Cost Modelling and Concurrent Engineering for Testable Design

A thesis submitted to Brunel University
for the degree of Doctor of Philosophy
by

Jochen Helmut Dick, Dipl.Ing.Univ.

Department of Electrical Engineering and Electronics
Brunel University
April 1993

To Stefani

Acknowledgements

I would like to acknowledge the guidance, encouragement and support of Professor A.P. Ambler throughout this research. I am grateful to Dr. E. Trischler for his support of this work and many helpful discussions. I am also grateful to my colleagues in the test engineering team at Siemens-Nixdorf and the ESPRIT EVEREST project. In particular I would like to thank Dr. C. Dislis whose technical support and critical appraisal has been extremely valuable, and Dr. J. Armaos for the helpful discussions in the field of Monte Carlo simulation.

Acknowledgements

I would like to acknowledge the guidance, encouragement and support of Professor A.P. Ambler throughout this research. I am grateful to Dr. E. Trischler for his support of this work and many helpful discussions. I am also grateful to my colleagues in the test engineering team at Siemens-Nixdorf and the ESPRIT EVEREST project. In particular I would like to thank Dr. C. Dislis whose technical support and critical appraisal has been extremely valuable, and Dr. J. Armaos for the helpful discussions in the field of Monte Carlo simulation.

ABSTRACT

As integrated circuits and printed circuit boards increase in complexity, testing becomes a major cost factor of the design and production of the complex devices. Testability has to be considered during the design of complex electronic systems, and automatic test systems have to be used in order to facilitate the test. This fact is now widely accepted in industry. Both design for testability and the usage of automatic test systems aim at reducing the cost of production testing or, sometimes, making it possible at all. Many design for testability methods and test systems are available which can be configured into a production test strategy, in order to achieve high quality of the final product. The designer has to select from the various options for creating a test strategy, by maximising the quality and minimising the total cost for the electronic system.

This thesis presents a methodology for test strategy generation which is based on consideration of the economics during the life cycle of the electronic system. This methodology is a concurrent engineering approach which takes into account all effects of a test strategy on the electronic system during its life cycle by evaluating its related cost. This objective methodology is used in an original test strategy planning advisory system, which allows for test strategy planning for VLSI circuits as well as for digital electronic systems.

The cost models which are used for evaluating the economics of test strategies are described in detail and the test strategy planning system is presented. A methodology for making decisions which are based on estimated costing data is presented. Results of using the cost models and the test strategy planning system for evaluating the economics of test strategies for selected industrial designs are presented.

Table of Contents

1. Introduction **1**
1.1. Description of the Problem 1
1.2. Testing, Error Free Design and Production 4
1.3. Structure of the Thesis 6

2. Test Methods **8**
2.1. Introduction 8
2.2. Description and Classification of Test Methods 10
2.2.1. Design for Testability Methods for VLSI Devices 10
2.2.1.1. Ad-Hoc Methods 13
2.2.1.2. Scan Structure Methods 14
2.2.1.3. Self Test Methods 16
2.2.1.4. Design Specific Methods 17
2.2.2. Test Generation Methods and Fault Simulation Methods 20
2.2.2.1. Test Pattern Generation Methods 20
2.2.2.2. Fault Simulation 22
2.2.2.3. Testability Analysis 22
2.2.3. Test Application Methods 22
2.2.3.1. Component Test 23
2.2.3.2. Bare Board Test 23
2.2.3.3. Board Test 23
2.2.3.4. System Test 27
2.3. Economical Impact 27
2.4. Summary 29

3. Test Strategy Planning **31**
3.1. Introduction 31
3.2. Definition of Test Strategy Planning and Classification of Test Strategy Planning Systems 31

4. Test Economics **36**
4.1. Introduction 36
4.2. Economic Modelling Techniques 37
4.3. Methods for the Estimation of Life Cycle Cost 39
4.4. The Impact of Test Strategies on the Economics of Electronic Systems 41
4.5. Structure of the Test Economics Model 44
4.6. A Life Cycle Test Economics Model for VLSI Devices and VLSI Based Systems and Boards 47
4.6.1. The Test Economics Model for ASIC components 47
4.6.1.1. The Production Costs 48
4.6.1.2. The Design Costs 50
4.6.1.3. The Test Costs 51
4.6.2. The Test Economics Model for Boards 52
4.6.3. The Test Economics Model for Systems 56
4.6.4. The Test Economics Model for the Field Costs 57
4.6.5. Consideration of Interest Rates 59

4.7. Summary 60

5. ECOTEST 61
5.1. Introduction 61
5.2. The Philosophy of ECOTEST 62
5.3. ECOTEST in a Test Engineering Environment 67
5.4. The EVEREST Test Strategy Planner 68
5.4.1. The Data Interfaces 68
5.4.2. The Functions of ECOTEST 72
5.5. Cost Modelling Techniques 75
5.6. The Test Strategy Planner 78
5.7. Conclusions 81

6. Sensitivity Analysis 82
6.1. Introduction 82
6.2. Description of the Problem 83
6.3. Monte Carlo Methods for Dynamic Sensitivity Analysis 86
6.3.1. Computation of Random Values for a Given Distribution Function 89
6.3.1.1. Marsaglia Table Method 90
6.3.1.2. Box and Miller Method 91
6.3.1.3. Central Limit Theorem Method 91
6.3.1.4. Test of the Accuracy of the Methods 92
6.3.2. Correlation of Input Parameters 93
6.4. General Sensitivity Analysis 95
6.4.1. Estimation of Mean Value and Variance of Sensitivity 96
6.4.1.1. The Algorithm 96
6.4.1.2. Estimation Error of the Monte Carlo Simulation 99
6.4.1.3. Results 104
6.4.2. Estimation of the Maximum Sensitivity 111
6.4.2.1. The Algorithm 112
6.4.2.2. Reduction of the Sample Size 115
6.4.2.3. Results 116
6.5. Iterative Sensitivity Analysis 117
6.6. Total Variation Sensitivity Analysis 118
6.6.1. The Algorithm 119
6.6.2. Determination of the Number of Simulations Needed 120
6.7. Summary 122

7. ECOvbs: A Test Strategy PLanner for VLSI based Systems 124
7.1. Philosophy of ECOvbs 125
7.2. System Overview 130
7.3. The Design Description 133
7.4. The Test Method Descriptions and Test Equipment Descriptions 136
7.4.1. Syntax of Test Method Descriptions 137
7.4.2. Syntax of the Test Equipment Description 138
7.5. The Cost Models and the Cost Evaluator 139
7.5.1. The Cost Modelling Technique 140
7.5.2. Description of the Cost Models 144
7.6. Calculation of Fault Spectrum and Defect Spectrum 146
7.6.1. Calculation of Defect Spectrum after Manufacture 148

7.6.2. Calculation of Fault Spectrum 149
7.6.3. Calculation of Defect Spectrum for Repair 150
7.7. The Test Strategy Planner 152
7.8. The User Interface 155
7.8.1. The User Commands of ECOvbs 156
7.8.1.1. The ECOvbs handler 156
7.8.1.2. The DSR handler 157
7.8.1.3. The TSP handler 158
7.8.1.4. The CM handler 160
7.9. Summary 162

8. Test Economics Evaluation 163
8.1. Introduction 163
8.2. The Economics of Boundary Scan 163
8.3. Test Strategy Planning with ECOTEST 169
8.3.1. Test Strategy Planning for Selected Industrial Designs 169
8.3.2. Test Strategy Planning with the Total Variation Method 176
8.3.3. Test Strategy Planning for RAND_CIRC with the Total Variation Method 177
8.3.4. Test Strategy Planning for DEMO_CIRC with Inaccurate Input Data 179
8.3.5. Test Strategy Planning for Industrial Designs with the Total Variation Method 181
8.4. Test Strategy Planning with ECOvbs 182
8.5. Summary 188

9. Conclusions 190
9.1. Summary of the Work 190
9.2. Conclusions 191
9.3. Future Research 192

References 195

List of Figures

Figure 1: The increasing IC circuit density [Sed92] 1
Figure 2: The increasing gate per pin ratio [Sed92] 2
Figure 3: The quality options for test strategies 5
Figure 4: Classification of Scan Structures 16
Figure 5: Structure of test strategies 32
Figure 6: Classification of test economics model parameters 43
Figure 7: Life cycle model for electronic systems with regard to test 46
Figure 8: Flow diagram of board level phases 53
Figure 9: The test/repair loop of the test application phase 55
Figure 11: Field repair loop 58
Figure 12: Shareability of test resources 66
Figure 13: The ECOTEST architecture 69
Figure 14: Example of a cost model and its internal representation 75
Figure 15: Structure of the cost model calculator 80
Figure 16: Standard error of Monte Carlo simulation for group 1 100
Figure 17: Standard error of Monte Carlo simulation for group 2 100
Figure 18: Standard error of Monte Carlo simulation for group 3 101
Figure 19: Mean sensitivity for a 1% variation and no DFT 102
Figure 20: Mean sensitivity for a 20% variation and no DFT 102
Figure 21: Mean sensitivity for a 1% variation and scan path 103
Figure 22: Mean sensitivity for a 20% variation and scan path 103
Figure 23: Mean sensitivity for a 1% variation and self test 104
Figure 24: Mean sensitivity for a 20% variation and self test 104
Figure 25: 99% range of sensitivity with a 1% variation, no DFT 106
Figure 26: 99% range of sensitivity with a 10% variation, no DFT 107
Figure 27: 99% range of sensitivity with a 20% variation, no DFT 107
Figure 28: 99% range of sensitivity with a 1% variation, scan path 108
Figure 29: 99% range of sensitivity with a 10% variation, scan path 108
Figure 30: 99% range of sensitivity with a 20% variation, scan path 109
Figure 31: 99% range of sensitivity with a 1% variation, self test 109
Figure 32: 99% range of sensitivity with a 10% variation, self test 110
Figure 33: 99% range of sensitivity with a 20% variation, self test 110
Figure 34: 99% range of sensitivity with a 1% variation, no DFT, low production volume 111
Figure 35: 99% range of sensitivity with a 20% variation, no DFT, low production volume 111
Figure 36: Maximum sensitivity values 117
Figure 37: Convergence of Monte Carlo simulation for no DFT 121
Figure 38: Convergence of Monte Carlo simulation for scan path 122
Figure 39: Convergence of Monte Carlo simulation for self test 122
Figure 40: Architecture of ECOvbs 132
Figure 41: Sensitivity of the total cost in DM to the variation of parameter values in percent of the nominal value for In-circuit test 167
Figure 42: Sensitivity of the total cost in DM to the variation of parameter values in percent of the nominal value for boundary scan 167

Figure 43: Sensitivity of the total cost difference in DM to the variation of parameter values in percent of the nominal value for In-circuit test minus boundary scan 168
Figure 44: Cost of test strategies for automatic test strategy planning of ERCO 171
Figure 45: Cost of test strategies for automatic test strategy planning of PRI 171
Figure 46: Cost of test strategies for automatic test strategy planning of AM2909 172
Figure 47: Cost of test strategies for automatic test strategy planning of AMS 172
Figure 48: Cost of test strategies for automatic test strategy planning of SCR 173
Figure 49: Automatic test strategy planning by using the previous automatic test strategy as the initial test strategy 174
Figure 50: Cost of automatically generated test strategies with different initial test strategies for the design PRI 175
Figure 51: Cost of automatically generated test strategies with different initial test strategies for the design AMS 175
Figure 52: Distribution of total cost for final test strategy 176
Figure 53: Distribution density function of total cost for RAND_CIRC 178
Figure 54: Distribution function of total cost for RAND_CIRC 178
Figure 55: Distribution function of total cost for DEMO_CIRC 180
Figure 56: Cost of board test strategies 184
Figure 57: Number of faults after final test 185
Figure 58: Number of remaining faults per test stage 186
Figure 59: Cost per detected fault in ECU per test stage for test strategy TS1 187
Figure 60: Cost per detected fault in ECU per test stage for test strategy TS2 187
Figure 61: Cost per detected fault in ECU per test stage for test strategy TS3 188
Figure 62: Cost per detected fault in ECU per test stage for test strategy TS4 188

List of Tables

Table 1: Self test methods 17
Table 2: Cost impact of test application methods 29
Table 3: Cost estimation methods [Mad84] 40
Table 4: Commands of ECOTEST 74
Table 5: Ranges for Marsaglia Table 91
Table 6: Ranges for normal distribution test 92
Table 7: Result of χ^2 test 93
Table 8: Distribution characteristics of cost model parameters 97
Table 9: Parameter classification concerning its sensitivity 105
Table 11: Main data of the industrial designs used for ECOTEST 169
Table 12: CPU times and cost savings by test strategy planning for industrial designs 173
Table 13: Description of normal distributed parameters 177
Table 14: Description of normal distributed parameters 180
Table 15: Test Strategies for DEMO_CIRC 180
Table 16: Run times for industrial designs 182
Table 17: Main design data of computer board 182
Table 18: Test strategies for the computer board 183

Chapter 1

Introduction

1.1. Description of the Problem

The increase in the complexity of integrated circuits (figure 1) and the increase of the gates per pin ratio (figure 2) of integrated circuits is now widely accepted to cause tremendous problems for IC testing. Both values have been increasing exponentially with time for the transistors per gate and with the generation of technology for the gates per pin ratio for the last 20 years. With the increasing complexity of ICs, the production cost per gate is decreasing, and if the testing cost would remain unchanged, they become more important as they are becoming an increasing portion of the total cost [Tur90].

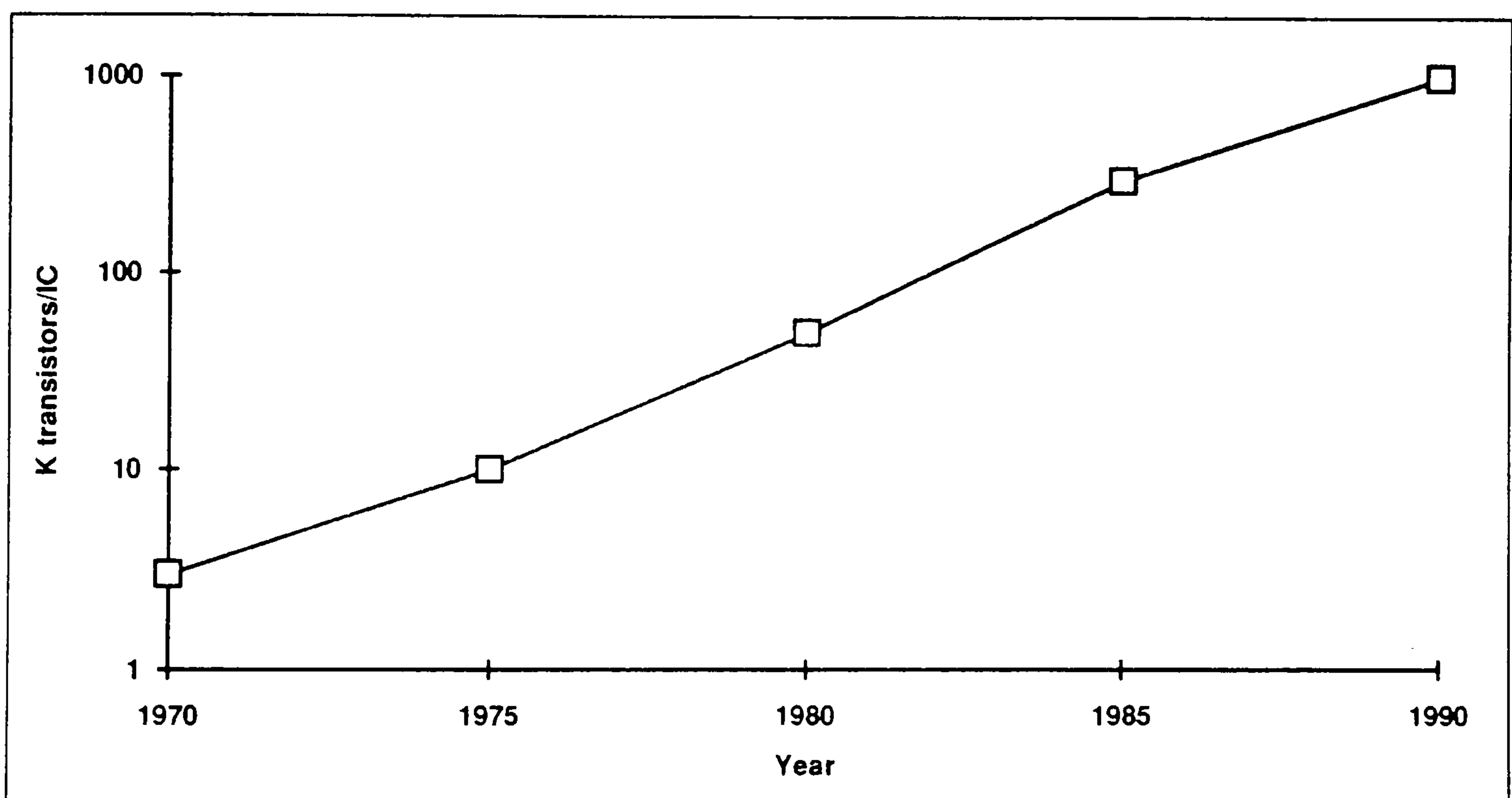


Figure 1: The increasing IC circuit density [Sed92]

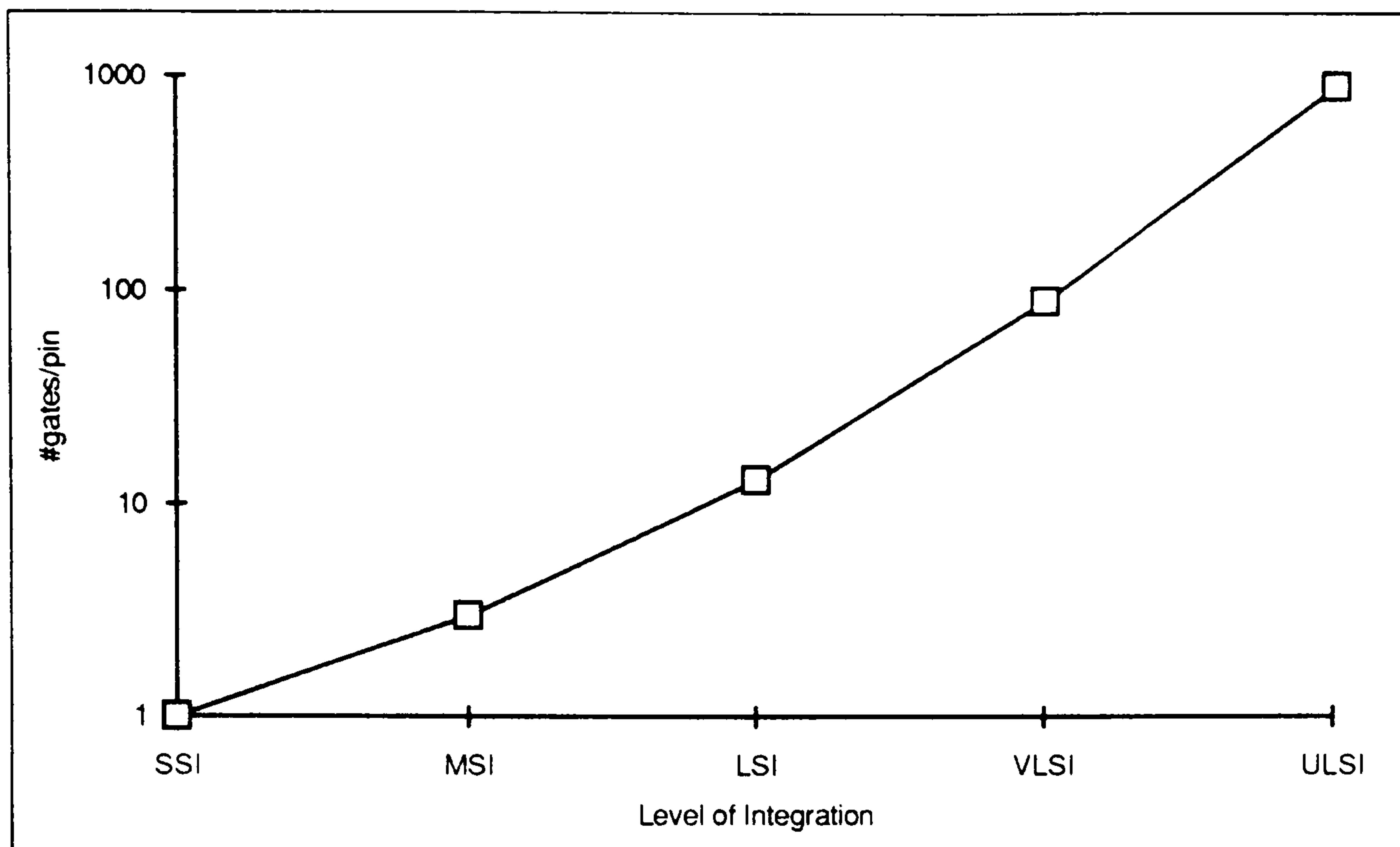


Figure 2: The increasing gate per pin ratio [Sed92]

But as the test of an IC is applied through the pins, the testing cost does not remain unchanged, but increases for higher levels of integration. The expenses which are related to test cost occur during the development for generating a test, which enables to detect as many defects as possible, and during production for applying the generated test to the ICs. The complexity of test generation increases linearly with the gate complexity of the IC but exponentially with the accessibility of the gates. The accessibility of the gates is a function of the gates per pin ratio. These dependencies lead to an exponential increase of the test generation costs over time, if the methods for test generation remain unchanged. A test is built upon test patterns which are applied to the IC through the pins. The cost for test application is a function of the number of test patterns to be applied. The number of test patterns again increases linearly with the number of gates and exponentially with the gate per pin ratio. These reasons have caused the significant attention been paid to the testing issue of the IC technology since the beginning of the 80's [Wil83].

The only way to keep testing cost in an acceptable bound is to develop new methods and equipment, which facilitate test generation and test application. This includes the development of new high-speed VLSI test equipment, tools for automatic test pattern generation (ATPG) and design methodologies for facilitating the test generation and test application tasks (design-for-testability, DFT).

Most companies are aware of this testing problem and they have accepted some of the DFT methodologies as well as ATPG systems or new test equipment. But all of these methods have certain economic trade-offs. For example, a DFT method facilitates the test and therefore reduces the related costs, but it may increase the production cost due to an increase in the die size and a reduced production yield. ATPG systems have to be purchased, and the price of such a system is still more than $50,000. And VLSI test equipment has become extremely expensive with the increasing performance requirements. The price for high-end testers are $5,000,000 or more.

