

7 M K Reddy, S M Reddy and P Agrawal (1985).
Transistor level test generation for MOS circuits.
Proc 22nd IEEE Design Automation Conference, pp 825-828.

8 R L Wadsack (1978).
Fault modelling and logic simulation of CMOS and MOS integrated circuits.
Bell System Technical Journal 57, pp 1449-1474.

9 S M Reddy, M K Reddy and V D Agrawal (1984).
Robust tests for stuck-open faults in CMOS combinational logic circuits.
Proc 14th IEEE Fault-Tolerant Computing Symposium, pp 44-49.

10 M Levi (1981).
CMOS is more testable.
Proc IEEE Test Conference pp 217-220

11 R H Kranbeck, C M Lee and H-F S Law (1982).
High-speed compact circuits with CMOS.
IEEE Journal of Solid-State Circuits, SC-17, pp 614-619.

12 D J Myers and P A Ivey (1985).
A design style for VLSI CMOS.
IEEE Journal of Solid-state Circuits, SC-20, pp 614-619.

13 V G Oklobdzija and P G Kovijanic (1984).
On testability of CMOS-domino logic.
Proc 14th IEEE Fault-Tolerant Computing Symposium, pp 50-55.

14 R David and P Thevenod-Fosse (1981).
Random testing of integrated circuits.
IEEE Trans, TIM-30, pp 20-25.

15 F Motika, J A Waicukanski, E Lindbloom and E B Eichelburger (1983).
An LSSD pseudo-random pattern test system.
Proc IEE Test Conference, pp 283-288.

16 T W Williams (1985).
Test length in a self-testing environment.
IEEE Design and Test of Computers, April 1985, pp 59-63.

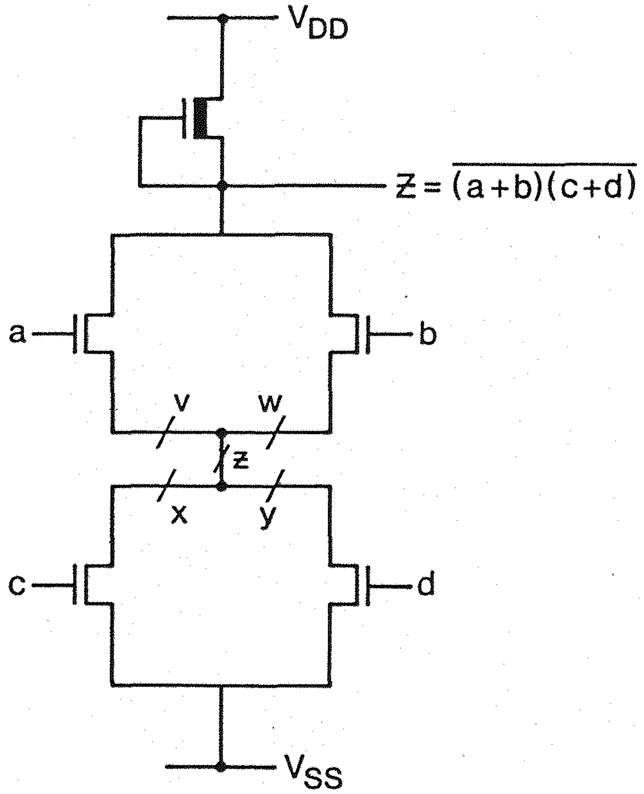


Figure 6.1 Example of an NMOS circuit laid out so that open circuits v-z are modellable as stuck nodes