The many options for performing the test of an IC allow to create many different test scenarios, which assure a high quality of the IC. The test scenarios are called *testing strategies*. An example for a test strategy is to design an IC such that an automatic test pattern generation system can be used and a VLSI test equipment is applied to perform the production test. A test strategy can be selected from those which fulfil technical requirements. The selection criteria can range from a subjective assessment of the test strategies by the persons who are involved in the implementation of it, to an economic evaluation of the test strategy. The author proposes to use economic evaluations for test strategy planning, because this method converts all impacts of a test strategy to a common reference point, which is the cost in £, $, ECU, DM, or any other currency. This is the only selection method which is fully objective. Other methods, such as the assessment by experienced test engineers or design engineers, or the evaluation of some costing parameters, which are based on different measure units, include a certain degree of subjectivity from the person who is evaluating the test strategy. Therefore the selected test strategy would be highly dependent on the person doing the evaluation.

This selection procedure becomes even more complicated when integrating the IC test strategy into a test strategy for the entire system. An electronic system is typically built upon printed circuit boards (PCB). In the past the testing problem of a PCB was not as critical as for ICs, because accessibility to internal nodes was possible by using a special class of test equipment, in-circuit tester. But today's surface mount technology for PCBs

uses wire separation of less than 100μm, double side mounted boards, and more than 2,000 internal nodes per board to access. The access to the internal nodes becomes extremely difficult in many cases or even no longer possible. This fact was the main motivation for developing the design methodology "boundary scan", which is standardised under IEEE, known as the 1149.1 standard. This methodology allows to access the pins of an IC through a serial shift register, which enables to get access to the internal board nodes.

Boundary scan is an integrated design methodology, which can be used not only for board testing, but also for troubleshooting at system level and in the field. This shows the complexity of evaluating a test strategy which includes boundary scan. This test strategy affects the IC design and production, the board production and test, the system test and the field test. The evaluation of the economics of such a test strategy includes almost all cost areas of the system during its life cycle.

For this reason the economics modelling techniques, which have been developed by the author, consider the whole life cycle of the electronic system. This is the only way to take into account all impacts of a test strategy for an objective evaluation.

1.2. Testing, Error Free Design and Production

One test strategy option is a strategy of not testing at all. This test strategy could be economical if it was possible to develop and produce the products perfectly. This involves a perfect design, perfect materials, a perfect manufacturing process and no ageing of the product [Sed92]. The term *perfect* is used here in the sense of "free of errors or defects". But, since perfect products are unlikely in the near future, especially under the aspect of ever increasing complexity, the major remaining option is testing a product more or less in all phases of its life cycle. The "more-or-less" forms the test strategy, which is defined by the criteria in figure 3.

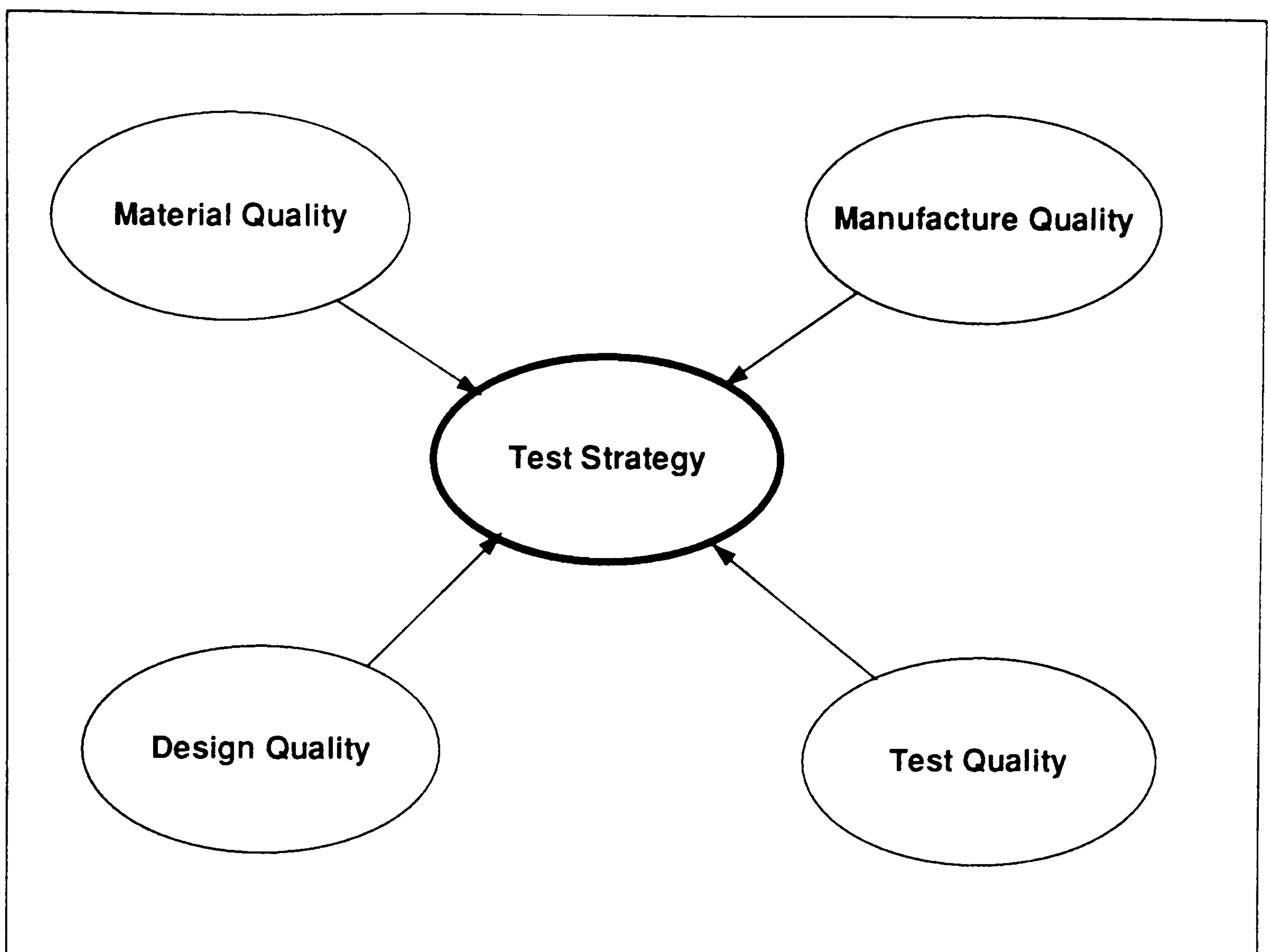


Figure 3: The quality options for test strategies

The *material quality* affects the production yield of the device and therefore the required test quality to achieve a certain quality level.

In the same way the *manufacturing quality* impacts the yield and therefore the test quality requirements.

The *test quality* defines what percentage of all possible defects can be detected by a test application. This test quality depends on the test generation capabilities, the test equipment capabilities and the effort which is spent in test generation and test application.

Under testing aspects the *design quality* is the degree of "design-for-testability" which is implemented in the design. This mainly affects the test capabilities and the economics of achieving a certain level of test quality.

The linkage of these factors into a test strategy shows that test strategy planning is a

typical concurrent engineering task. A test strategy influences nearly all engineering aspects of the product, and therefore they should be taken into account.

1.3. Structure of the Thesis

This thesis describes economics driven methods for test strategy planning, which take all these factors into account. The test strategy planning system, which has been developed by the author, comprises two test strategy planners. ECOTEST is a test strategy planner, which is used for ASICs, and ECOvbs is a test strategy planner for VLSI based systems. These tools can be used in combination in order to derive the most economic test strategy for VLSI based systems. The tools take into account the concurrent engineering aspects, which have just been described. This is a novel approach for test strategy planning, which integrates the aspects of design, test and manufacture during the life cycle of a system.

The rest of the thesis is organised as follows: Chapter 2 will present an overview on test methods. The elements of test methods will be described. The test methods will be classified and the important classes of test methods will be described.

Chapter 3 describes the process of test strategy planning and outlines a range of systems which are used for test strategy planning. The scope of test strategies and test strategy planning will be discussed.

Chapter 4 discusses the economics modelling aspect of test economics. It describes various cost modelling techniques, and it will present the life cycle test economics model which has been developed by the author.

Chapter 5 describes ECOTEST, the test strategy planner for ASICs. The philosophy of ECOTEST is discussed. The usage of ECOTEST in test engineering is described and discussed, the EVEREST test strategy planner [Dis92], which is the basis of ECOTEST is outlined and the enhancements of ECOTEST against the EVEREST test strategy planner will be described and discussed.

Chapter 6 describes a range of methods to study the impact of the inaccuracy of economic parameters on the resulting cost. The parameters are generally studied in terms of their impact on the sensitivity of the total cost. The description of the problem is presented, the need and the gain of this work are discussed, and the different applications of the sensitivity analysis are introduced. The author describes a method of analysing the variation of all parameters at the same time and presents three applications of sensitivity analysis.

In chapter 7, the author first discusses the philosophy of ECOvbs, the test strategy planner for VLSI based systems, and the arising needs of such a system in industry. An overview of the architecture of ECOvbs will be given and its components will be described in detail.

Chapter 8 will present and discuss the results from analyses perfromed using the test economics models which are described in chapter 4, using ECOTEST and using ECOvbs.

The life cycle cost models described in chapter 4 were used to study boundary scan test strategies. ECOTEST was used for two types of experiments. In the first set of experiments, the author used the system for several large industrial designs in order to prove its applicability. The second experiment is related to performing test strategy planning with inaccurate data by using the sensitivity analysis techniques which are described in chapter 6. The last section of this chapter will present test strategy planning results for a large computer board by using ECOvbs.

In chapter 9 the author draws the conclusions on his thesis. This includes a summary of the work and several proposals for future work in this field. This is mainly related to further improvements of ECOTEST, such as test partitioning and an improved accessibility analysis, the full integration of field test aspects into ECOvbs and some ideas on the automatic generation of life cycle test strategies.

Chapter 2
Test Methods

2.1. Introduction

The test of VLSI based systems aims to ensure a given quality for the system under test. This quality level is typically defined in the early product planning phase, and it is driven by cost analysis, market requirements, laws or by marketing or company strategies. A test can be performed in several phases and levels of the design and manufacturing process of the system. To achieve the required quality level, there are many different options:

- The quality level of the material used:

  The complexity of material can range from raw material like solder to complex devices like VLSI components.

- The manufacture quality:

  The manufacture quality impacts the defect rates of all parts created in the manufacture process plus new defects introduced into the material during the manufacture process. An example for these new defects may be a destruction of a chip by too high temperatures during the soldering process.

- The test quality:

  The quality of a test is defined by the relation of the number of possible faults of the device under test. The test quality depends mainly on the test method and the characteristics of the device under test.

- The test level:

  A test can be performed at many levels of manufacture. The author defines three test levels as follows:

  - The **component test** is applied to the components which are assembled into a system. These are ICs, bare boards, or discrete elements. A component test may be applied after the production of the components at the suppliers

facilities (production test) or at the customers facilities after the delivery of the components (income test).

- The **board test** is applied to sub assemblies like printed circuit boards (PCBs) or multi chip modules (MCMs).
- The **system test** is applied to the whole system in order to guarantee the function of the whole system.

Component and board tests are options, which are used in addition to the system test in order to make testing more economical.

- The test phase:

  Quality assurance is usually defined in all phases of a product's life cycle. This option is needed to detect occurring faults as early as possible, which is in many cases an economical solution for the quality problem. Quality assurance may be document review, computer simulation of the design model, a prototype verification or the test of a manufactured device.

- The test strategy:

  A test strategy is defined as the sequence of test activities at several life cycle phases and levels of manufacture as described above. This definition of the term *test strategy* is given in [Ben89].

This chapter will describe the state-of-the-art test methods for VLSI circuits and complex boards and systems. A test method consists of three components:

The *test generation* is the process of generating stimuli for a circuit which will demonstrate its correct operation [Wil83]. Various methods exist for test generation. Which one is used depends on various aspects, such as the design style or the availability of tools.

The *test application* is actual execution of the test for a given device. Test application can be performed manually or automatically by using certain tools or equipment for stimulating the device under test and measuring the results. The equipment

can be incorporated into the circuit, which means that the circuit is self testing.

The *Design for Testability (DFT) methods* are design styles and techniques, which are more or less integrated into the functional design, and which facilitate the test generation and test application, and therefore reduce the accompanied costs.

A test method always consists of these three components. Even if no design methods are implemented for facilitating the test, this is considered as a the DFT method *no DFT*. A test method can be applied to parts of the design at several levels of assembly. The definition, of which part of the design which test method is applied at which level, is called *the test strategy*.

The following sections will present various DFT methods, test generation methods and test application methods. The methods will be presented by categorising them, by describing briefly the technique, and by describing its economical impacts.

2.2. Description and Classification of Test Methods

2.2.1. Design for Testability Methods for VLSI Devices

The facilities of VLSI led to heterogeneous and multi functional chips with increasing complex functions [Zhu86]. Attributes of these circuits are high gates-per-pin rates and highly embedded functional blocks. By being embedded the blocks' accessibility is poor. Due to the complexity of the functional blocks the controllability and observability of internal nodes becomes more difficult. For testing, access to the inner circuit is essential. So the cost for testing is increasing rapidly as the accessibility is decreasing. This fact is confirmed by the conventional design process, which separates the design- and test-phases. The need for considering test aspects during the design phase in order to decrease test cost becomes evident.

So techniques, called **Design for Testability** (DFT), were developed to assure the testability of VLSI circuits. The main purposes of DFT are :

- The generation of high-quality tests is enabled.

Circuits are designed in such a way that an ATPG system can generate high-quality tests sets automatically. Rules to be followed are checked by a DFT-rule checker accompanying the design phase.

- The test itself becomes cheaper.
  DFT techniques reduce the test set length by reducing the number of steps for controlling and observing inner nodes. Also the usage of cheaper test equipment is enabled by providing the circuit with self-test techniques.

Some of the techniques were developed for specific functions [Zhu88], some of them can be applied [Wil83] generally. All techniques affect the design- and test-process in different ways.

DFT methods in general can be defined as methods which aim in assuring high-quality products by facilitating the test. The DFT methods studied here are specific for VLSI products. This overview is divided into two parts :

- The general methods can be applied to every type of circuit. Mainly design methods are described, which assure economic testing of VLSI products.

- For some specific circuit structures, especially regular structures, there are design-specific methods existing. Normally these structures are functionally described, and so the classic methods for test pattern generation cannot be used, because they are based on the logic structure. But because of the regularity of these structures, formal methods to calculate high-quality test pattern sets were derived.

DFT has different aspects: It can be seen as a philosophy as well for designers as for the management. DFT must be considered in all design phases (functional design, logic design, layout) by following specific rules to make a product testable. The DFT methods can be partitioned into two groups :

Ad-Hoc techniques can be applied quickly and easily. They are well suited for application after the logic design. Because they are based on heuristics, no reliable prediction of testability improvement is possible. These methods are mostly used to make an existing design more testable.

Structured methods must be considered for the whole design. Therefore, they must be applied from the beginning of the design. These methods are supported by CAD tools. They usually decrease design times, and good testability of the circuit can be assured. The expression "Design-For-Testability" fits much better for the structured methods, because "Design-For-Testability" means designing for testability rather than redesigning for testability.

Major testability aims of the methods are:

- Partitioning:
  The effort for testing increases exponentially with the number of gates. Therefore the effort can be reduced by dividing a circuit into smaller sub circuits for testing purposes. Most of the Ad-Hoc techniques are based on this principle.

- Increase of Controllability and Observability:
  Especially for deeply sequential circuits controllability and observability of inner nodes is very difficult. High controllability of control lines is important for the testability of the controlled logic.

- Universal Testing:
  Designs based on regular structures (e.g. PLAs) enable in some cases function-independent testing. These methods are described in 3.2 and 5.2.

- Dual Mode Design:
  The circuit is provided by a mode pin to switch between test-mode and system-mode.

- Scan Design Methods:

  These methods enable access to inner nodes by using storage elements. The aims are an increase of the testability and a decrease of the sequential depth for testing. By using a full scan design, sequential circuits can be modelled as combinational circuits (sequential depth equals zero). Then automatic test pattern generation becomes much cheaper.

- Self Test:

  Self testing circuits are very efficient in terms of reduced test application cost. Also the costs for test pattern generation can be very low by testing with random patterns or testing exhaustively ([McC86]).

- Compaction:

  The responses at the outputs of the circuit under test are compacted internally. Only a signature is evaluated.

2.2.1.1. Ad-Hoc Methods

This section will give an overview on the state-of-the-art Ad-Hoc techniques. Most of them increase the testability for general logic designs.

1) Maximisation of Controllability and Observability

In general an increase of the controllability and observability of inner nodes of the circuit under test leads to an increase of testability of the whole circuit. Especially the controllability of control nodes, like clocks, set/reset lines, data select, enable/hold of microprocessors, enable and read/write of memories, and control and address lines of bus structures, sensitises the testability of the whole circuit. Also test points should be included to the following nodes to increase the observability :

buried control lines, outputs of memory elements, outputs of data-funnelling elements,

redundant nodes, nodes with high fanout, global feedback paths.

2) Synchronisation of Storage Elements

For high-frequency circuits, problems like races can occur, especially for faulty circuits. To avoid these problems, memory elements should be clocked. For testing, all memory elements should be clocked by the same source, and the clock line should be directly controllable.

3) Initialisation

To reduce the test length, it should be easy to bring the memory elements into a defined state. Therefore all memory elements should be provided with set- and reset-inputs and these inputs should be easy to control.

4) Partitioning

As already mentioned, the effort of testing and test generation is exponentially related to circuit complexity. So partitioning of circuits into smaller sub circuits for testing can increase the testability.

2.2.1.2. Scan Structure Methods

Scan structure methods are considered during the whole logic design phase. The main purpose of scan structures is to make memory elements directly accessible or to control and observe inner nodes by direct access. Typically, scan structures operate in two modes. In system mode, the circuit operates to fulfil its normal function. In test mode, the scan elements can be observed and controlled directly. The type of access depends on the scan structure type.

The scan structures can be classified by four properties ([Tri82]) :

- Degree of Integration :

  If a scan element is integrated into the normal function, it is an internal scan

structure. If extra scan elements are used to access internal nodes, we have an external scan structure. Internal scan structures are mostly used to reduce the sequential depth of the circuitry, where external scan structures are used for partitioning and accessibility purposes.

- Completeness of Scan Structures :

  If all memory elements of a circuit are scan elements, and these elements are integrated into a scan-path-chain, we call this structure full scan path. A combinational model of the circuit can then be derived, to generate test patterns by an ATPG system for combinational logic. To reduce the silicon overhead, the partial scan path is used, where only a subset of memory elements are accessible via a scan path. These elements are selected in order to reduce the sequential depth and to make nodes with low accessibility more accessible.

- Type of access :

  The scan elements can be accessed serially or in parallel. For serial access, the elements are interconnected into one shift register. For parallel access, the scan elements are grouped. For every group, direct access is then possible. The groups can be selected via a decoder.

- Scannability of Scan Structures :

  The author distinguishes between passive scan structures, which allow only observation (scanning) of the scan elements, and active scan structures, which allow both, observation and control. Passive scan structures are typically external, and they are used for diagnostic purposes. While running in system mode, the circuit can be monitored.

By this classification, we can derive the following classification tree:

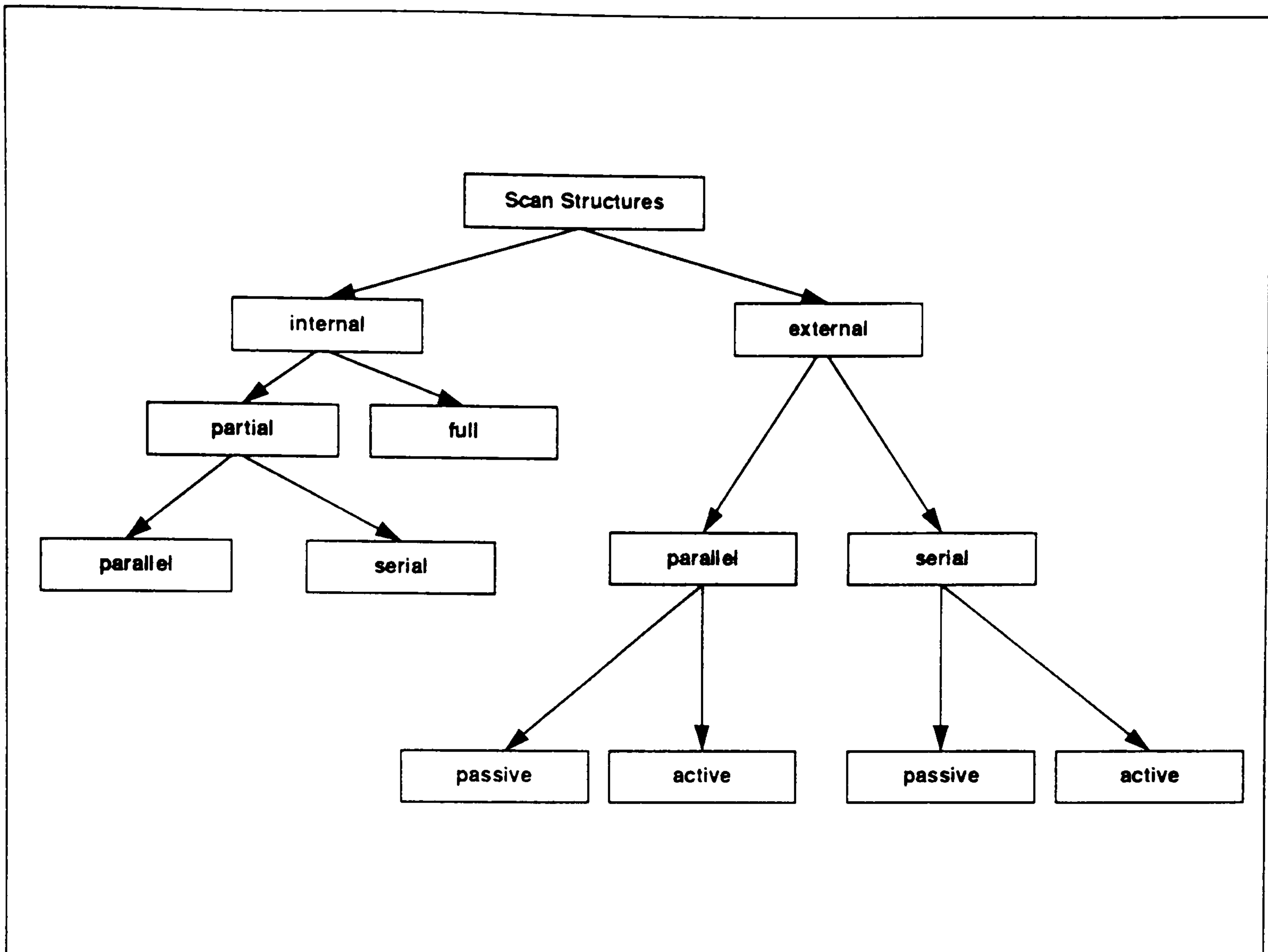


Figure 4: Classification of Scan Structures

2.2.1.3. Self Test Methods

The main purpose of self test methods is the reduction of test application cost and test generation cost by the following reasons :

- Test equipment costs increase dramatically due to the high pin count and performance requirements of VLSI circuits. Also the lifetime of high-end testers becomes shorter, because requirements like maximum pin number or maximum operating frequency are increasing very fast. Self test methods enable the usage of cheap testers, because the function of the tester is only to control the test.
- The complexity of VLSI based systems (VBSs) makes in-field tests costly. Self test methods support the diagnosis, and so the in-field testing of components in assembled systems becomes cheaper.
- Self test enable exhaustive testing and random testing ([McC86]). This reduces the test generation cost, because the test sets must not be generated

deterministically. For random testing, the fault coverage can be predicted by using testability measures (see 3.1.2). Costly fault simulation can be omitted. For exhaustive testing, the fault coverage is always 100%.

- Self test structures can be applied at system speed. Therefore a self test is rather dynamic than static. That means, that dynamic failures are covered by self tests.

A test method must consist of a strategy for generating input stimuli to be applied, evaluating the responses of the device under test and an implementation mechanism ([McC86]). Every self-test-method must fulfil these requirements. Some methods accomplish all requirements, some of them support self testing only partly. A complete self test must consist of all three elements. The most important self test methods are listed in table 1.

Structures for Stimuli Generation	linear feed back shift register (LFSR), non-linear shift register (NFSR), ROM, counters
Structures for Output Response Analysis	parallel signature analysis, serial signature analysis
Structures for Self Test Architectures	built in logic block observer (BILBO), in system at speed test (ISAST), circular self test path (CSTP)

Table 1: Self test methods

2.2.1.4. Design Specific Methods

General testability enhancing methods are not always the best option for a specific functional block. Building blocks such as RAMs have a regular structure with specific failure mechanisms. Using the scan approach for example, may incur a large area overhead. On the other hand, there are many algorithms which test for specific fault models, and the test strategy for the RAM can be based on these, thus taking advantage of the regular structure. These observations apply mostly to embedded RAMs, where accessibility often is a problem.

Another example of a regular structure is the PLA. In order to obtain a high fault coverage, cross point faults need to be considered in addition to stuck-at faults. Test

generation algorithms need to take these specific fault models into account. Many general purpose test algorithms, as well as random test patterns, do not efficiently test PLAs due to the high fan-out, fan-in and redundancy. Often deterministic and random techniques are combined to arrive at a high fault cover test set within a reasonable time and cost [Eic80]. There are also test generation algorithms for PLAs that arrive at a function independent test set. These universal tests eliminate the test pattern generation costs. However, modifications to the design are needed in order to apply the patterns and observe the responses.

The test methods can be categorised according to the way they achieve their objective. For example, in the PLA case, there are methods that use random patterns, self test methods, parity checking methods and partitioning methods. These classifications are intended only to make the study of test methods easier, and in fact, certain methods may belong to more than one category. The classification of test methods is a continuous process. However, methods which are obviously not suitable for an application or which are only minor variations on an existing method should be considered carefully before inclusion: the number of possible combinations increases very fast as the number of available test methods increases, and run time should be taken into account. Hereinafter, a selection of the most commonly used DFT methods for PLAs and RAMs are described. The selection demonstrates the classification process and is not intended as a complete set.

The Saluja method [Sal85] uses a shift register for partitioning. The shift register is used to control the product terms. However, in this case product terms are partitioned into groups so that they can be tested in parallel. Therefore, the length of the shift register for selecting product lines is shortened from the number of product lines to the number of groups. The number of test patterns is also reduced.

The Fujiwara method [Fuj81] uses a parity strategy. Adding a product line with its associated connections will ensure an odd (or even) number of connections on every bit

line. A cross point fault in the AND array will change the parity of a bit line. Faults in the OR array can similarly be detected by adding an extra output line and a parity checker. In order to achieve this, control of the product lines is needed, which is achieved by the use of a shift register. An input decoder is also used, to ensure control of the bit lines. The test used is a universal test set (low or nil test generation costs), which is function independent, but requires vectors to be either pre stored or externally generated.

The Treuer method [Tre85] is also a parity checking method, but it eliminates the requirement for a stored set of vectors. It has a very high fault coverage, covering all single and most multiple cross point faults. The test vectors are self generated, and parity signals are accumulated in a parity counter, the value of which is checked at specific times.

The Daehn method [Dae81] is another self test method, which is based on partitioning. BILBOs are used to partition the circuit, and are placed after input decoders, the AND array and the OR array.

RAM test methods exploit the regular structure of the block in order to test for the groups of faults specific to memories, due to their regularity and high density. One way to test an embedded RAM is to implement test structures which ease the application of a specific memory test algorithm. One example of this type of method is the implementation of the March test, a simple, widely used memory test algorithm [Dea89]. The modified hardware required would be shift registers for data in and a comparator register, as well as a modification to the address register to also act as an up/down counter. The March test can then be applied and the responses evaluated using an N bit comparator. For a memory with an N bit word, the test length would be 2m+8Nm, m being the number of words. This algorithm tests to 100% for single stuck at faults (ssa), but does not perform very well for pattern sensitive faults. The addition of simple control and test generation circuitry would make the memory self testable. It is relatively simple to modify the shift register/counter/comparator arrangement to implement more complex

algorithms. The test generation effort is almost nil.

Another group of methods uses pseudo random patterns for self test of RAMs. The Illman [Ill86] method converts the data input and address registers into maximum length LFSRs and randomly sets the write enable. The application of patterns is repeated until the required fault cover (determined probabilistically) is achieved. The requirements for the method are: the data input register as well as the address register can also act as a shift register and a maximum length LFSR with preset. A data output register is required that can act as a MISR with preset. A hard wired word size comparator is necessary, as well as a 2 to 1 MUX to select between functional and test mode. LFSRs are needed to produce the random write enable signals, as well as the set of initial seeds. A set of counters is also needed in order to scan in the new seed, keep a count of the number of tests applied and ensure that all cell addresses have been accessed. Also a small amount of random logic is required to handle the clock control of the registers and control each test stage. The number of test patterns depends on the number of repetitions of the test.

2.2.2. Test Generation Methods and Fault Simulation Methods

To apply a test, test input stimuli and output responses must be derived as a precondition. These test patterns must be evaluated in respect to their fault coverage.

Test patterns are generated for the detection of failures. These failures are represented by a fault model. Algorithms were derived to generate a test pattern set automatically (Automatic Test Pattern Generation, ATPG) and to evaluate the test pattern set by simulating the faulty circuit. ATPG is typically supported by a testability analysis. Beside guiding the decision-making-process of the ATPG testability analysis is also used for improving the testability of circuits by applying Ad-Hoc DFT strategies. In the following the author gives a brief overview on the algorithms.

2.2.2.1. Test Pattern Generation Methods

The purpose of ATPG algorithms is to determine a test vector, which detects a given

fault ([DiG89]), or to prove the given fault as a redundant fault. For most algorithms, a test vector for a given fault must satisfy two objectives ([DiG89]) :

- The first objective is to activate the fault.
- The second objective is to observe the effect of the fault at one of the circuit outputs.

The algorithms can be divided into four categories ([DiG89]):

- Tabular Methods:

  For each possible input combination output values for the fault-free and each faulty circuit are calculated. This approach is used in [Hwa86] and [Gho89]. In [Dic87] it is shown, that cases can be constructed, where the algorithm fails due to complexity problems.
- Algebraic Methods:

  The mostly used algorithm of this class is based upon the Boolean difference method ([Sel68], [Lar89]).
- Functional Methods:

  Tests are derived from the functional description of the circuit ([Hey89], [Su84]). Within a class of realisations, the test generation is possible for stuck-at-faults ([Poa63]).
- Gate Level Algorithms:

  Algorithms based on the gate level description of the circuit generate test vectors typically for every single stuck-at fault. The basic algorithm is the D-algorithm ([Rot66]). Based on the D-algorithm, the algorithms PODEM ([Goe81]), FAN ([Fuj83]) and SOCRATES ([Sch88]) were developed to improve the efficiency for very complex circuits. All these algorithms were developed for combinational circuits. For sequential circuits, the algorithms were extended to solve the problem, that the circuit responses depend on the circuit state in addition to the input stimuli ([Mar86]). This matter of fact leads to a new complexity dimension for generating test patterns, because the number of states can increase

exponentially with the number of memory elements.

2.2.2.2. Fault Simulation

Fault simulation tools are used together with ATPG to speed up the TPG process and to reduce the test set length (without fault simulation a test pattern must be generated for every testable fault). Existing test sets (e.g. the functional test patterns used for logic simulation) can be evaluated by using a fault simulator. Several approaches ([Arm72], [Sch84], [Rog85]) were presented for general circuits, and for scan-based circuits Parallel-Pattern-Single-Fault-Propagation ([Wai85]), Critical-Path-Tracing ([Abr84]) or Fast-Fault-Simulation ([Ant87]) led to high efficiency in terms of computing time.

2.2.2.3. Testability Analysis

Testability analysis is a method of grading the testability of a circuit by measuring the controllability and observability of the internal nodes. As mentioned earlier, especially for sequential VLSIs the results of ATPG and fault simulation are very poor. The need of DFT in order to improve the testability is evident. One approach is to apply structured DFT methods like the internal scan path. For Ad-Hoc DFT strategies information is needed on the location of testability problems. Therefore testability analysis tools were introduced, which are based on testability measures ([Gol80], [Gra79], [Kov81], [Ben80], [Rat82], [Tri84], [Brg84]). Due to neglecting signal correlations caused by reconvergent fanouts, the results of the testability analysis do not always give the exact information about circuit regions with poor testability ([Agr82]).

Testability measures are also used to improve the search heuristics within the ATPG.

2.2.3. Test Application Methods

In this section the author will give an overview on the most important test application methods and their use. The description will include the test application procedure, the required DFT methods, the test equipment and the fault coverage which can be achieved.

The test application methods are grouped into component tests, board tests and system tests.

2.2.3.1. Component Test

A component can be tested by an **income test** in the same way as it should be tested by the component supplier by the **production test**. This might be a complex VLSI test for VLSI components or a test of the specification for simple passive components like resistors or capacitors. In the production phase the test is usually applied to each component, whereas an income test strategy mostly applies the test only to a sample per delivered lot. The sample size depends on the delivery quality and the deviation from this quality and may range from 0 (no incoming test is performed) to all incoming components.

2.2.3.2. Bare Board Test

The bare board test tests the wiring of an unassembled PCB. It uses special bare board test equipment, which includes capabilities to contact the board, to generate a test program automatically by analysing a golden device, and to perform the test. The fault coverage is 100% of the opens and shorts of the wiring on the bare board.

2.2.3.3. Board Test

In contrast to the component test, the task of board test is not only a go/no go test but also a fault diagnosis in terms of fault location for repair. This means, that the quality of a test is described by the fault coverage and the fault diagnosis capability.

By a **visual inspection**, a board is inspected visually by a person for gross defects such as solder splashes. The degree of inspection depends on the type of test following the visual inspection. This test method is very cheap but has a low fault coverage

The **manufacture defect analyser (MDA)** is an In-circuit tester that examines the board construction. Normally power is not applied to the board to be tested. As a DFT

requirement, test pads are needed for all nodes to be adapted. The test equipment is a special MDA equipment, which requires a separate fixture for each device under test. Only passive elements and the board construction can be tested. This includes the test of resistors, capacitors and the solder defects such as shorts or some opens. The fault diagnosis is straightforward, i.e. the fault can be located in parallel with the identification of the fault. Multiple fault identification can be performed in one test run.

An **In-circuit test** (ICT) examines construction of a board by isolating and examining each component on the board as an independent entity. Therefore the board must be accessed at each electrical node on board level through a fixture called "bed-of-nails". Each component on the board is tested in power-on condition by both stimulating the inputs and evaluating the outputs of the component through the bed-of-nails fixture. This requires back driving the outputs of the components, which are connected with the inputs of the component to be tested. The following DFT methods are required:

- Test pads must be designed for all nodes to be adapted.
- It must be assured, that back driving does not destroy the components.
- Every component must be separately testable.

The test equipment is a special ICT equipment. A special purpose fixture for every board type to be tested is needed. If the DFT requirements are fulfilled, the board level fault coverage of static faults can be 100%. Dynamic faults cannot be covered, if the ICT is performed statically. A dynamic In-circuit test requires more DFT and a more expensive In-circuit tester. This type of test is able to cover some of the dynamic faults. The coverage of analog defects depends on the capabilities of the ICT equipment. The fault diagnosis is straightforward. Multiple fault identification can be performed in one test run.

A **functional test** accesses the board under test (BUT) only by the edge connectors. A functional tester emulates the environment of the board, i.e. the system in which the board will be used. Typical (functional) test stimuli are applied, and the board's output responses are evaluated. If the BUT fails the test, guided probe or fault dictionary

techniques are used to locate the fault. The generation of test data is done by using a simulation system. The design must be logically partitioned, so that the test units for which test data should be applied, can be easily accessed from the edge connector. In addition, all chip level DFT requirements, such as the ability of initialisation, must be fulfilled. Testability is an essential requirement for functional board testing. The functional board test uses a special board tester, which consists of configurable digital and analog signal sources and measuring devices. For the application of guided probe techniques, probe sensors are needed. The achievable fault coverage strongly depends on the DFT methods applied. Dynamic faults can only be covered, if the test system is capable for performance tests at the required speed. Analog parts can only be tested, if a clear separation from the digital parts is possible. Multiple fault detection and isolation in one run is not normally possible.

A **combinational test** combines in-circuit test and functional test. The functional test capabilities are combined with the in-circuit test capabilities and its diagnostic capabilities. Depending on what type of test to run, the DFT requirements are a combination of the ICT and functional test requirements. A special combinational test equipment is needed. The fault coverage capabilities are a combination of the ICT and functional test fault coverage. The diagnosis capability is equivalent to the ICT capability.

An **emulation test** is a special variation of the functional test, where specific board components are replaced by an adaptation to the test equipment. The function of the replaced component is emulated by the test system. Typical components which are emulated are:

- Processors: The processor is replaced by a processor oriented device (POD), which is capable of imitating the processor's function and timing.
- Memories: The memory on board is replaced by the tester's memory. This allows to control the processor's action directly by the test system.
- Busses: The processor on board is controlled via the main bus of the board. This

means, that no board component needs to be replaced by an emulation adapter. An emulation test is used for real time tests of complex processor boards. The DFT requirements are the same as for the functional test. A functional tester with additional processor specific emulation units can be used as test system. The emulation test is applied mainly for the dynamic test of the processor and its periphery. Therefore the fault coverage for dynamic faults and the diagnostic capabilities for dynamic faults are high. For covering static functional faults and construction faults, an additional static test should be applied.

The **memory test** is a special feature of testing, which enables to apply algorithmic test patterns rather than deterministic test patterns. The number of test patterns for memory test are typically very high, but the test patterns are of algorithmic nature and can therefore easily generated online. This type of generation reduces the test pattern memory requirements of the test system. The memories on board must be directly accessible. Any functional test system with online test data generation feature may be used. The fault coverage is 100% for memories or other devices, which can be tested by algorithmic test data.

A **burn-in test** aims to force early life failures by accelerating the time of operation. This is achieved by overdriving parameters, which cause the early life failures, such as temperature or voltage. No special DFT is required. A burn-in test is executed in combination with any of the test equipment mentioned here. If environmental parameters such as the temperature should be overdriven, an equipment to overdrive these parameters, e.g. a climate chamber, is needed. The type of faults covered depends on the test equipment used. The percentage of early life failures forced depends on the early life behaviour of the device under test and the acceleration factor of the burn-in test.

A **hot bed tester** is a one-of-a-kind tester used to verify that the board under test actually operates in the final product [Pyn86]. Normally a hot bed tester exists of the entire product except the board to test. The board to test is inserted into the hot bed tester, which can also be called *the reference system*. The test consists of evaluating the correct operation of the whole product. No DFT is required. The fault coverage depends

on the percentage of functions being tested by the product's test operation. This type of test covers best dynamic faults. The diagnosis is normally performed manually by skilled technicians, who have detailed knowledge about the operation of the board.

2.2.3.4. System Test

The **installation test** is a procedure where the completely assembled product (the system) is operated under field conditions. No special test equipment is required. The only requirement is the ability to provide field conditions. All type of faults can be covered. But normally the installation test aims in detecting dynamic functional faults and faults caused by the final assembly. The faults coverage of manufacture faults is normally not known, because a fault simulation of the test program would be too expensive. The diagnosis of faults is usually done manually by the support of special diagnosis software.

A **benchtop tester** is a small test system which provides analog, digital functional and in-circuit test capabilities. The test system is not optimised for throughput, so the setup times and test times are long compared to fully automatic test systems, such as an In-circuit tester or a functional tester. The benchtop tester consists of a small computer, e.g. a PC, a general purpose card cage and a variety of stimulus and measurement boards, where the user can select from, in order to mix and match the test system to meet the special test requirements. The fault coverage depends on the quality of the test mix. Nearly any type of faults can be detected and located by this test system, if enough time is given.

2.3. Economical Impact

It is becoming widely accepted that provision for test in the design stage is not only desirable, but in many cases essential. This section will attempt to address the economic aspects of DFT. These are taken into account in the economics model developed.

The advantages of incorporating DFT into a design are well known. One of the major ones is the fact that test generation is made easier and test application times become

shorter. Test generation for a large design can be a long and expensive process, especially when a very high level of fault cover is required. Using appropriate design for test techniques can alleviate this problem. Not only is test generation faster (and therefore cheaper), but often a higher level of fault cover can be achieved. In the case of self test techniques like signature analysis, the cost is reduced to simply predicting the correct signature.

The use of self test techniques also reduces the test application costs by removing the need for complex (and expensive) automatic test equipment (ATE). However, savings in test application costs can be made in more indirect ways. For example, the ability to use an inexpensive ATE for a project may free a heavily used ATE for use by other projects. The net effect is a speeding up of the test process, and possibly earlier shipping of the product. In a competitive market, this is a distinct advantage.

One reason that DFT methods may not be chosen is the possible increase in design time due to the introduction of new techniques. To solve this problem, the test support system being developed will provide the designer with data on the chosen test methods, so that implementation can be made easier. In any case, it should be born in mind that the design time increase can be minimised if the CAD system supports automatic overlay of test structures. One more point is that making the investment in terms of design time may well lead to considerable savings in terms of overall development time, as expensive redesign due to inadequate testability provision can be eliminated. In some cases, it has been found that the structured design style required to implement some test methods led to an overall reduction in design time.

The provision for testability also leads to the manufacture of high quality products. It is cheaper to discover faults in the early stages of a product's life cycle. The increased reliability of high quality products has definite advantages for the manufacturer's reputation.

Conversely, DFT structures often involve extra silicon. This has been a major reason why

DFT methods were often discounted as uneconomical. However, the only way to decide on the suitability of a DFT technique is to make a careful economic evaluation of all the relevant factors. The economics model has been developed for that purpose. Taking the concept one step further, the economic model can be used as a method of choosing which mix of DFT techniques to use in a particular situation.

The economical impact on the board and the system of various test application methods, which are presented here, is shown in table 2. It presents a qualitative cost impact on the various design and production phases of the system and on the board yield. The cost impact is rated as follows:

- ++ means high additional costs
- + means moderate additional costs
- o means no cost impact
- - means small cost savings
- -- means high cost savings

Test Application Method	Design	Test Preparation	Board Manufacture	Board Yield	Board Test	System Test
Income test	o	+	o	+/++	+/-	-
Bare board test	o	+	o	+/++	+/-	-
Visual Inspection	o	o	o/+	+	-	-
MDA	+/o	+	o	o	+/-	o
ICT	+	+	+/o	o	+	--
Functional test	+	++	o	o	+/++	--
Combi test	+	+/++	+/o	o	+	--
Emulation test	+	+	o/+	o	+	--
Memory test	o	+	o	o	-	-
Burn-in test	o	o	o	+	-	-
Hot bed test	o	o/+	o	o	-	-
Installation test	o	+	o	o	o	-
Benchtop test	o	+	o	o	o	+

Table 2: Cost impact of test application methods

2.4. Summary

In this chapter, the author described selected test methods, which are widely used in industry. This included the description and classification of Design-for-Testability methods, test generation methods and test application methods. The economical impact

of the test methods was discussed, and a qualitative cost evaluation of the test application methods was performed. The actual cost of a test method depends on many factors, and these can only be derived for an actual design.

The test strategy planning uses the knowledge which was presented in this chapter in order to create an appropriate test strategy for a given design. The approaches for test strategy planning will be presented in the next chapter. If the economics of test methods are to be taken into account, a quantitative evaluation method for test methods is needed. The related techniques will be described in chapter 4.

Chapter 3

Test Strategy Planning

3.1. Introduction

This chapter describes the process of test strategy planning and outlines a range of systems which are used for test strategy planning. The author will define and discuss the scope of test strategies and test strategy planning. Test strategy planning is driven by selection criteria, which are different for the test strategy planning systems. The author will discuss these criteria and will show, that the economics of a test strategy is the most relevant optimisation criterion. A classification of existing test strategy planning systems will show that there is no general test strategy planning system, which considers all aspects of a test strategy. The test strategy planner ECOvbs (see chapter 7) which was developed by the author is the first system, which takes this general view of a test strategy into account.

3.2. Definition of Test Strategy Planning and Classification of Test Strategy Planning Systems

Test strategy planning for electronic systems is defined as the task, in which the specification of a test strategy for an electronic product is performed. This task is based upon more or less knowledge about the design, the test methods which form the test strategy, and the environment, in which the product is developed and designed. What test strategy planning actually includes, depends on what a test strategy embodies. Therefore test strategy planning ranges from the optimisation of a Design-for-Testability method, such as partial scan selection ([Tri83], [Che89], [Gun90]) or BIST selection ([Bar92]) to a global approach as performed in this thesis.

A test strategy planning system is a system which supports the specification or the implementation or both of a test strategy. Due to different meanings of a test strategy, the scope of test strategy planning systems can be completely different. Some test

strategy planning systems target at optimising a single DFT method for a given design, some aim at specifying a DFT strategy for a heterogeneous ICs, some aim at selecting a test equipment from different suppliers and some aim at defining a procedure of test applications during the manufacturing process. The author has therefore structured test strategy planning by classifying the target test strategy into several classes (Fig. 5).