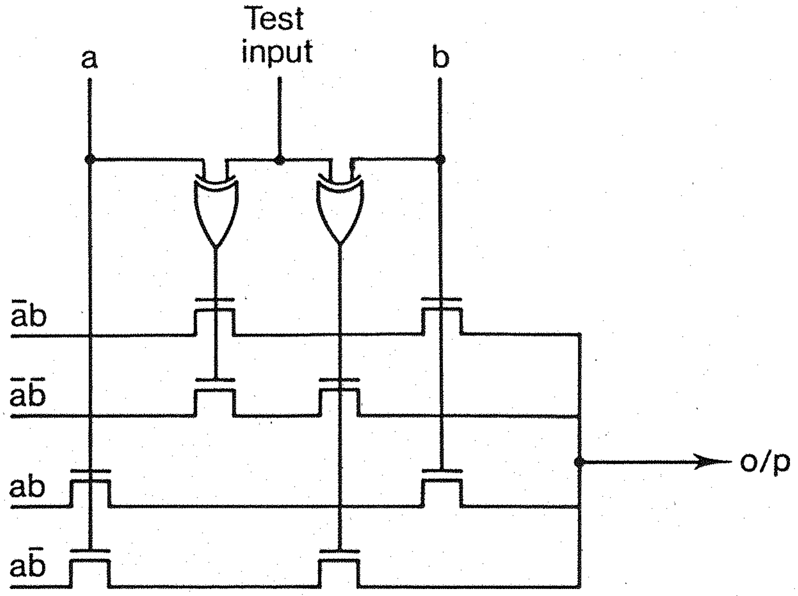


Figure 6.2 Function block modified to permit path testing

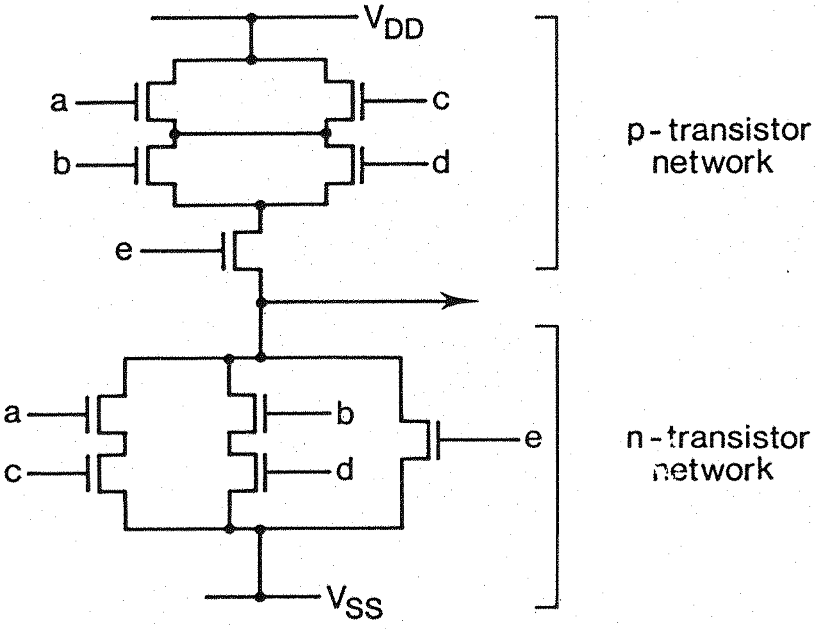
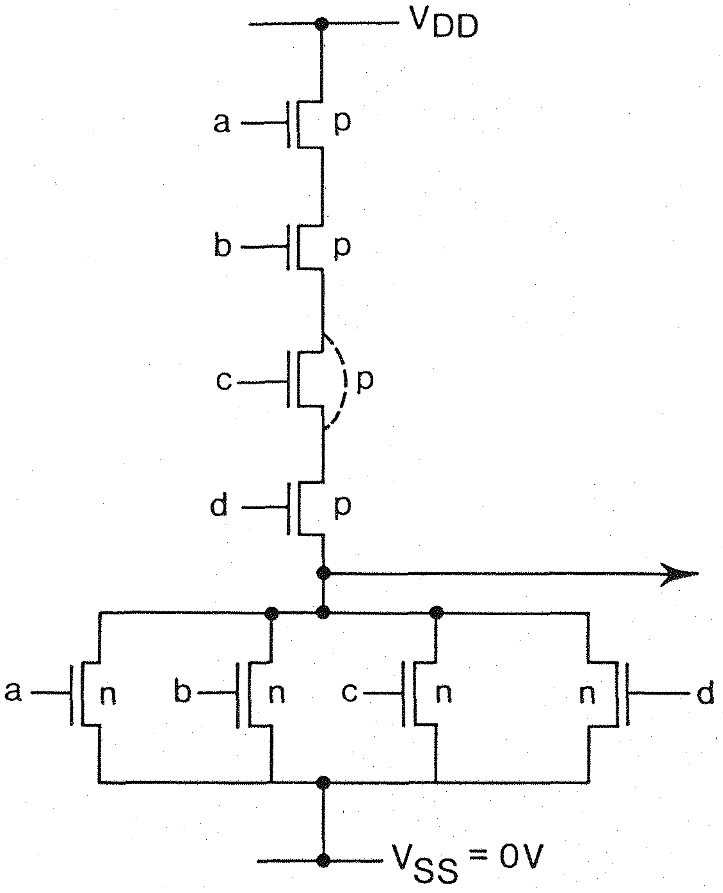
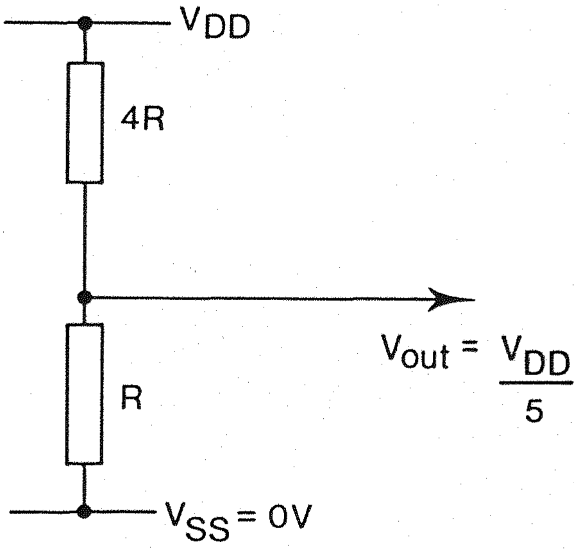