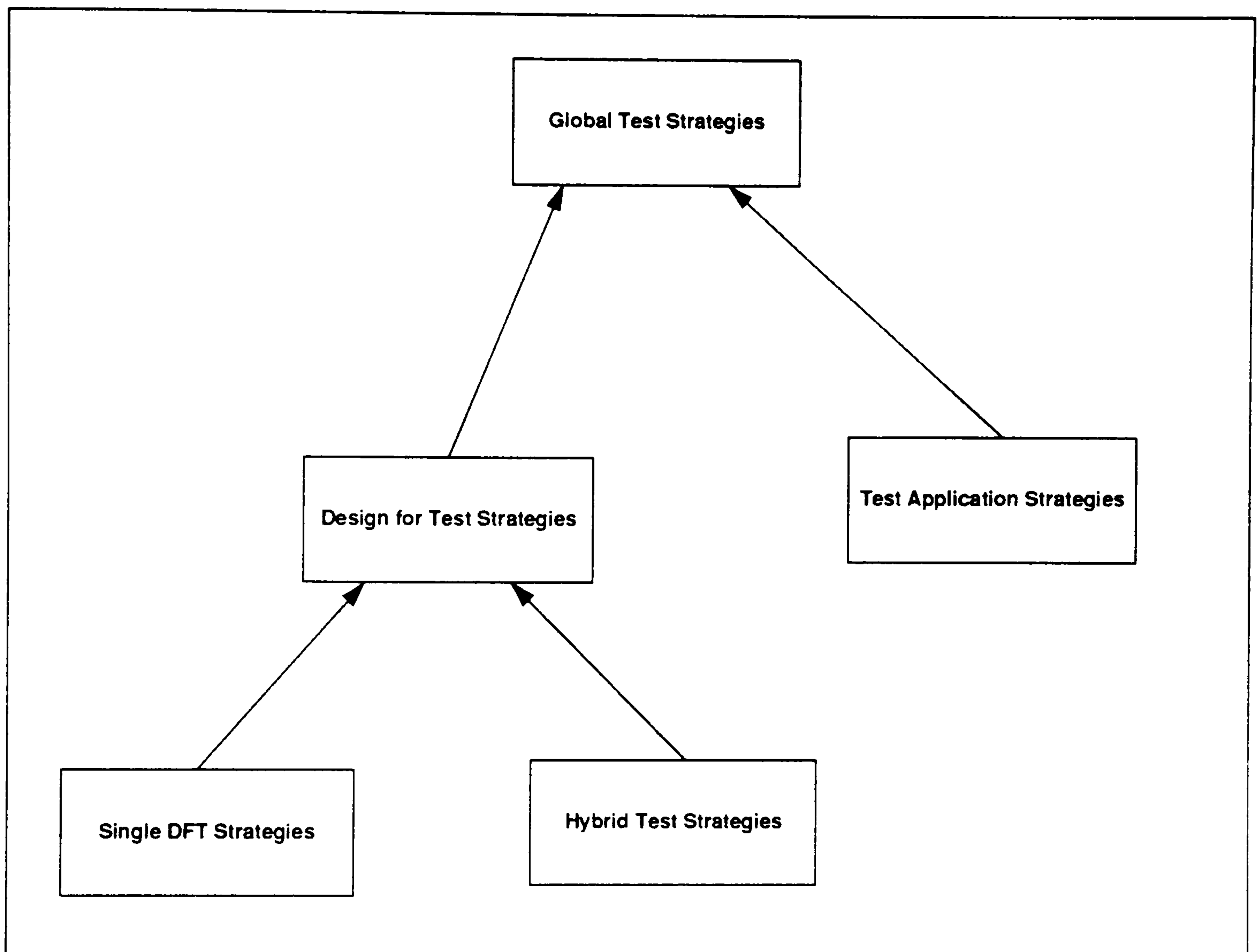


Figure 5: Structure of test strategies

The DFT strategies classification was taken from [Dis92]. There, test strategies are related to structured design to test methods for an integrated circuit. For an integrated circuit the planning of a test strategy is concentrated on planning the design-for-testability strategy. Different DFT methods can be applied in heterogeneous designs and the test application is mostly a single test equipment, which is used once or at most twice (wafer test and component test) in the production phase. Also, by planning a DFT strategy, the selection of a test equipment is mostly included, because the selected DFT methods determine the type of test equipment to be used. This can range from an expensive VLSI test equipment, with DFT depending facilities such as algorithmic

pattern generators or special serial registers for loading and unloading the scan path, to a cheap PC with some extensions, which is used to control the test of a fully self testable IC.

An earlier use of the term test strategy and test strategy planning was related to the test application scenario in the production([Dav82], [Pyn86], [Mah88], [Ham87]). Even the selection of a new test equipment from the market can be seen as test strategy planning ([Dav82], [Ben87], [Pab87]). Most of these test strategy planning systems consider the economics as a selection factor, but the economic evaluation is limited purely to test application.

The global test strategy approach which is incorporated in this work by the test strategy planner ECOvbs, takes all factors of test methods into account for defining a test strategy. The author's definition of a test strategy is as follows:

> *A test strategy is an arrangement of test procedures in the production of a system and the specification of appropriate design methods and test generation methods which support the test of the system.*

This means that a test strategy defines a scenario of test methods used for testing different parts of the system at different levels of assembly. A test method consists of three components, namely design-for-testability, test generation and test application, as defined in the previous chapter. This view of test strategies is especially important since the introduction of boundary scan, which was the first design-for-testability method specially designed for board level test. Today many DFT methods, including BIST, are proposed for facilitating the board level test. Therefore an important economic factor of a test strategy for an electronic system is the DFT methods used. On the other hand, evaluating DFT strategies without evaluating the related test application scenario can also lead to misleading results, especially if the DFT methods can be used for several tests at different levels of assembly, such as boundary scan, or even scan path.

The factors which affect the selection of a test strategy are technical constraints, which can immediately reduce the number of possible solutions, and costing parameters, which are used to quantify the advantages and disadvantages of the test strategies against each other, and which are subject to optimisation. Some systems only use technical constraints for test strategy planning [Laf91]. If several solutions towards a test strategy are possible, the user of the system is free to select one. If costing parameters are used for test strategy planning, these can range from a single parameter to complex test economics models.

An example for test strategy planning by optimising a single parameter is a partial scan selection, which aims at finding a minimum number of flip flops, which breaks all loops in a design ([Che89], [Gun90]). Some systems use several costing parameters, but these are based on different measure units ([Aba85], [Jon86], [Aba89]). The problem here is to compare the different costing parameters to each other. Therefore user defined weighting factors have been introduced, which allow the user to define the importance of each parameter for the selection procedure. But this makes test strategy decision strongly dependent on the user's opinion and no more objective. What the user thinks to be important may not necessarily relate to what is actually important.

What is important for selecting a test strategy out of technically possible solutions, is to minimise the total cost, which is affected by the test strategy. This idea lead to the approach, which was first developed at Brunel University ([Dis89], [Dis91]), and which are used in this thesis. This method is called *economics driven test strategy planning*. Previous economics driven test strategy planning systems concentrated on optimising DFT strategies. The system ECOvbs (see chapter 7), which was developed by the author, aims at optimising the complete test scenario for an electronic system, from design-for-testability selection for the IC components to the final test. This system will be described in chapter 7.

The economical evaluation of a test strategy automatically allows to compare different

costing factors, such as chip area and test time, to each other, because all parameters are converted into real costs. The common measure unit is then £, $, DM or any other currency. This conversion of all cost affecting parameters into the total costs is performed by a cost model. The weighting factors are part of the cost model, which means that the weighting of costing parameters is no longer left to the user. The weighting factors in the cost model convert the costing parameters into real cost, which is an objective conversion into a common measurement unit.

The next chapter will describe the techniques to develop these cost models, and it will present the various cost models which were developed by the author, and which are used in the test strategy planning systems ECOTEST and ECOvbs.

Chapter 4

Test Economics

4.1. Introduction

This chapter will describe the terms and techniques of economics modelling in general and test economics modelling in particular. These techniques are used to develop a hierarchical life cycle test economics model which is used in the test strategy planning systems ECOTEST and ECOvbs developed by the author. The test economics model will be described.

The terms "economics model" and "cost model" have the same meaning throughout this thesis. The term cost is more general. It is not only used for financial parameters but also for technical parameters, such as test time or chip area. Therefore the term cost model is often used for models which calculate these non financial parameters. In the author's approach, cost is always related to financial parameters. This is one reason why the term "economics model" is used instead of cost model. Another reason is the educational aspect. People mostly relate the term cost to additional cost but not to cost savings. But the scope of an using economics models is to save cost. The term *economics* is much more related to cost savings or to the positive effect of a method or strategy. Therefore the term *economics model* is better suitable for the usage in test strategy planning than the term *cost modelling*, because it automatically includes the saving aspect.

Section 4.2 will describe economics modelling techniques by describing some terms, the various types of economics models, and the techniques of developing economics models. In section 4.3 the economic impact of test methods, especially the impact of DFT, will be described and discussed. Section 4.4 will present the structure of the test economics model. In section 4.5 the life cycle test economics model, which has been developed by the author, will be described.

4.2. Economic Modelling Techniques

The main objective of economics modelling is to provide a method for analysing the financial costs of a product or procedure or both **before** they actually occur. Economics modelling is a predictive technique for performing an economic analysis. An economics model allows to estimate the economics of a certain product by predicting the values of the cost-affecting the parameters. A test economics model is an economics model, which is used to predict the economics of test strategies. The analysis is performed to make certain decisions. These decision range from the purchase of equipment to methodical strategies, like design strategies. Therefore an economics model can be seen as a decision model [Din69].

A *decision model* is built upon *parameters* and *variables*, which are used to determine a *target value* [Din69]. In a decision model the target value is subject to optimisation by determining the optimal combination of the variables for a given set of parameters. The parameters are assumed to be fixed for a certain decision to be made, whereas the variables can vary within a defined range. Various techniques exist for determining the optimum mix of variables. These techniques are known as linear or non linear programming techniques [Lue73]. Using test economics modelling techniques for test strategy planning means that all costing parameters, which vary from test strategy to test strategy are the variables of the decision model, and all costing parameters, which are test independent are the parameters of the decision model. The linear programming techniques assume that a variable is continuous in its defined range. This is not the case for the test strategy planning task, because the variable values are fixed for a given test strategy, and each test strategy is defined by its own set of values for the variables. So the optimisation problem for test strategy planning is a combinational optimisation problem, and therefore it is different from the classical linear and nonlinear programming problem. For that reason these techniques cannot be used for the test strategy planning task. Nevertheless the test economics model can be seen as a decision model, but with a different meaning of the term "variable". In order to avoid misunderstandings, the author

calls the "variable" a *test dependent parameter*.

In order to consider all effects of a decision, an economics model should consider all costs during the life cycle of the product, which is subject to the decision. Such a life cycle cost model considers all the costs from the design to the wear out of the product. The process of economics modelling and its applications, such as test strategy planning, is composed of four objectives [Wue84], and therefore it should be performed in four steps:

- The *interpretation objective* aims at recognising the costing relations and the structure of the cost. The cost structure is used to categorise the costing parameters in all phases of the life cycle. This structure can also be hierarchical, as this is the case in the test economics model.
- The *estimation objective* is aims in the development of costing relations, which describe the relation of the cost effecting parameters. These relations can be modelled for example in mathematical equations.
- The *modelling objective* is related to modelling the cost for a given product by using the previous two objectives. This includes the gathering of input data for the economics model as well as the calculation of the cost.
- The *configuration objective* is the analysis of the calculated cost and the decision making process, which is based on this economic analysis. A life cycle cost analysis is defined in [Bla78] as follows:

  A life cycle cost analysis may be defined as the systematic analytical process of evaluating various alternative courses of action with the objective of choosing the best way to employ scare resources.

  This definition matches exactly the scope of test strategy planning. Therefore the configuration objective is the actual test strategy planning process.

The result of the interpretation objective and the estimation objective is the actual economics model. In the author's approach, this economics model is developed once and is used multiply as part of the test strategy planning system. The modelling objective and

the configuration objective are applied each time a test strategy planning session is performed. The test strategy planning systems ECOTEST and ECOvbs provide features, which support the process of achieving the last two objectives. The first two objectives are reached by the development of the economics model, which has been done by the author and which will be described in the following sections. The last two objectives can be reached by using the test strategy planning systems ECOTEST and ECOvbs for a given electronic system, which provide features for gathering the data and calculating the cost (the modelling objective) as well as features for evaluating and assessing the various alternative test strategies.

4.3. Methods for the Estimation of Life Cycle Cost

The determination of the life cycle cost has to be performed very early in the design phase. This is because the decision on the test strategy includes design decisions which have to be taken before the electronic system is fully implemented. This leads to the major problem of economics based test strategy planning. The life cycle costs are not known in this phase of the life. They must be predicted, and in fact this prediction is subject to uncertainty. This problem cannot be solved completely, but by using various methods the risk, which is related to this uncertainty, can be minimised. One of these methods was developed by the author and is described in chapter 6. Another method is the incorporation of the cost driving persons into the process of the modelling objective. This fact is discussed in several publications (e.g. [Hil85], [Kir79]).

Several methods are proposed in the literature in order to estimate costs ([Mad84], see figure 3). The author will present the most common methods and discuss them. Table 3 outlines and classifies these methods.

Methods	Application requirements	Areas of application	Limitations
1. Judgement - expert judgement - educated guess - rough order of magnitude	- experts/experience - rough product definition - analogy material	- early stage - situations without risks - independent crosschecks - budget estimations	- subjective - undefined accuracy - not applicable for price negotiations
2. Parametric - cost estimation relationship(CER) - statistics - models - cost formulas	- historical data - regression analysis - CER material	- concept comparisons - budget planning - tender analysis - independent crosschecks	- extrapolation of data bases and models is often difficult - estimation accuracy can be vague
3. Detailed - work package estimation - work preparation estimation - cost formulas - engineering cost estimates	- time schedule - statement of work and specification - detailed technical material - price tenders	- situations with high risk - price negotiations	- expensive and time consuming - low flexibility - can lead to cost increases

Table 3: Cost estimation methods [Mad84]

The *judgmental methods* are typically used in the very early phase of a product, where no detailed information about the product is available. The methods are based upon expert judgements, analogies, and estimations of a rough of order of magnitude. The estimations can be driven by subjective criteria, and its application is very limited.

Due to that fact many authors propose to use the *parametric methods*. These methods can lead to detailed results even in the early phase of a product's life. But its application requires a certain level of knowledge about the design, production and field phases and methods. A parametric cost model is a cost model which consists of primary and secondary parameters. The primary parameters are the cost driving factors, and the secondary parameters are costs or cost factors, which are derived from the primary parameters or other secondary parameters by a mathematical equation. The most important parametric method is the cost estimation relationship (CER) method. It uses the experience from similar previous projects or products to develop a relation between cost driving factors and the costs. This allows the usage of statistical methods like a regression analysis.

The detailed methods are mostly based upon work packages. The project to estimate is

broken down into small work packages, for which an accurate estimation is possible. The detailed cost estimates assume much more information about the project than the parametric method, which may not be available when the estimation is performed.

The author has selected a combination of the detailed method and the parametric method for the development of the test economics model. The test economics model is structured into several smaller sub models, which are related to work packages. This structure will be presented later in this chapter. The estimation of the costs of the work packages is performed either by parametric models or a mixture of a parametric method and a detailed method. For example, the design time of the ASICs is calculated by a parametric method. The calculation of the assembly cost of a board is based on the number of components per assembly type and the assembly cost per type. The calculation of the assembly cost is a detailed method, but the calculation of the assembly cost per assembly type is based on previous productions and is therefore parametric.

This mixture of the two estimation methods is a good trade-off between the accuracy of the prediction and the cost for the prediction process. The test economics model is intended to be used as the last step of the design specification. The structure of the test economics model reflects the detailing level of a specification. Most of the information about the parameters, which must be defined in order to determine the equations, is available at the end of the specification.

4.4. The Impact of Test Strategies on the Economics of Electronic Systems

VLSI test strategies affect not only the economics of the VLSI itself but also the economics of higher levels of assembly, and later stages of the product's life cycle. This impact is described in a test economics model by the *test dependent parameters*. For example, a self test which is designed for testing the component can also be used for the field test and diagnosis. If for such a self test only the cost at component level are evaluated, the result might be, that the self test is less economical than other test strategies. But considering cost savings, which are made by using the self test in the field

might invert this result. Thus an economic evaluation of test strategies should be done for all test-sensitive cost areas of the product's life cycle.

In the same way the economic evaluation of test strategies for the entire system or for a board, such as boundary scan, should take into account all costs, which can be affected by the test strategy. The scope of this section is to identify the effect of test strategies on the life cycle of electronic systems, and to develop a method, which enables to link the test dependent parameters to the economics model.

Most test methods influence almost all phases of the life cycle. As an example, the test method *boundary scan* is taken to illustrate this fact. Boundary scan has an influence on the design cost of the components, the boards and the entire system. It does affect the production cost of the component, the board and the system. It impacts the test and diagnosis of the board and the system. And boundary scan can be used for field diagnosis, and it therefore has an impact on the down times of a system in the field and the field diagnosis cost. In addition, boundary scan can increase the product's quality, which means, that the mean time between failure decreases, and again the down times are decreased. Also time-to-market may be shortened by implementing boundary scan. This is because of a lower risk of a redesign due to untestability of the design, and because of lower fault diagnosis times for the system, which can easily exceed one week for a complex VLSI based system. So the test method boundary scan does affect nearly all phases of a product's life cycle. Therefore all these phases must be considered for evaluating the economics of boundary scan.

By making a life cycle cost analysis a test strategy is not limited to the combination of test methods for a part of the design, or a particular life cycle phase of the system. A *global test strategy* can be optimised for the whole system in all phases of the life cycle. This leads to much better results from an economic point of view than optimising a test strategy just for a part of the design of the life cycle. But the main objective of a test strategy is to get the system work rather than to achieve a specific fault coverage for a

part of the system. An additional advantage of life cycle cost analyses is, that its economic results are best accepted by the persons who are involved in the implementation of a test strategy.

The impact of a test method on the economics of the system's life cycle is described by the *test dependent primary parameters*. The values for these parameters are different from test method to test method and therefore they are different for alternative test strategies. Beside the test dependent primary parameters, the test economics model comprises of *design dependent primary parameters* and *design independent primary parameters*. Figure 6 shows these elements of the test economics model.

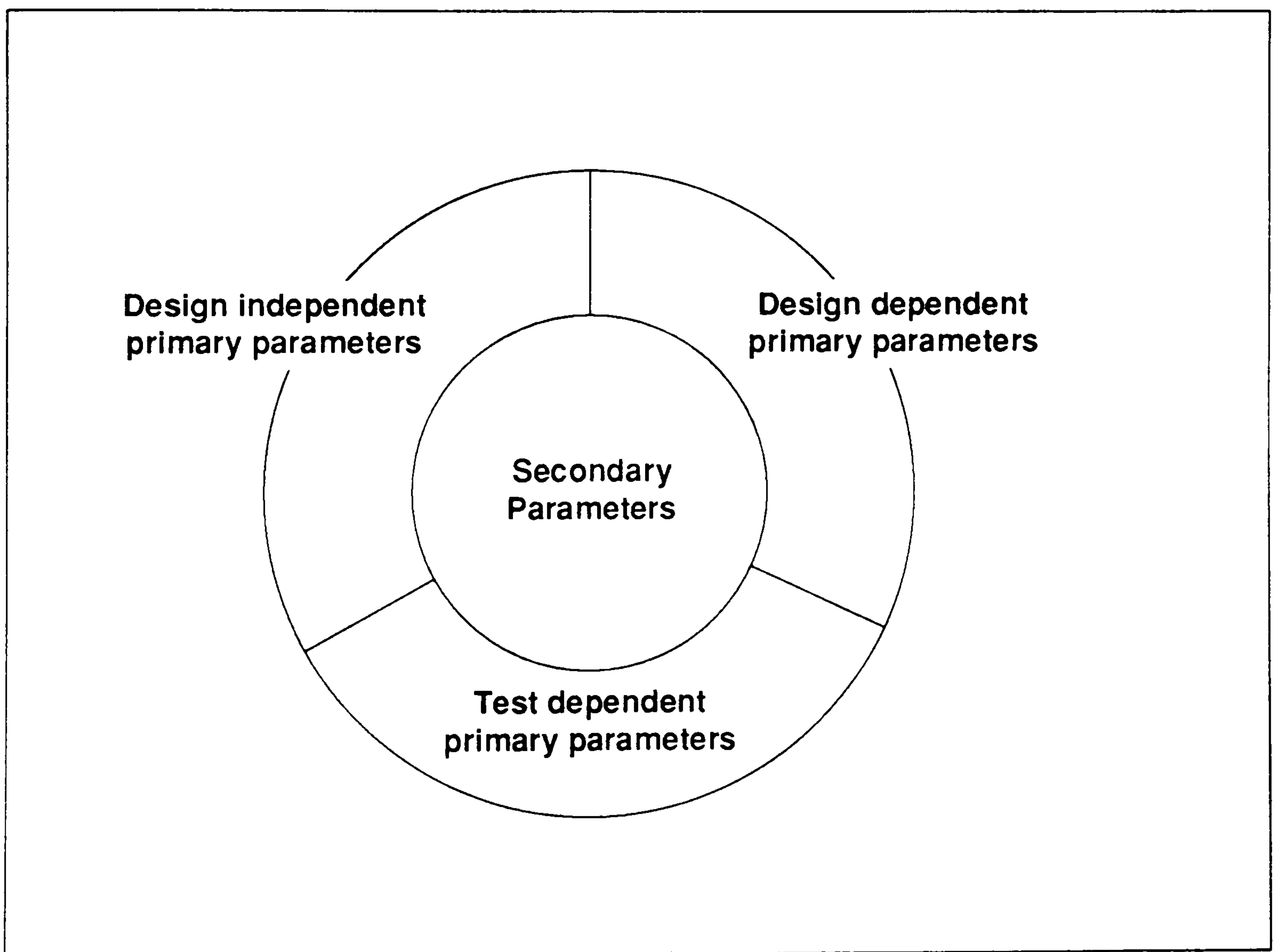


Figure 6: Classification of test economics model parameters

The secondary parameters are based on the primary parameters. Depending on its classification, the primary parameters are provided from different sources:

- The *design independent primary parameters* are those, which do not vary from

design to design. Their values depend on the design environment (such as the productivity of the CAD system used), or they are simply normalising factors. The values of these parameters are stored in a data base, which can be updated whenever the values change. The values are company specific and once they are set, they will rarely change.

- The *design dependent primary parameters* are those, which vary from design to design and which are test independent (e.g. the production volume). Their values are provided through a CAD interface.
- The *test dependent primary parameters* have been described earlier. Its values are provided by test method dependent cost models. This means that for each test method a separate cost model is developed, which calculates the test dependent primary parameters from design dependent primary parameters and test method dependent primary parameters. This means, that the structure of the entire economics model depends on the test strategy, because parts of the economics model depend on the test methods which are applied.

4.5. Structure of the Test Economics Model

The test economics model will be partitioned into several cost sections. The partitioning process will be driven by two criteria:

- The life cycle of the system will be partitioned into several phases and sub phases. The phases will be derived especially from a test view.
- The electronic system will be partitioned from the assembly and production view. This means that every level of assembly will be represented by a separate cost section with its own life cycle phases.

So the test economics model can be seen as a two dimensional model. One dimension is the time (the life of the system), the other dimension is represented by the parts of the system. In the following the author will describe the phases of the life cycle and the levels of assembly.

The electronic system is built upon four levels of assembly:

The whole *system* includes all parts needed to perform the specified functions.

- The system is composed of several *boards*. A board is defined as a module containing several electronic components or sub-modules, which can be assembled into one unit. The connection of boards within a system is implemented by plug-in connections.
- A *module* is composed of electronic components. A module is simply a sub-assembly of a board, and for the test economics model, it will not be seen as a separate assembly level.
- A *component* is the smallest unit for assembly. Electronic components are grouped into passive/active and digital/analog/hybrid. Digital components are usually produced as ICs. In VLSI based systems, the essential components are VLSIs. They may be designed in-house, and different test strategies may be evaluated for them. Therefore a separate test economics model is developed for VLSIs. The other components are assumed to be purchased, and only the purchase cost per component are considered.

The test economics model is partitioned into the system level, the board level and the component level, where the cost of each level are included in the model on top of it.

Phases	Sub phases	Areas of cost origin
Formation	Initiative	
	Planning	Test strategy planning and decision making
	Development	Design, prototype production, verification test engineering
Implementation	Production	Purchase, manufacture and assembly
	Test	Test, Diagnosis, Repair
Usage		Installation, Maintenance, Field diagnosis and repair, Logistics
Phase-out		
☐ phases are not considered		

Figure 7: Life cycle model for electronic systems with regard to test

Figure 7 outlines the phases of the life cycle of electronic systems. The partitioning into sub phases was performed with regard to the impact of test strategies. The test economics model can be simplified by neglecting those phases, which are not relevant for the economic analysis. These are the initiative phase and the planning phase, because the related tasks are performed **before** the test economics model is used, and therefore related costs are not influenced by the test strategy. The phase-out phase does not include test strategy dependent cost (see previous section), and therefore the related cost can also be neglected for the test economics evaluation.

All other phases are considered for the test economics model. Some of the phases are included at all levels of assembly, some of them are included only for a subset of the assembly levels.

4.6. A Life Cycle Test Economics Model for VLSI Devices and VLSI Based Systems and Boards

The test economics model, which was developed by the author, is based on the structure, which was presented in the previous section. Each level of assembly consists of a set of cost models, which are related to the life cycle phases of the assembly unit. The author has separated the costs occurring in the field from the other phases, because they occur only for the entire system. The total cost per assembly unit is derived by the sum of cost in the phases, and the total life cycle cost is derived by including the cost of lower levels of assembly. The economic data needed in the model from a lower level of assembly are grouped into three classes:

- The *volume related costs (VRC)* are all expenditures which occur per device to produce.
- The *non-recurring costs (NRC)* are those expenditures which occur once per product type.
- The *investments* are all expenditures which can be shared with other products.

The following sections will describe the economics models per component, board, system and in the field. A detailed description of the mathematical equations is given in appendix B.

4.6.1. The Test Economics Model for ASIC components

The test economics for ASICs enables the prediction of all costs influenced by the test strategy. The model fits for the development of cell-based ASICs. By cell-based ASIC's the author means semi-custom ASICs, which are developed by using a cell library. A cell library typically contains :

- simple cells like and-gates or inverters
- more complex cells like counters or shift registers
- macro cells like PLAs and RAMs.

The model should be used by ASIC designers and their managers. Test strategy planning should be done hierarchically in the same way as the design itself. Values for the parameters which vary from design to design should be easily accessible, e.g. through a CAD data base. Initial effort is required for data gathering to derive the design independent primary parameters. But the cost for daily use of the model should be negligible. This was an important aspect of the model development.

The costs forming the model are divided into three parts:

- The *production costs* are the costs of the VLSI supplier. These costs are not calculated by a model, because the user of the model (this is the designer) cannot influence the costs by controlling the production process. He has to pay the price which is offered by the supplier, no matter what he would calculate by a cost model. Nevertheless the price is influenced by characteristics of the design, especially the gate number. A kind of data base as customer/supplier interface is defined.
- The *design costs* are the costs related to the design and development of cell-based ASICs.
- The *test costs* are the costs related to testing purposes. These include the costs for fault simulation and test pattern generation. ATE costs as part of the production are only considered, if the ASIC price is influenced by the size of the test set. Costs for an incoming test are usually related to board cost.

4.6.1.1. The Production Costs

The production costs are modelled by the detailed method (see 4.3). The risk of predicting those costs is on the ASIC-supplier's side. He makes an offer, and the ASIC-customer has to pay the offered price. Nevertheless this price depends on several parameters. The values for these must be known, so they are part of the design specification. The idea is to call for price offers for several sensible test plan configurations. Several ASIC-suppliers confirmed, this would be possible. The

production costs are also influenced by negotiations. The prices depend on particular market interests of the supplier. If, for example, a supplier wants to secure business from a company, he would probably start with low price offers. Also it is possible to have more gates on the chip than predicted but not to pay more.

To get a price offer for a specific ASIC the supplier must have knowledge about the following data :

Production Volume: Usually the volume is not one number. The expected delivery volume has to be quoted for every year. Typically the volumes over five years are predicted. For example, the prices can differ, if you have for the same ASIC a production volume of 100,000, and the delivery is distributed in one case over two years and in the other case over five years.

Complexity: The complexity is usually quoted in "number of equivalent gates". The number is based on the equivalent gate number of the cells used. It usually includes the gate number of the i/o-pads and excludes all macro cells. The complexity of the macro cells is determined by their parameters (e.g. the number of bits for RAM's).

Pin number: The number of pins has influence as well on the die size as on the package size.

Number of Test patterns : Some suppliers limit the test set length. If the actual length exceeds this limit, the customer has to pay more. This limit depends on the pin memory size of the ATE.

Technology : Technology of the silicon.

Kind of Package : Package size and stock.

The chip price is composed by two different cost parts :

The **non recurring engineering (NRE) charges** include all services from the supplier. The NRE charges incur per design. What they really include depends on the customer-vendor interface (see [Ard87]).

The **production unit costs** is related to all costs incurred during the production of the chip. They include the costs for silicon, package, wafer production, packaging and testing. These costs are incurred per production unit. If there is a surcharge for long test sets, this additional price is part of the test costs.

4.6.1.2. The Design Costs

The design costs depend strongly on the designers environment. Looking at some real data about design schedules the author has found that the complexity of the design cannot only be measured in terms of gates. The correlation between design time and the number of gates to design is not very strong. The author supposes that in some cases the Parkinson principle ("work expands to fill the available volume") is the most suitable model for estimating design time. Nevertheless it should not be the aim of test economics modelling to follow this principle.

The model must be used with the following assumptions :

- The ASIC to be developed is a semi-custom design as described earlier.
- A top-down design by taking advantage of the hierarchical architecture is assumed.
- The CAD-system runs on a workstation. For workstations the CPU-time costs are usually included in the engineer's cost-rate. If an outside design-centre is used or some of the CAD tools run on a mainframe, these additional costs are drawn explicitly.

The following design phases are taken into account for the prediction of design costs:

- Initial development of the circuit.
- Design capture.
- Simulation and Verification.

The calculation of the design time is not only based on the gate count but also on other design parameters and design environment parameters. The design parameters are the

gate count, the number of cells, the pin count, the originality of the design and the performance criticality. The design environment or productivity parameters are the productivity of CAD system which is used, the experience of the designers and the productivity of the cell library, i.e. how well do the functions of the cells provided by the cell library match to the actual design.

Test pattern generation and fault simulation are also part of the design process, but in the test economics model the effort are considered as test costs.

4.6.1.3. The Test Costs

Test costs from [Var84] are composed of test pattern generation costs and test application costs.

The test application costs amount to wafer test and component test. The wafer test and the component test are part of the ASIC production. Test costs as part of the ASIC price is usually constant. In some cases it is dependant on the test set length. A fixed number of test pattern is included in the ASIC price and additional patterns have to be paid extra. This extra price is step like. The steps depend on the pin memory size of the ATE. So the extra price is part of the test costs, the basic test application costs are hidden as part of the ASIC price.

The test pattern generation cost depend on the circuit, the fault coverage to achieve, the performance of the test pattern generator and the type of test pattern generation (sequential test pattern generation or combinational test pattern generation). Deterministic test pattern generation can be performed by a tool (ATPG) or a person (manual test pattern generation, MTPG). The generation of random test pattern causes no test pattern generation cost. So the costs for test pattern generation for all self tests, which are based on random pattern, are negligible.

Manual test pattern generation is done, where the ATPG system has problems, or where an ATPG system is not available. Problems for ATPG systems usually occur having

circuits with large sequential depth, and for asynchronous circuits. Especially for asynchronous circuits MTPG is done in some cases. MTPG is done for the faults uncovered by the test pattern from ATPG and the functional pattern used for the design verification. It is also done for generating the propagation of patterns through the circuit. The model for ATPG cost is based on [Goe80]. The evaluation of data measured about ATPG for scan-based circuits showed, that there is no significant correlation between the CPU time and other circuit characteristics than the gate count.

Most of the ATPG systems for sequential circuits multiply the circuit for every time frame. Therefore the effort on the average sequential depth of the circuits as well as the number of gates.

The correlation of test pattern generation costs and fault coverage is also derived from [Goe80]. After reaching a fault coverage between 80% and 90 % very fast (exponential shape, phase I), the curve becomes linear (phase II). This linearity is probably caused by the fact that now most of the generated test pattern cover just a few additional faults. So the slope depends on the fault coverage, where the slope becomes linear, and the complexity of the circuit for test pattern generation.

4.6.2. The Test Economics Model for Boards

The test economics model for boards includes the two main phases "development" and "production". Figure 8 shows a flow diagram of the phases.

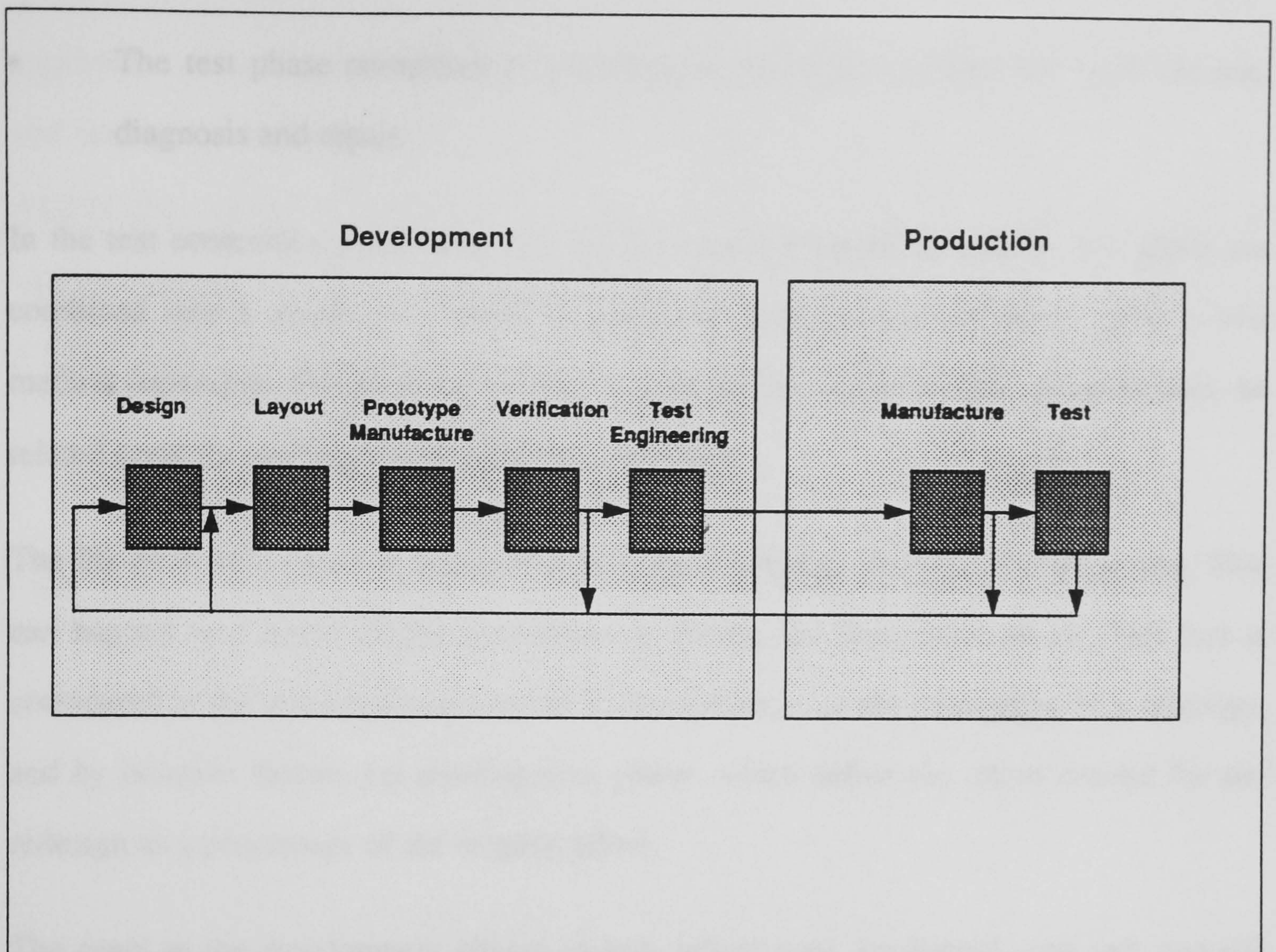


Figure 8: Flow diagram of board level phases

The development phase includes the following sub phases:

- The design comprises of the initial design, design entry and computer simulation.
- The layout phase includes the placement and floor planning of the PCBs.
- The prototype manufacture covers the construction and manufacture of prototype boards.
- In the verification phase the evaluation of the board by verifying the specified functions with the prototypes.
- The test engineering phase covers the generation of test patterns, the generation of the test programs and the manufacture of test tools, such as a bed-of-nails fixture for an in-circuit test.

Due to the test view the production phase is partitioned into two sub phases:

- The manufacture phase includes the production preparation, fabrication and assembly.

- The test phase comprises of test application, which includes the costs for test, diagnosis and repair.

In the test economics model structure the test engineering phase and the test phase are combined into a single cost model because the cost structure of these costs is test method dependent. For each test method a separate cost model exists, which models the related costs for test engineering and test application.

The life cycle of a board includes a chance that a redesign may become necessary. This can happen as a result of the verification or during the production phase. This fact is considered in the test economics model by the definition of the probability of a redesign, and by iteration factors per development phase, which define the effort needed for the redesign as a percentage of the original effort.

The costs in the development phases include labour cost, equipment cost and material cost. The labour costs are determined by the hourly labour rates and the predicted effort. The equipment costs are related to computer equipment, which are needed for the development of the design. They are also calculated by the hourly costing rates and the estimated time, the equipment are needed. If the costs are included in the hourly labour rates of the designers, the equipment costs can be set to zero. Material occur only for the manufacturing of the prototypes.

The manufacture costs are composed of the production preparation costs, the material costs and the assembly costs of the board. The material costs include also the total costs for the components, which are calculated by the ASIC test economics model. The calculation of the assembly cost is based upon the assembly cost per component and per assembly type and the number of components per assembly type. In addition to the costs the number of defects per defect type are calculated. These data are needed for the test phase.

The test cost model includes the calculation of the test engineering costs and the test

application costs. The test engineering costs are composed of test tool manufacture costs and test generation costs. The calculation of the test generation costs is based on the engineering effort, the engineering labour rate, the usage of equipment and the equipment rate. The test application phase is built upon the test phase and the diagnosis phase as shown in figure 9.

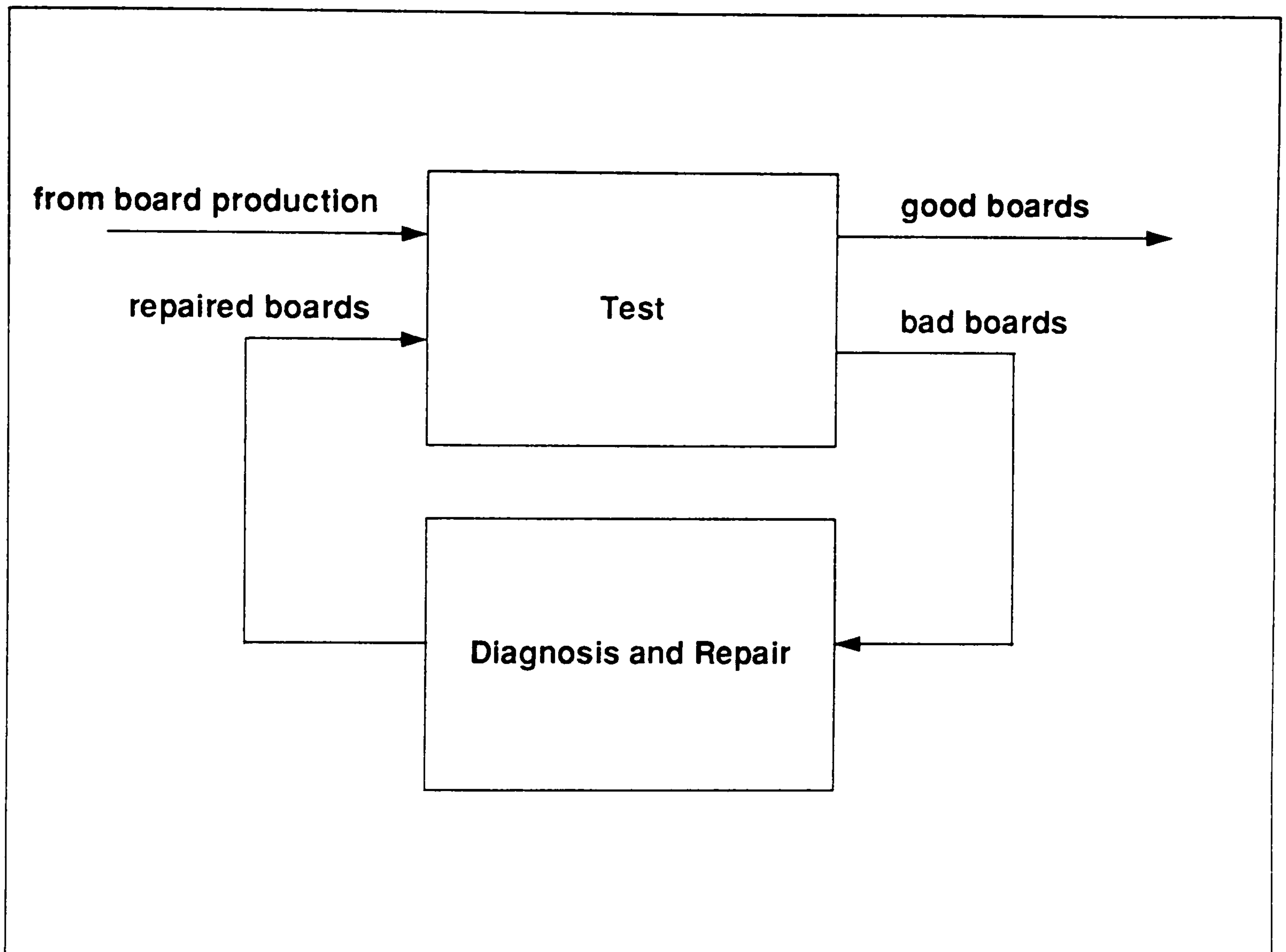


Figure 9: The test/repair loop of the test application phase

The test is applied to all boards coming from production. The diagnosis and repair are applied only for those boards, for which a defect has been identified during the test. All repaired boards are retested, in order to detect multiple defects, and to detect defects, which are injected during the repair.

In addition to the test and diagnosis/repair partitioning, the test economics model is partitioned into four parts:

- In the quality model the fault coverage of the test is calculated, which is based on

the fault spectrum of the previous stage. Depending on the test strategy, the previous stage can be the manufacture phase or another test phase.

- The time model calculates the total times for test application, diagnosis and repair, depending on the times per board and the number of boards going to test and to diagnosis/repair.
- The cost model calculates the actual financial cost. This includes the procurement of test equipment and the usage of these test equipment, and the labour costs for the test personnel. In addition the repair costs are calculated. These are based on the labour costs and the material costs. For the calculation of the material costs the type of the defect is taken into account.

The times and the financial costs are calculated for the test phase and the diagnosis and repair phase.

4.6.3. The Test Economics Model for Systems

The test economics model for electronic systems includes the phases development, production and field usage. The field usage will be described in the next section. This section describes the economics models of the development phase and the production phase of the system.

In the area of the *system development* only the test engineering costs are relevant for the test strategies. The test strategy dependent development costs are mainly related to the board development and the component development, and the related costs are included in the total system costs as part of the total board costs. The test engineering costs are combined with the test application costs in the same way as for boards.

The *production cost model of the system* includes the costs for the assembly of the boards into a system, the total costs of the boards and optional device costs for incorporated test devices, which support the test and diagnosis of the system in the field.

The system test costs comprises of test engineering costs and test application costs in the

same way as for board test. The test/repair loop is the same as for boards.

4.6.4. The Test Economics Model for the Field Costs

The test relevant parts of the field phase of the life cycle for electronics systems are the installation of the system, the field maintenance and the field breakdown. All three phases consist of a diagnosis/repair loop. The main difference between the phases is the test phase. The installation test is different from the maintenance test, and the breakdown phase doesn't consist of a test phase at all, because the system is known to be defective. But in the case of a breakdown, the system is always defective and goes through the diagnosis and repair loop, whereas in the installation phase and the maintenance phase only a portion of the systems goes to diagnosis and repair.