Figure 6.3 CMOS complex gate



Transistor diagram of 4-input CMOS NOR gate with p-transistor c stuck-on



Equivalent resistor network for test input abcd = 0010

Figure 6.4 Example of an "undetectable" stuck-on fault in a simple CMOS gate

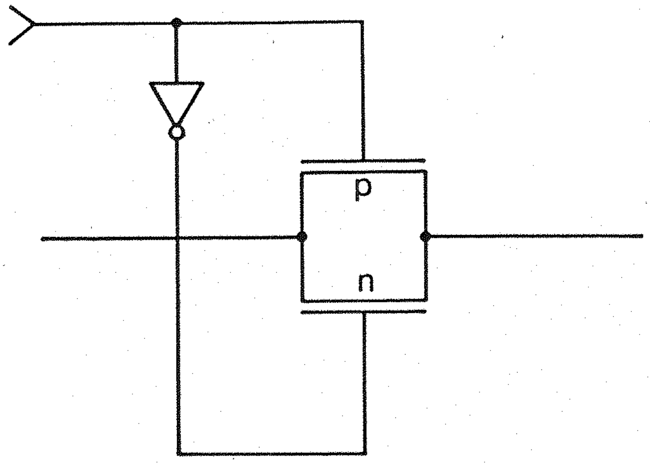


Figure 6.5   CMOS transmission gate

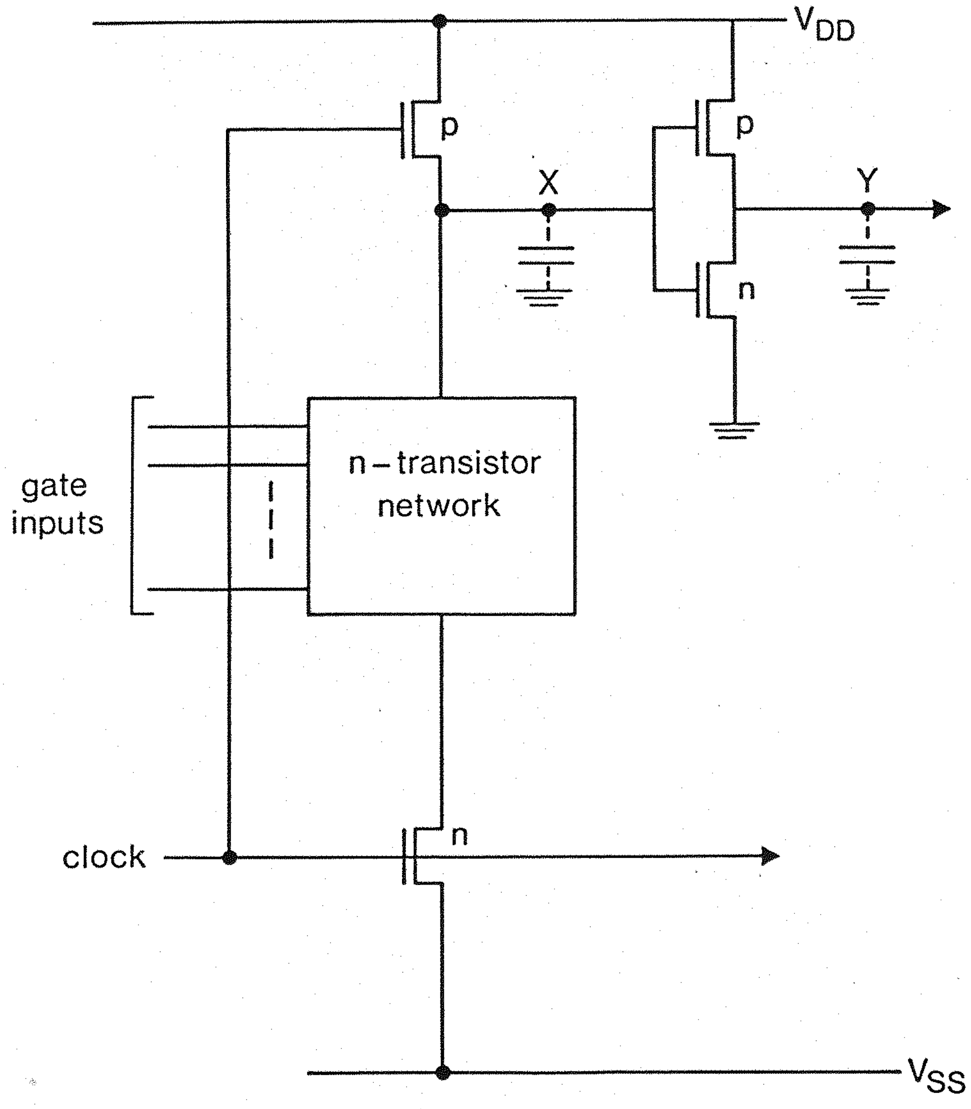


Figure 6.6   Schematic diagram of a dynamic CMOS circuit

# CHAPTER 7 SUMMARY

The central aim of this project was to assess the validity of the stuck-at fault model for NMOS VLSI circuit testing. This has been achieved by studying the available literature on the subject, by consultation with processing and reliability engineers at British Telecom, and by hardware and software analyses of faulty circuits. Originally, the proportion of common NMOS failure mechanisms that manifested themselves as non-stuck-at nodes was to have been calculated from the data collected; however, the relative incidence rates of the different failure mechanisms were found to vary so much that this calculation proved to be impossible to perform. Furthermore, the same failure mechanism could produce both stuck-at faults and non-stuck-at faults, as a function of its severity. Nevertheless a clear picture of the relationship between the various failure mechanisms and their associated fault-effects was built up, and the overall conclusion of the various investigations was that the stuck-at fault model did not accurately reflect the faults commonly found in NMOS circuits. The most serious deficiency in the model was considered to be its use of the gate-level rather than transistor-level representation of a circuit, since these two representations are not topologically equivalent. This was found to result in the possible non-coverage of real faults "in the silicon" (especially in pass transistor networks), and in potentially misleading and inaccurate test pattern generation and evaluation.

A new fault model, operating at the transistor level of abstraction, was proposed which reflected more accurately actual failure mechanisms. The model assumed that a single transistor could be stuck-on or stuck-open, or that a piece of interconnect could be open circuit or short-circuited to another piece of interconnect. The failure mechanisms were thus considered to lead to the introduction of spurious paths through a network which should not have existed under fault-free conditions (stuck-on  transistors or short circuits between pieces of interconnect), or to the absence of paths which should exist (stuck-open transistors or open circuits in pieces of interconnect). Using the new fault model, test patterns could be readily generated using an adaptation of the D-Algorithm that employed graph-theoretic methods to determine possible paths through a network, by treating transistors and pieces of interconnect as edges of graphs. Unfortunately, time permitted only a part of the described algorithm to be implemented in software and therefore no