The diagnosis and repair phase consists of three stages. Figure 10 shows the field repair loop. If a system is defective, the defect is diagnosed in the field. If the defect can be detected and repaired in the field, it will be repaired there. If not the system will be repaired in the field by replacing the defective boards and the defective boards go to the service centre for further diagnosis. The service centre provides the spare boards. If the defective board can be repaired in the service centre, it will become a spare board. If not the defective board will go into the depot, which is mostly identical to the production facility. If the board can be repaired in the depot, it will be sent back to the service centre to go into the spares stock. If it cannot be repaired, it will be sorted out.

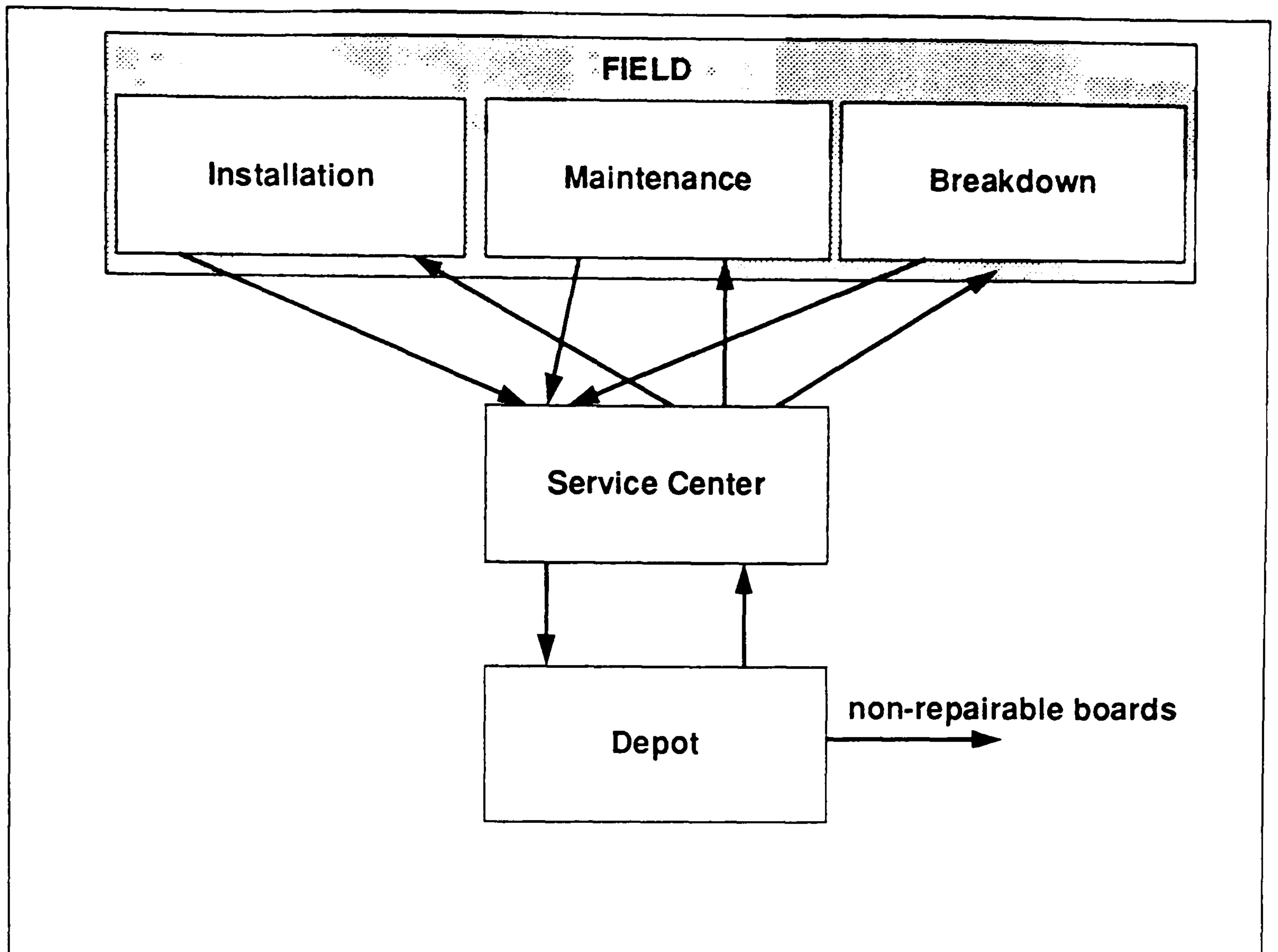


Figure 11: Field repair loop

From this scenario the following cost models are developed:

- The **field usage cost model** consists of field installation, field maintenance and the field repair costs. This includes the defect rates of the system to be installed, the defect rates of maintained systems and the mean time between failure in order to calculate the breakdown rate. The number of systems, which cannot be repaired in the field, is determined from the field repair rate, and the number of boards, which are going to the service centre is derived from the average number of boards which are replaced at one failure and the number of failures.
- The **service centre cost model** includes the costs for the provision of spare boards and the repair costs for the boards, which are actually repaired in the service centre. The number of spare boards which are needed takes into account the time needed to repair a defective board.
- The **depot cost model** includes the repair costs for the boards and the production of new spare boards, which are needed due to the sort-out of non-repairable

boards. The number of non-repairable boards is derived from the total number of boards going to depot repair and the percentage of boards which are not repairable.

4.6.5. Consideration of Interest Rates

Interest rates can be an important factor, if the time for return-of-investment is considerably long and if the market interest rates are significant. Therefore the costs for the early development phases differ from the same costs in the production. The following example should make this fact clear:

The total development time of a system takes two years. The market interest rate is constant at 12%. The initial design is performed at the beginning of the development phase, and the test program generation is performed at the end of this phase. For simplification, these two phases are assumed to be very short compared to the total development time. The costs for both phases is £10,000. To compare these two costs, they must be related to the same time point. We take the end of the development time as the common time point. This means, that considering the interest rate impact, the £10,000 for the test engineering phase remain unchanged, whereas the £10,000 for the initial design become £2,544.

So, expenditures which are made early are more expensive as the same expenditures, which are made later. This fact can be important for test strategy planning, if the interest rates are high and two test strategies differ in expenditures which have to be made at significantly different times.

For that reason the author has included the interest rate aspect. All costs are converted to a common time point, which is the end of the life cycle. This is done by using the present value method [Blo88].

4.7. Summary

This chapter presented economics modelling techniques and their uses in test economics modelling. The impact of test strategies on the economics of electronic systems during the life cycle of the system was described and discussed. The test economics model which was developed by the author was described. This test economics model will be used for test strategy planning in order to evaluate the economics of test strategies. The author has developed methods and advisory software systems for supporting this test strategy planning task. ECOTEST is a test strategy planning system for ASICs, and ECOvbs is a test strategy planing system for VLSI based systems. These two systems and underlying methods and algorithms will be described and discussed in the following three chapters.

Chapter 5
ECOTEST

5.1. Introduction

ECOTEST is a test strategy planner, which is an enhancement of the EVEREST test strategy planner [Dis92]. The author has enhanced certain parts of the EVEREST test strategy planner, which were identified to be weak. But most of the concepts remained unchanged.

The development of the EVEREST test strategy planner was a collaborative project between Brunel University and Siemens-Nixdorf. The author has developed many concepts, which are implemented in the EVEREST test strategy planner, and he has also developed major parts of the software. The EVEREST test strategy planner is described in detail in [Dis92]. This chapter will concentrate on how ECOTEST is integrated into a test engineering environment, and on the testing philosophy behind ECOTEST. The complete system will be described as an overview, and the enhancements of ECOTEST against the EVEREST test strategy planner will be described in detail.

Previous work in this area has mainly been addressed in [Aba89], [Dis91] and [Laf91]. TIGER [Aba89] performs test partitioning and test plan generation in addition to automatic test strategy planning (ATSP). The aim is to make a design testable and stay within certain design limits such as area or test time. However the cost measures adopted to drive the search for the best test strategy are based on different measure units. For that reason they cannot be evaluated against each other. ITSELF [Laf91] is a system to check the applicability of test methods, which is an important task when the large variety of test methods is considered, but does no selection, if several solutions are possible. The system only checks whether the design and test constraints are met. The work at Brunel University in this field began with the development of a test economics model [Var84], where design-for-testability methods like scan path or self test were studied under

economic aspects. The advancements of this work resulted in the test strategy planning system BEAST [Dis92]. In BEAST, the decision process of selecting test methods test methods is driven by the economics of its application. This means, that the selected combination of test methods (the test strategy) should be the cost optimal solution. In the joint project with Siemens-Nixdorf a new test economics model (see chapter 4) was developed and included in the EVEREST test strategy planner([Dis91], [Dis92]). ECOTEST is an enhancement of the EVEREST test strategy planner.

The scope of ECOTEST is to find the most economic test strategy for a given design, which fulfils all the technical constraints. Other systems (e.g. [Aba89]) limit the search space by setting for example area and test time constraints. Such limitations are normally supplied by the designer. In addition to the limitations, user defined weights are used in order to allow the comparison of different parameters. This comparison can then be used to select a test strategy among the ones which meet all constraints. But the final selection depends on the objectivity and experience of the designer in weighting parameters which are not always directly comparable. This fact may lead to non optimal solutions. For that reason, the EVEREST test strategy limits the user defined restrictions to technical constraints only, while cost related parameters are compared using the test economics model. This provides a common reference point for all cost related parameters, which can therefore be compared objectively.

In the following section the philosophy of ECOTEST will be described. In the rest of this chapter the usage of ECOTEST in a test engineering environment will be described and discussed, the EVEREST test strategy planner, which is the basis of ECOTEST, will be outlined, and the enhancements of ECOTEST against the EVEREST test strategy planner will be described and discussed.

5.2. The Philosophy of ECOTEST

Today's IC technology allows to integrate different types of circuits on one chip. A typical VLSI of ASIC consists not only of random logic, but contains also other design

methodologies such as RAMs, ROMs, PLAs and complex macros as e.g. microprocessor cores, data paths, or multipliers. Some of these blocks are created via logic synthesis, silicon compilers or module generators, others are predefined macros. Due to the variety of structures and functions on such an IC, this type of circuits is called *heterogeneous circuit*. In order to keep the costs of testing of chips within reasonable bounds, a variety of design-for-testability methods have been developed. Most of the advanced design-for-testability are optimised for a specific class of circuits. Especially for RAMs and PLAs a large variety of DFT methods have been developed and published. For example, in [Zhu88] about 20 DFT methods for PLAs are presented. The heterogeneity of the circuits and the variety of DFT methods have led to the idea of modular testing or macro testing ([Bee90], [Rot89]). This method allows the use of different test methods and DFT methods for different parts of the circuit. The independently tested parts are called *testable units*. For every testable unit, a different DFT method can be applied. In the macro test methodology [Bee90] is coupled to a silicon compilation approach, and the testable units are related to the functional macros which are generated by a silicon compiler. The modular test methodology does not link the testable units, which describe the test hierarchy, to a certain design hierarchy. Therefor the testable units can be selected by minimising the related costs.

In ECOTEST, we have adopted this modular test approach. The test hierarchy is defined by the netlist hierarchy. The netlist hierarchy can be provided by a test partitioning tool, or by the designer. Different test hierarchies may be evaluated in order to find the cost optimal solution.

A test method describes a procedure, which includes a DFT method, a test generation method and a test application method. A test application method always includes a test pattern driver and a test response receiver, which are called here the *test resources*. In the case of an external test, these are represented by the chip inputs and outputs. In the case of BIST, the test resources are represented by the BIST logic. A testable unit is defined as a part of the circuit, which is homogeneous respecting the design

methodology, and to which a test method is applied. The testable unit must be accessible by the test resources of the test method to be applied. This accessibility can be achieved through transfer paths in the circuit, or through additional DFT logic. In ECOTEST, this additional DFT logic is called *external test method*, whereas the test method to test the testable units is called *internal test method*.

The objective of test strategy planning in ECOTEST is to find the most economic test strategy for the target design with a given test hierarchy. By varying the test hierarchy, the user can evaluate the impact of different hierarchies on the economics of the test strategies.

This leads to the way ECOTEST should be used. ECOTEST is a testability advisor, which advises the designer in how to create a design, which is testable by minimum total expenditures. This creation of a testable design is called test strategy. The test strategy is the specification, which testable unit is tested by which test method. So the decision on a test strategy includes the decisions about the design methodology (DFT), test generation methodology and the test application methodology. From these decisions, the first to be made is the design decision. The point of time, when this decision must be made, depends on how deeply the DFT method is integrated into the functional design. For some DFT methods, such as all scan design techniques, the decision must be made, before the design starts, i.e. during the specification phase. But the answer to the question, which scan design technique is the best, may be left until the functional design is completed. This late decision is advantageous, because the data, on which the test strategy decision is made, i.e. the costing parameters of the test economics model, are more accurate in this phase. For that reason ECOTEST should be used in two phases of the design:

- In the design specification phase, ECOTEST should be used in order to make some general decisions, such as whether synchronous design is required or not, or whether a scan design will be implemented or not, i.e. all decisions which affect the functional design of the circuit.

- After the completion of the functional design, the detailed test strategy planning should be performed. This includes the decision on what scan design, what kind of self test, or what test hierarchy is optimal. During this phase most of the costing parameters are known very well. For example, parameters such as the gate count or the number of flip flops can be extracted from the netlist description instead of predicting them. In addition, simulation tools can be used, which simulate a certain DFT condition in order to derive parameters such as the number of test patterns, or the achievable gate count. Such simulation tools have been developed , and they can be linked to ECOTEST. For example, the test pattern generation system TENsocrates[TEN91] includes a preview mode, which simulates a full scan path for a given design, and generates test pattern for it in order to derive the number of test patterns and the achievable fault coverage as main costing parameters. Or the BIST advisor TENstar [Bar92] simulates several self test methods for a given design in order to derive the achievable fault coverage and the number of test patterns. By linking these tools to ECOTEST, a very accurate economic analysis can be performed.

The local application of a test method to a testable unit may have global implications to the design, which will affect the test economics of the whole design. These implications result from accessibility implications, which will affect the accessibility of other TUs, the exceeding of technical constraints, the sharing of test pins between TUs, the shareability of test resources, and the nonlinear behaviour of costing functions.

A local test method application can affect global accessibility, due to transfer of data to neighbouring TUs. If a TU is made accessible through a test method, all ports of the TU become accessible, and therefore also all ports, which are connected to ports of the TU under consideration. If such a port is part of a block, which is capable of transferring the data, e.g. a multiplexer, or a register, other ports also become accessible through that transfer function under certain conditions. These conditions are the controllability of this transfer function. For example, a multiplexer can transfer data in a controlled way only, if

the control line of the multiplexer is accessible.

If a test method application implies the need for additional test pins, the same pin may be used for several test methods applied to different TUs.

If a technical constraint such as the maximum number of pins is exceeded, this exceeding may result from additional pins which are needed for different test methods which are applied to different TUs. So, the application of a certain test method to a certain TU may exceed this global limit, but the exceeding is not directly related to this application but to the application of all test methods which need additional pins.

Self test methods often use a special logic to generate test patterns and to evaluate the test responses online. This logic is called *test resource*. These test resources can be shared between TUs. Figure 12 shows an example for the shareability of test resources.

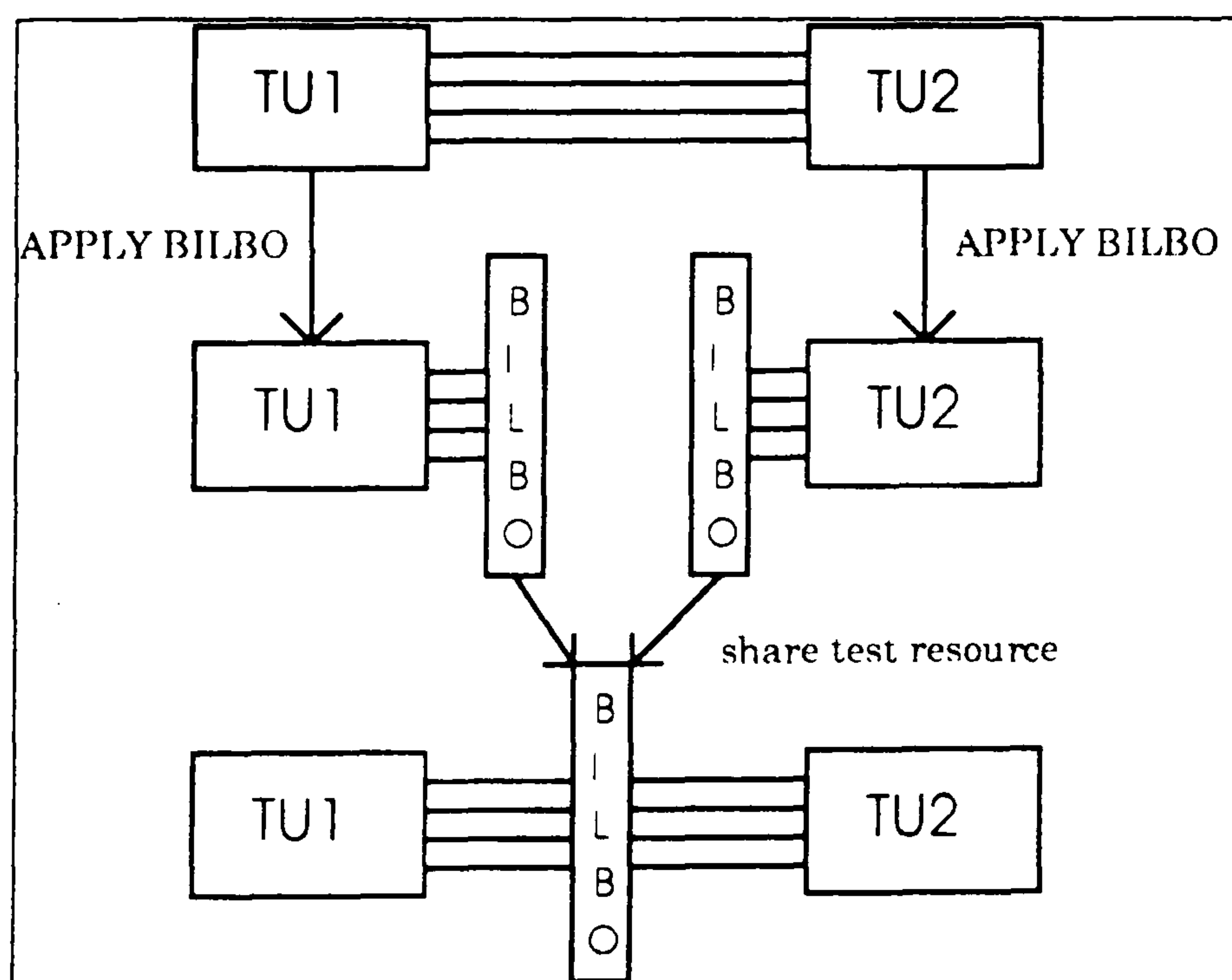


Figure 12: Shareability of test resources

Assuming, that the test method BILBO [Koe79] would be found to be the best test method for TU1 and TU2, the test method application procedure would require for each TU a BILBO register at the inputs and the outputs. Due to the fact, that the outputs of TU1 drives the inputs of TU2, the BILBO register at the output of TU1 can be easily shared with the BILBO register at the input of TU2.

5.3. ECOTEST in a Test Engineering Environment

ECOTEST as an advisory tool should be used in combination with other test engineering tools, which support the implementation of the selected DFT methods, and which support or automate the generation of test plans, test patterns and test programs. The variety of test methods strongly depends on the availability of such tools. For example, it makes no sense to select a scan design test strategy with automatic test pattern generation (ATPG), if no ATPG system is available. Also, ECOTEST should be totally integrated into the CAD system, which is used, in order to provide data such as the netlist, or cell library data. The following links between CAD tools and other test engineering tools should be incorporated:

- ECOTEST should be linked to the CAD system by a netlist interface, a functional interface and a cell library data interface. The netlist interface should provide all data about the test hierarchy, i.e. which are the testable units, the type of the testable units (e.g. RAM, combinational random logic, sequential random logic), and the structure of the design. The functional interface should provide the information about the data transfer properties of the testable units. And the cell library data interface should contain per cell data like the size (in equivalent gates, in mm^2, or in bits for RAMs) or the number of basic storage elements (e.g. number of flip flops).
- ECOTEST should be linked to tools which derive costing parameters, or which analyse the design concerning testability characteristics. These tools include advisory tools as mentioned in the previous section, testability synthesis tools such as partial scan selection tools ([Tri83], [Che89], [Gun90]), and tools, which analyse the accessibility of the testable units, such as SPLASH [Keu93].
- The test method data base of ECOTEST should take into account the availability of test methods due to the availability of test generation tools and testability synthesis tools. The economics of a test method strongly depends on tools which support this test method, in terms of automatic test pattern generation as well as

in terms of testability hardware synthesis.

These aspects are considered in ECOTEST through its interfaces, which allow to access the data coming from other tools. ECOTEST comprises of an EDIF netlist interface, which provides integrity to nearly all CAD systems on the market. It consists of a cell library data interface, which allows to access automatically data about the complexity of the testable units. The data transfer characteristics of the testable units are accessed through an interface, which was introduced by Philips [Bee90]. And the test method descriptions are provided such that the user can maintain the data in order to adapt them to its own environment.

5.4. The EVEREST Test Strategy Planner

5.4.1. The Data Interfaces

Figure 13 shows the architecture of ECOTEST. ECOTEST uses the following data interfaces:

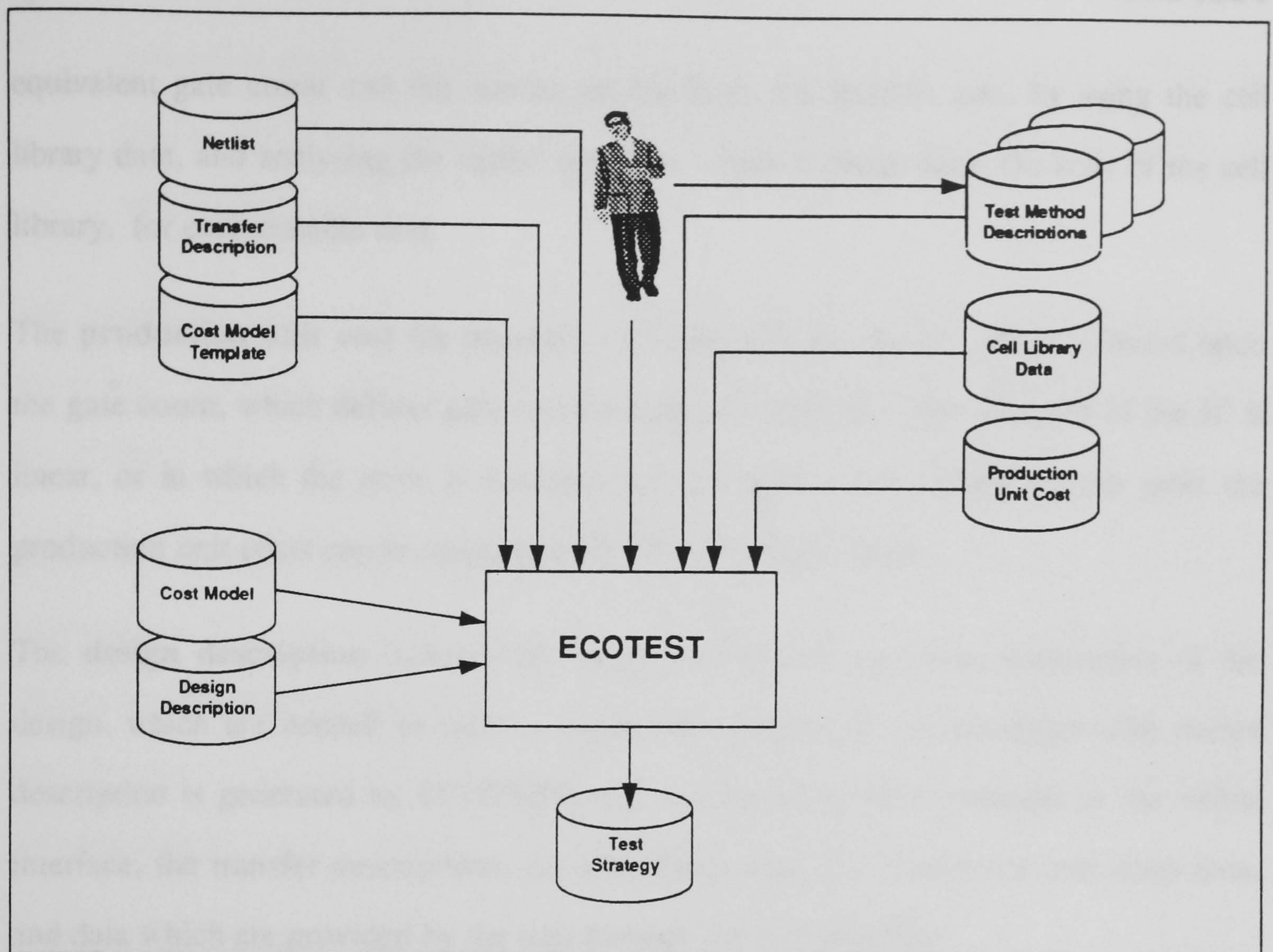


Figure 13: The ECOTEST architecture

The **netlist** provides the data about the structure of the circuit, the definition of the testable units, and a classification of the input ports of the testable units. All cells, which are part of the top level cell, i.e. the main circuit, are considered as testable unit. The type of the testable unit is defined by a property entry. Cells, which are defined on that level, but which should not be considered as testable units, such as pin driver cells in a standard cell design, can be classified as no testable units by a specific property class. The format of the netlist is EDIF.

The **transfer description** provides the information under which conditions test data can be transferred through a testable unit.

The **cell library data** file contains the complexity data of the cells in a cell library. This complexity data include the equivalent gate count and the number of storage elements for random logic cells, the number of bits for RAMs or ROMs, and the number of product lines for PLAs. ECOTEST comprises of a function, which automatically calculates the

equivalent gate count and the number of flip flops per testable unit, by using the cell library data, and analysing the netlist structure, which is based upon the cells of the cell library, for each testable unit.

The **production unit cost** file provides a pricing table for the IC, which is based upon the gate count, which defines gate count ranges, in which the price increase of the IC is linear, or in which the price is constant, e.g. for gate arrays. Based on this table the production unit costs can be calculated for the related gate count.

The **design description** includes all netlist data and all economic parameters of the design, which are needed to make an economic analysis of test strategies. The design description is generated by ECOTEST, and it is based on data provided by the netlist interface, the transfer descriptions, the cell library data, the production unit costs data, and data which are provided by the user through the user interface.

The **cost model template** is the basic cost model. This is design independent, and it contains the general secondary parameters, the templates for secondary parameters, which occur per testable unit, and templates for all primary parameters. The cost model consists of primary parameters and secondary parameters. The primary parameters provide the input values to the cost model, and they are classified into three groups:

- The *design dependent primary parameters* are those, which do not vary from design to design. Their values depend on the design environment (such as the productivity of the CAD system), or they are simply normalising factors (such as normalising the values to the currency which is used). These parameters are company specific and once they are set, they will rarely be changed.
- The *design dependent primary parameters* are those, which vary from design to design, but which are test independent (e.g. the production volume). Their values are extracted from the design description.
- The *test dependent primary parameters* are those, which vary from test strategy to test strategy. Their values are calculated in separate cost model, which are

defined in the test method descriptions, and these parameters are linked to the related test method descriptions, depending on which test method is applied to which testable unit.

The secondary parameters form the cost model kernel. These parameters are set to equations, based on either primary parameters or previously calculated secondary parameters.

Based on the cost model template and the design description which contains the information about the testable units and the primary cost model parameters, the **cost model** is automatically generated by ECOTEST. All parameters which are defined per testable unit, such as the gate count, are expanded to the correct number of testable units by extending the name of parameter by the name of the testable unit.

The **test method descriptions** provide all the information about test methods which is needed to perform test strategy planning. This includes the suitability of the test methods for the particular testable unit, basic design implications of the test method application, and a cost model, which defines the test dependent costing parameters as a function of the design parameters. The design implications are the type of the test method (internal or external, self test or not), a pin compatibility class and the accessibility implications. The pin compatibility class defines the shareability of the additional pins between the test methods of different testable units. The accessibility implications define, whether the accompanied test method provides accessibility to the inputs or outputs the testable unit, to which it is applied. The test dependent parameters are:

- The gate count.
- The number of cells.
- The design originality.
- The performance impact.
- The number of additional input pins, output pins and bidirectional pins.
- The number of test patterns per test pattern type. The test patterns types are

normal test patterns, self test patterns, scan test patterns. The test equipment may handle these test pattern different, and therefore the test application costs may be different for the different test pattern types.

- The achievable fault coverage.
- The test pattern generation cost.

These parameters are described in formulas in the same syntax as the cost model. The formulas are based upon the design parameters. The test dependent parameters are used as input parameters of the cost model.

The **test strategy** file is generated by the ECOTEST, and it contains the definition of the final test strategy and the values for all costing parameters.

5.4.2. The Functions of ECOTEST

ECOTEST comprises of four functional blocks:

The **design specification reader (DSR)** prepares the netlist data and the other design related data for the internal design representation and creates the design description file and the design specific test economics model (the cost model). It allows to modify the cost related data of an existing design description.

The **cost model (CM)** contains the knowledge base about the cost relations of the design. This functional block includes the parsers of the cost model file and the test method related cost models, the building of the internal cost model structure, the evaluation of the cost model, the printing of the cost model parameters, and a sensitivity analysis for a single parameter by varying this parameter and calculating the impact of this variation on another parameter of the cost model.

The **test strategy planner (TSP)** evaluates the economics of the test strategies by using the test method descriptions, the cost model and the design description. The test strategy planner allows interactive test strategy planning as well as automatic test strategy

planning. Therefore it provides the following functions:

- Various automatic test strategy planning functions are provided. These enable automatic test strategy planning of the whole design, of single testable units (TUs), automatic test strategy planning to make the TUs accessible and automatic test strategy planning without making the TUs accessible.
- Interactive test strategy planning can be performed by using a function, which applies a user defined test method to a user selected TU.
- The accessibility function calculates the accessibility of the input- and the output ports of the testable units. The calculation is based upon the netlist, the transfer functions of testable units, and on accessibility enhancing characteristics of the test methods, which are applied to the testable units.
- The test method descriptions are parsed and provided for the test strategy planner.
- The applicability of a test method is checked, before it is applied. This includes a check of the suitability of the test method to the type of the TU (e.g., for RAMs, only RAM test methods can be applied), and a check of the violation of constraints. the constraints are user defined, and they include a maximum gate count, a maximum pin count, and a maximum self test time.

The **user interface** is implemented as a command handler system, and it provides four different command handlers:

- The ECOTEST handler includes general commands for data handling and for maintaining the handler.
- The DSR handler provides several commands fro the set up and modification of design data.
- The TSP handler provides commands for the modification and handling of test strategies.
- The PRINT handler provides several print commands.

Table 4 provides a list of all commands, which are implemented in ECOTEST.

Command name	Description
ECOTEST handler	
alias	define a new command
enter	enter another command handler
shell	execute a shell command
echo	print a message
read	read commands from a file
help	prints a list of available commands together with a brief description
mod_startup	modify the startup file
quit	terminate the execution of ECOTEST
DSR handler	
global	modify global costing data
tu	modify TU related costing data
trans	modify transfer functions of TUs
sigclass	modify type of input ports of the TUs
sigwidth	modify the bundle width of ports and nets
TSP handler	
apply	apply a test method to a testable unit
atsp	execute automatic test strategy planning, either for the whole circuit or for a specified block
atsp_ext	execute automatic test strategy planning only for making all TUs accessible
atsp_int	execute automatic test strategy planning without making the TUs accessible
set	set a parameter to a certain value
reset	reset all parameters to initial values
calculator	start the X11 graphical calculator
save_ts	save the test strategy for later reload or for comparisons to other test strategies
reload_ts	reload a previously stored test strategy
delete_ts	delete a previously stored test strategy
ts_history	print a previously stored test strategy
PRINT handler	
cm	print cost model parameters
tu	print TU related cost model parameters
ts	print test strategy
table	execute sensitivity analysis and plot the related curve on screen
access	print non-accessible input and out ports of the testable units
circuit	print design data
tmdhelp	print a user friendly description of test methods on screen

Table 4: Commands of ECOTEST

The implementation of ECOTEST is based on the EVEREST test strategy planner. The DSR functions remained nearly unchanged. The cost model and the user interface are completely new. The accessibility function of the test strategy planner and the automatic test strategy planning algorithms remained unchanged. The rest of the test strategy planner is completely new.

Where the EVEREST test strategy was written in C, ECOTEST is a mixture of C and

C++. All parts which were taken from the EVEREST test strategy planner are in C, whereas all new functions are implemented object oriented in C++.

The concepts which were taken from the EVEREST test strategy planner were developed in a collaborative project. The concept and the implementation of the design description was developed by the author. The concept of the accessibility functions was jointly developed between Brunel University and Siemens-Nixdorf. All the new concepts and functions implemented in ECOTEST have been developed by the author.

The next section will describe the concepts of the cost modelling techniques and the test strategy planner which have been implemented in ECOTEST. Both concepts are object oriented, which makes the methodologies very flexible and its applicability very general.

5.5. Cost Modelling Techniques

The cost model evaluator calculates the costing parameters, which are based upon the input values. The calculation rules are defined by the operators and the order of the parameter definitions.

The cost models are represented by a directed graph. Each operation is represented by a node, and the edges represent the links to the operators of the operation. Figure 14 presents an example cost model.

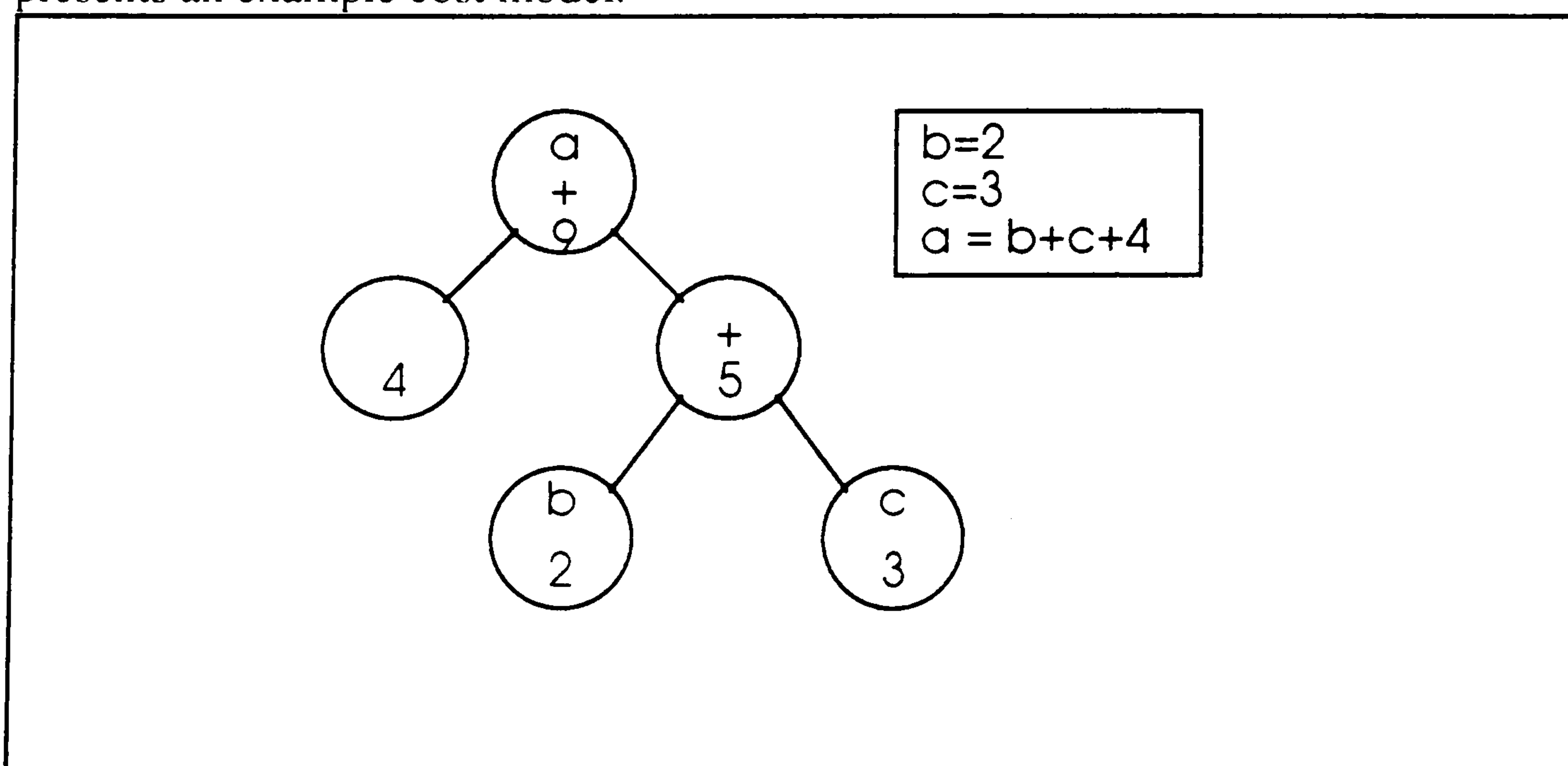


Figure 14: Example of a cost model and its internal representation

Each parameter and each operation are represented by a node. An assignment operator does not create a new node but adapts the parameter name to the assigned term. Each operation node includes a list of operators, which are implemented as pointers to the related nodes. Input nodes are those which do not perform an operation, but which are provided with a certain value. Needless to say, that the input nodes do not have any operands. The nodes are implemented as C++ classes with a inheritance hierarchy. The generic *opnode* class includes the name of the parameter, the list of operands, the list of the users of the node (backward pointer) and the value. In addition, it provides several functions print or to read the name and the value. For each operator, a class is implemented which inherits all data and functions of the opnode class. These operator classes do not provide any data but do provide the calculation function which is specific to the type of the operation. Input nodes also inherit the functions and data of the opnode. In addition, they consist of the input value and the calculation function, which assigns the input value to the parameter value. A special connect node is provided for input parameters, which are connected to a parameter of another cost model. This class provides a pointer to the related parameter, and a calculation function which reads the value of the related parameter and assigns it to its own value.

All nodes of a cost model are stored in an ordered list. The ordering criteria is the order in which the nodes must be calculated. The calculation of the complete cost model is then performed by calculating the nodes of the ordered list. An alternative to this method of calculation is a recursive calculation of each node, starting with the last parameter in the cost model. But in the case of reconvergencies of the graph, this method would lead to multiple calculation of some of the nodes, which reduces the performance of the cost model calculation. Reconvergencies occur, when a parameter is used in more than one term.

The ordered list is part of the class *graph*, which provides in addition to the ordered list of nodes several functions to create the ordered list, to find a single node in the list, to calculate the list, and to print the data of the nodes.

The class *cost_model* is itself a node, which means, that it inherits all data and functions of the class opnode. This allows to handle the hierarchical cost model approach, where a cost model may be built upon several sub cost models. The cost model class contains the following data and functions:

- the graph with all nodes, which are part of the cost model.
- a list of user parameters, which is a subset of all nodes, and which are visible to the user.
- the list of input parameters, which is a subset of all nodes.
- a function to parse the cost model file and to create the cost model graph.
- several functions to access parameters and parameter data of the cost model.
- a function to connect input parameters of the cost model to parameters of another cost model.
- a function to calculate the cost model.

The cost model parser was implemented by using the UNIX tools LEX and YACC. These tools are very efficient in parsing files with a certain syntax and for describing a certain grammar. The software module, which is automatically generated from a YACC description, delivers the nodes of the cost model in the right order, so that an explicit ordering of the nodes is not needed.

The method described above for representing and calculating cost models is very efficient. The calculation time is about 200 times faster than the method implemented in the first version of ECOTEST (see [Dis92]) and only about 10 times slower than a hard coded cost model. A hard coded cost model is an implementation in which the cost model equations are part of the source code. This makes the cost model calculation very fast - a further improvement could only be achieved by a hardware accelerator - but inflexible concerning cost model modifications. The author's approach is an efficient mix of flexibility and performance.

Some of the design parameters are test method dependent. The test method dependent

parameters are the number of not accessible input ports, output ports and bidirectional ports per testable unit. These parameters are used by the external test method cost models. They use these parameters in order to calculate the parameters, which depend on how many ports need to be made accessible. In the case of an external scan path, the number of flip flops to be included, and consequently cost model parameters like the additional gate count, depends on how many ports of the related TU are not accessible yet and need to be made accessible. As discussed earlier, the accessibility of ports depends also on the test methods which are applied to other TUs. Therefore the number of non-accessible ports of a TU is test strategy depend, and needs to be recalculated with each test strategy. In ECOTEST these three parameters are implemented as functions, which means, that the related value is calculated by calling a C-function. This function calculates the accessibility of the TU and returns the related number of non-accessible ports.

In the same way, the number of pins is calculated. A special function calculates the number of input pins, output pins and bidirectional pins. The algorithm to calculate the pin numbers is as follows:

```
set number of pins to number of pins without any test method
for each pin compatibility class do
  select all applied test methods which match to the pin compatibility class
  add the maximum of the additional pins from the selected test methods to the number of pins
end do
```

The test strategy planner is also modelled as a cost model, and it contains the blocks of figure 16 as nodes. The test strategy planner is described in the next section.

5.6. The Test Strategy Planner

In ECOTEST, the test strategy planner is a cost model with some additional attributes and functions. In the object oriented language, this means, that the test strategy planner is a class which inherits the class *cost model*. This means, that it inherits all functions and

data attributes of the cost model, such as the calculate function, the print function, or the ordered list of nodes to be calculated. But in the test strategy planner class, the functions have a different implementation.

The calculation of the total cost for a given test strategy is the objective of the calculation function of the test strategy planner. This calculation is based on the main cost model, on the test method cost models of the applied test methods, and on the design parameters, which are linked to the input parameters of the cost models. Figure 15 shows the structure of the test strategy planner cost model. This example shows the structure for a design with three testable units, TU1, TU2 and TU3.

The list of nodes in the test strategy planner is fixed. It contains the design parameters, which are a cost model and therefore a node, the testable units, which are also nodes, the pin calculation cost model and the main cost model.

The list of nodes of the testable units consists of the internal and the external test method cost model, which are applied to the testable unit. This means, that the testable unit as a cost model changes, when one the two test method changes, i.e. if a new test method is applied to the testable unit. In addition to the cost model attributes and functions, the testable unit contains a list of all applicable test methods, from which the user of the test strategy planner selects the test methods to be applied.

The internal test method cost model is based on the design parameters, and the external test method cost model is based on the design parameters and the test dependent parameters of the internal test method cost model.

The calculation of the number of pins is based on the original number of pins, the additional number of pins from the test method applications and the pin shareability conditions of the test methods, which are applied.

The calculation of the main cost model is based on the design parameters, which provide the values for the design dependent primary parameters, the external test method cost

models per testable unit, which provide the values for the test dependent primary parameters, and the pin calculation cost model, which provides the total number of input pins, output pins and bidirectional pins.