data was available with which to assess the relative "efficiencies" of stuck-at fault testing and path testing. However, the indications were that use of such a physically-realistic fault model, even though it operates at a more detailed level, as well as explicitly testing for realistic faults, might actually reduce effort by avoiding any necessity to introduce additional complexity to overcome shortcomings in the higher-level stuck-at fault model. For example, the algorithm was readily extended to generate "robust", hazard-free tests to uncover parasitic latch behaviour in CMOS circuits. This was possible because the fault model operates at the appropriate level of abstraction, enabling a problem to be cast in such a way that its solution was soon apparent.

The investigations of the failure mechanisms also yielded valuable insights into the benefits of testing chips at elevated temperatures and adjusted power supply voltages. The investigations showed that non-severe failure mechanisms could produce timing degradations and gate output voltage level shifts, leading to intermittent faults. Further analysis showed that so-called "burn-in" techniques would uncover many of these problems by exacerbating the severity of the faults, thus effectively converting them to stuck-node faults. Reports in the literature have borne this point out and this work confirms the value of stress testing VLSI components to uncover so-called "soft" faults.

Further work needs to be carried out before detailed quantitative assessments of the proposed test pattern generation algorithm can be made. In particular, the modified D-Algorithm must be implemented in software in conjunction with the existing program that determines complex gates' singular covers to produce a fully automatic NMOS test pattern generator. Test pattern sequences produced by the proposed algorithm could then be evaluated against those of other test pattern generation systems by comparing them for effectiveness (by identification of good and faulty circuits) and cpu time and memory requirements. Other work might include implementation of the extensions to the algorithm to perform CMOS testing, and investigation of graph theoretic techniques to replace the modified D-Algorithm.

Ultimately, testing activities are governed by economic and commercial considerations, including chip cost, and the reputation a company has for producing and selling high-reliability components at an acceptable price. Any testing methodology is thus selected (and judged) on these criteria, as much as on what may be termed "academic rigour". The work described in this thesis plainly has not been evaluated against commercial criteria; however its subject matter - fault modelling - underlies all testing activities. Therefore, the main conclusion of this project - namely, the stuck-at fault model is inadequate for MOS VLSI circuit testing - is expected to be borne out by fundamental changes in VLSI testing strategies in the years to come.