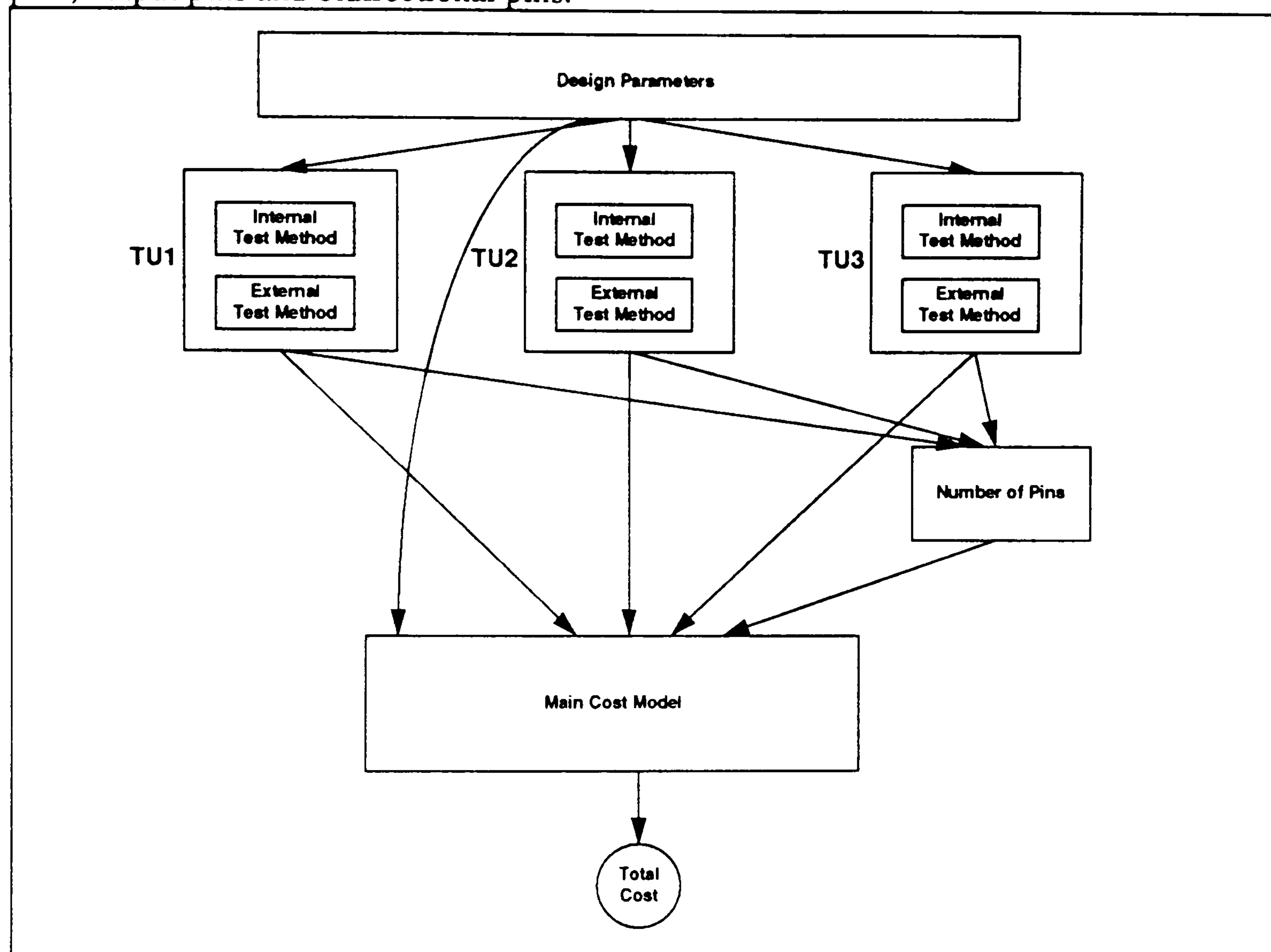


Figure 15: Structure of the cost model calculator

Beside the attributes and functions of the cost model, the test strategy planner includes further attributes and functions. The most important functions will be described subsequently.

The methods for automatic test strategy planning are implemented as functions of the test strategy planner. The implemented algorithm is algorithm 1 of [Dis92] with some extensions. The extensions allow to choose, whether the test strategy planning uses a fixed initial test strategy, or whether the initial test strategy is user defined. In addition, the automatic test strategy planning function can be used for the whole design, for selected single blocks, or the test methods to evaluate can be limited to external test methods or to internal test methods.

An apply function allows a user to apply selected test methods to a user selected testable unit. This function is used for interactive test strategy planning.

The verification function checks, whether the current test strategy violates the design constraints. If a test method is applied, this function is called, and in the case of a violation the test method is reset.

The cost model architecture is fully hierarchical. Each of the blocks in figure 15 is modelled as a cost model. All these cost models are connected as shown in the graph and they form the cost model of the test strategy planner.

5.7. Conclusions

In this chapter, the author has described the test strategy planner ECOTEST. ECOTEST is used for economics based test strategy planning of VLSI circuits. The main concepts and the advancements to the EVEREST test strategy planner have been described. Especially the improved performance of the new method of cost modelling is essential for the techniques which will be described in the following chapter.

The main concern about economics based test strategy planning is the question of the inaccuracy of the economic parameters and the economics model. In order to study this inaccuracy, and to use the methods of ECOTEST with inaccurate data, the author has developed methods to calculate the impact of inaccurate input parameters to the inaccuracy of the total costs. These methods will be described and discussed in the next chapter.

Chapter 6

Sensitivity Analysis

6.1. Introduction

The techniques used in this work to make test strategy decisions are based upon cost estimations. These estimations are subject to inaccuracy, and the estimation of the cost value can be a difficult task. By using a cost model, the costs can be estimated more accurately and easier for the following reasons:

- The cost estimate does not need to be made directly. It is based upon the estimation of parameters, which are easier and more accurately to estimate.
- We make sure that all cost effects, i.e. the cost model parameters, are considered for the cost estimation.

The cost is now based on estimating the values for the input parameters. The accuracy of this estimation depends on how much effort is spent for this task. For example, to estimate the gate count of the design very accurately, one may even have to complete the design task, which is then an expensive estimate of the parameter. For practical reasons, the estimation of the parameter data will be subject to inaccuracy in most cases. This is due to the trade-off between the accuracy of the data and the effort or cost of the data gathering task.

In this chapter, the author describes the methods to study the impact of this inaccuracy on the resulting cost. The parameters are generally studied about their impact on the sensitivity of the cost.

In following section the description of the problem is presented, the need and the gain of this work are discussed, and the different applications of the sensitivity analysis are introduced. The following section describes the method to analyse the variation of all parameters at the same time. In the rest of the chapter the author will present three applications of sensitivity analysis, and a summary of the chapter will be given in last

section.

6.2. Description of the Problem

Test strategy planning as treated in this thesis is based upon economic evaluations, which are performed by evaluating a cost model. Therefore the cost model can be seen as a decision model, where the decision is driven by minimising the resulting cost value. This cost model evaluation requires the provision of input data. The following problems may occur in the process of data acquisition:

- Data gathering can be expensive.
- Due to being estimates, the data can be very inaccurate.
- Some of the input data may not be defined yet; several alternatives are possible.

The problems mentioned above are addressed in many publications. Myers states, that "The difficulty in performing a hypothetical analysis such as this is, of course, that the applicability of results is tied directly to the original assumptions made for the model's input variables." [Mye83]. His solution to this problem is to range for a number of input variables about the basic quiescent operating point. Bellman says, that "Considering the many assumptions that go into the construction of mathematical models, the many uncertainties that are always present, we must view with some suspicion any particular prediction. One way to obtain confidence is to test the consequences of various changes in the basic parameters. This stability or sensitivity analysis is always essential in evaluating the worth of results obtained from a particular model" [Bel61]. Dinkelbach states, that these problems are the major reason, why the practical usage of cost modelling techniques is still very limited [Din69]. This statement is manifested by many critical comments about using cost modelling techniques for test strategy planning. Illman [Ill89] comments the test economics modelling work performed under the ESPRIT project EVEREST [Dis91]:

"In my experience the accurate prediction of design cost is very difficult. ...the cost of using such tools should be well understood". Similar comments were made by several

other experts, e.g. [Aga90]. Extending cost modelling techniques by integrating sensitivity analysis techniques is therefore of major importance.

In [Din69], the sensitivity analysis is defined as follows:
"A sensitivity analysis is the analysis of the relation between the parameters and the decision, i.e. the resulting cost."

Also in [Din69], the following types of sensitivity analysis applications are defined:

1. Analysis of a solved problem to study the impact of the variation of parameters in advance, especially for those parameters which cannot be estimated very accurately. This type of application is also addressed in [Saa61]: "...a sensitivity analysis of the solution to aid concentration of decisions on the parts of the operation whose parameters the solution is most sensitive to.".
2. Some parameters are not defined yet. You can assume several values and study and analyse the result. This can be done by performing several test strategy planning sessions with different input values, or by performing a single test strategy planning session with a specific value for the parameters not defined yet, and to vary these parameters for the optimum test strategy in order to study the impact of the particular parameter.
3. Estimation of uncertainties in plans to study the impact of an unknown parameter on the resulting cost. An example for this application is the impact of the production volume on the economics of a test strategy.
4. After a decision with inaccurate but inexpensively obtained parameters was made, it is important to answer the question of which parameters are sensitive in order to provide for those parameters more accurate information. The result of the sensitivity analysis shows, that for some input data an increase in accuracy is not needed, whereas for other data the accuracy is of essential importance. In this way the cost for data acquisition can be reduced tremendously.

Summing up, it may be said, that the objective of a sensitivity analysis is to study the

affect of parameter changes on the decision to be made.

Sensitivity analysis of type 2 and type 3 have been studied and addressed in detail in previous work ([Dis92], [Dis89], [Var84]). Therefore the author will not deal with these types of sensitivity analysis here in detail.

In previous work ([Dis92], [Dis89], [Var84], [Din69]) sensitivity analysis was performed by varying one parameter and keeping all other parameters constant. This implementation does not show the real impact of inaccuracy of input data. The reason for this is, that the sensitivity for the parameter under consideration is calculated statically, i.e. with a specific value for all other parameters (static sensitivity analysis). If these other parameters are subject to inaccuracy, any other combination of input values - due to its inaccuracy - could lead to completely different sensitivity results. This thesis will describe a novel method of sensitivity analysis, which was developed by the author, and which considers the variation of all parameters during the sensitivity analysis for one parameter. This type of sensitivity analysis will be called *dynamic sensitivity analysis*, because the calculation to analyse the sensitivity is performed by dynamically varying all other parameters.

The basic algorithms and methods used for the dynamic sensitivity analysis will be developed in the next section. In the subsequent sections the author will develop and perform three different sensitivity analysis applications, which are based on the dynamic sensitivity analysis:

- A *general sensitivity analysis* to classify each parameter of the cost model concerning its sensitivity to the cost in general, i.e. independent of a specific case.
- A *iterative sensitivity analysis* will be introduced. This method allows to detail the input data of the cost model iteratively, depending on for which parameter value an increase of the accuracy will increase the accuracy of the total cost estimate.
- The *total variation sensitivity analysis* allows to study the probability density

function of the total cost, which is based upon the inaccuracy of the estimated input values, where the inaccuracy is handled as a variate with a known distribution function.

6.3. Monte Carlo Methods for Dynamic Sensitivity Analysis

The problem to solve by dynamic sensitivity analysis is to handle the inaccuracy of data due to estimations made. This inaccuracy can be seen as a variate with a defined distribution. The type of the distribution is an equal distribution, if the estimate defines a range, or a limited normal distribution, if the estimate is given by a mean value and a deviation from that mean value. The normal distribution is limited mainly for technical reasons. For example, the gate count of a VLSI design cannot be negative. But if the gate count is normal distributed with a $\sigma > 0$, where σ is the standard deviation of the normal distribution, the probability of the gate count to be negative will be greater zero. Therefore the gate count is limited by the value zero.

The following definitions will be used in this chapter:

$\mathbf{x}$ vector of the cost model parameters.

$f(\mathbf{x})$ probability density function of $\mathbf{x}$.

$C'_i(\mathbf{x})$ is the sensitivity of $C(\underline{x})$ against the parameter x_i. $C'_i(\underline{x})$ is given by

$$C'_i(x) = \frac{\partial C(\mathbf{x})}{\partial x_i}$$

$N(\mu, \sigma)$ Normal or Gaussian distribution, where

μ is the mean value, and

σ is the standard deviation.

$U(x_1, x_2)$ Uniform distribution, where

x_1 is the lower limit, and

x_2 is the upper limit.

Based on these definitions, the mean value of the resulting cost, which is based upon variates as input parameters, is given by

$$\mu = \int_{-\infty}^{\infty} C(\mathbf{x}) \cdot f(\mathbf{x}) d\mathbf{x} \quad (1)$$

The variance of the total cost is given by

$$\sigma^2 = \int_{-\infty}^{\infty} (\mu - C(\mathbf{x}))^2 \cdot f(\mathbf{x}) d\mathbf{x} \quad (2)$$

These equations are multiple integrals, where the number of integrals to solve is given by the dimension of $\mathbf{x}$. The problems to be solved with the sensitivity analysis are the following:

1. Determine the mean value and its variance by solving the integral of the cost model in order answer the questions:
   "How sensitive is the resulting cost concerning the inaccuracy of the input values?"
   "What is the sensitivity behaviour of the resulting cost considering the variation of one parameter with inaccuracy of the other parameters?"
2. Solve the differential of the cost model in order to answer the following question:
   "What is the maximum sensitivity of the resulting cost concerning the variation of one parameter with constraints for the other parameters?"
   This problem is an optimisation problem, which can be solved by an extrema analysis.

Problem 1 can be solved analytically. But this would imply to solve the multiple integrals defined above over a very complex, discontinuous function, i.e. the cost model. Problem 2 cannot be solved analytically without modifications, which are related to discontinuities in the cost model, because they cannot be differentiated. Nevertheless, within the continuos regions the differentiation of the cost model could be performed analytically. However this analytical approach is not practical for the following reasons:

The analytical method implies the solution of very complex integrals and differentials, which is strongly tied to the related function. But the function changes from test strategy

to test strategy, because parts of the function are related to the test methods, which form the test strategy. This means, that the function to differentiate or to integrate is different for each test strategy, and therefore the integral or differential to be solved analytically is different from test strategy to test strategy. The multitude of test strategies, which is based upon the multitude of test methods (see chapter 2), would require the solution of many complex integrals and differentials, which is extremely complex and therefore error prone. Secondly, this approach is very inflexible, because for each modification of the cost model or for each new test method to be considered, the whole manual integration and differentiation work has to be redone.

For these reasons, the author has chosen the Monte Carlo method for solving the integration problem and the optimisation problem:

- The method is independent from the function to analyse and therefore it is very flexible concerning its application to changing functions.
- The nature of the input parameter values are variates. Monte Carlo simulation is also based on variates[Rub86], and therefore the method fits very well to the nature of the underlying problem.
- The method can be used for both the integration and the optimisation problem [Ham65], [Rub86].

In the following the author will introduce the Monte Carlo method as used here. In the subsequent sections the methods will be adopted to the related problems to be solved.

"Problems handled by Monte Carlo are of two types called probabilistic or deterministic according to whether or not they are directly concerned with the behaviour and outcome of random processes" [Ham65]. The type of Monte Carlo used in this thesis is deterministic, because the process, which is simulated by the cost model, is deterministic. The behaviour of the cost model is not random, and the behaviour of the input data is also known. Also, in theory, the problem can be solved deterministically, as shown above. Hammersley [Ham65] defines deterministic Monte Carlo as a numerical

solution of a deterministic problem. Deterministic Monte Carlo simulation is also called *Sophisticated Monte Carlo.*

The essential feature of Monte Carlo simulations is that a random variable is replaced by a corresponding set of actual values, having the statistical properties of the random variable ([Ham65]). This random variable can be part of the process under consideration, or it can be modelled from a deterministic variable. The actual set of values, so called the random numbers, are used to analyse the process. This procedure is called the Monte Carlo simulation. Consider the following example:

A process is defined by the cost model and some input data, which are normal distributed. The question to answer is: What is the probability that the resulting cost exceed a certain limit? This question can be answered through Monte Carlo simulation as follows:

- Generate random numbers for the normal distributed input parameters having the statistical properties of the given normal distribution
- Calculate the related resulting cost value
- Repeat the above procedures many times
- Measure the probability of exceeding the limit by calculating the relation of resulting cost values exceeding the limit to the number of calculations.

Instead of solving the complex multiple integral of the cost model over the distribution function of the input parameters, we simply measure the result by performing Monte Carlo simulation. Summing up, the Monte Carlo method is based upon the generation of random numbers relating to given distribution properties, and the author will present in the next section the methods to compute these random numbers for the distribution types used in this work.

6.3.1. Computation of Random Values for a Given Distribution Function

The Monte Carlo simulation is based on the generation of random variates X with a known cumulative distribution function F_X. This function can be given deterministically

or as a table of observed stochastic data. X can be generated by the inverse transformation method, which transforms uniformly distributed values U between 0 and 1 - for which standard functions exist in C - into random variates with F_X as follows:

$$X = F_X^{-1}(U) \quad (3)$$

It remains now to determine the inverse cumulative distribution function from the cumulative distribution function. This is simple for uniformly distributed functions between a and b, and there are several approximation methods described in [Ham65] for the most important distribution types, such as the normal distribution or the exponential distribution. The methods differ mainly in computation efficiency and accuracy in the approximation of F_X. In this thesis the computation of the variates is not significant compared to the computation effort for calculating the cost model. Therefore the main attention in the selection process among the different methods was spent on the accuracy of the method concerning the approximation of the distribution type. The distribution functions needed are the uniform distribution and the normal distribution. A general uniformly distributed value U_{ab} from a to b is generated from a (0,1) uniformly distributed value U_{01} as follows:

$$U_{ab} = a + (b-a) \cdot U_{01} \quad (4)$$

Three methods are proposed in [Ham65] to compute random numbers for a normal distribution. The author has implemented all three methods and has compared them concerning computation times and approximation accuracy. They will be described in the following sections.

6.3.1.1. Marsaglia Table Method

This method is based on a lookup table, which represents the inverse cumulative normal distribution function [Ham65]. The table consists of the $F^{-1}(U)$ values for selected values of U between 0 and 1 in increasing order of U. The interval size, i.e. the difference between two successive values, is constant within five ranges. Table 5 provides the

interval size for the range of U between 0 and 1. A normal distributed random number is now generated as follows:

- compute a value Y, which is uniformly distributed between 0 and 1.

  find the two succeeding values in the table, for which

$$U_{i+1} \geq Y \geq U_i$$

- Compute the normal distributed value N(0,1) by linear approximation

$$N(0,1) = F^{-1}(U_i) + (F^{-1}(U_{i+1}) - F^{-1}(U_i)) \cdot \frac{(Y - U_i)}{U_{i+1} - U_i} \quad (5)$$

- Compute $N(\mu,\sigma)$ by

$$N(\mu,\sigma) = \mu + N(0,1) \cdot \sigma \quad (6)$$

Range of U	Interval size
0.00 - 0.05	0.002
0.05 - 0.20	0.005
0.20 - 0.80	0.010
0.80 - 0.95	0.005
0.95 - 1.00	0.002

Table 5: Ranges for Marsaglia Table

6.3.1.2. Box and Miller Method

This method produces normal deviates in independent pairs N_1, N_2 as follows:

$$\begin{aligned} N(\mu,\sigma)_1 &= \sqrt{(-2 \cdot \ln(U_1))} \cdot \cos(2\pi \cdot U_2) \cdot \sigma + \mu \\ N(\mu,\sigma)_2 &= \sqrt{(-2 \cdot \ln(U_1))} \cdot \sin(2\pi \cdot U_2) \cdot \sigma + \mu \end{aligned} \quad (7), (8)$$

where U_1 and U_2 are independent, (0,1) uniform deviates U(0,1).

6.3.1.3. Central Limit Theorem Method

This method relies on the central limit theorem (see [Kre75], or any other statistics book):

$$N(0,1) = \sum_{i=1}^{12} U_i - 6 \quad (9)$$

The central limit theorem method produces a normal deviate with a mean value of 0 and

a variance of 1. 12 uniform deviates are needed to get a variance of 1.

6.3.1.4. Test of the Accuracy of the Methods

The accuracy of the methods above is tested by using the χ^2- Test [Kre75]. The author has implemented this test as follows:

Test of Methods for Normal Distribution

- compute 1,000,000 random numbers by using the method to be tested with a normal distribution of N(0,1).
- divide the scale for the random values - from -∞ to +∞ - into 12 ranges as listed in table 6.
- for each range do:
  - count the number b_i of random numbers, which are in the range
  - compute the optimum number b_i from F(x) by using the C-function *erf()*
  - calculate $\chi_i^2 = (b_i-e_i)^2/e_i$
- end do
- calculate mean value of χ^2 by $\chi^2 = \Sigma\, \chi_i^2/12$

-∞ to -2.5	0.0 to 0.5
-2.5 to 2.0	0.5 to 1.0
-2.0 to -1.5	1.0 to 1.5
-1.5 to -1.0	1.5 to 2.0
-1.0 to -0.5	2.0 to 2.5
-0.5 to 0.0	2.5 to ∞

Table 6: Ranges for normal distribution test

Test of methods for uniform distribution

- compute 1,000,000 random numbers U(0,1) by using the C-function *drand48()*.
- divide the scale for the random values - from 0 to 1 - into 10 uniform ranges.
- for each range i do:
  - count the number b_i of random numbers, which are in the range
  - the exact number e_i for each range is
  - $e_i = 1/10*1.000.000 = 100.000$
  - calculate $\chi_i^2 = (b_i-e_i)^2/e_i$

- end do
- calculate mean value of χ^2 by $\chi^2 = \sum \chi_i^2/10$

The results are presented in table 7. The test was run on a 12 MIPS HP Apollo 400t workstation.

Method	Marsaglia	Box and Miller	Central Limit Theorem	drand48
χ^2	0.97	1.01	21.42	0.78
CPU time	99.37 s	102.2 s	692.13 s	61.22 s

Table 7: Result of χ^2 test

The methods of the normal distribution are based upon the generation of (0,1) uniformly distributed values by using the C-function drand48(). As one can see in table 7, this computation does not give exactly distributed numbers, and the χ^2 value depends on the initial seed. Therefore the author has performed each test to calculate χ^2 ten times with different initial seeds, and χ^2 was set to the mean value. Table 7 shows, that the Marsaglia table method and the Box and Miller method do not differ significantly in its χ^2 value and in CPU time, whereas the central limit theorem method is about 20 times worse in its χ^2 value, and about seven times worse in CPU time. So, taking the χ^2 value as the first selection criteria and the CPU time as the second selection criteria, both the Marsaglia table method and the Box and Miller method may be selected. The author has chosen the Marsaglia table method, because this method is independent of the machine dependent implementation of the calculation of sine and cosine.

6.3.2. Correlation of Input Parameters

When the input parameters of the cost model are varied randomly, we have to take into account, that some of the parameters are correlated to each other. E.g., the number of cells of the design is correlated to the gate count of the design. This means, that if the gate count is high, there is a high probability, that also the cell count has a high value. If the correlation is not taken into account for the generation of the random numbers, we can get unrealistic or even invalid combinations of input parameters. For example, if the

gate count is uniformly distributed between 1000 and 10000, and the cell count is uniformly distributed between 300 and 3000, there is a probability of 0.22, that the cell count will be higher than the gate count, if these two values are independent. As we know, this cannot happen in reality, because in cell based designs a cell contains at least one gate equivalent. In order to get realistic results for the Monte Carlo simulation, we must take into account the correlation of parameters for the computation of the random numbers. If we see the input parameters as a vector **x**, we can generate a random number vector as follows:

$$\mathbf{x} = \mathbf{C}^{*} \cdot \mathbf{x}_0 + \mu \qquad (10)$$

where **C*** is given by

$$\mathbf{C}^{*} \cdot \mathbf{C}^{*T} = \mathbf{C} \qquad (11)$$

where **C** represents the covariance matrix. A proof is given in [Arm82]. Because **C** is a positive definite matrix, **C*** can be derived by using the Cholesky decomposition method (see [Arm82]).

This method is used to calculate correlated multi variates, and it allows a correlation between all variates. The correlation is defined by the covariance matrix. In the case of the cost model, we have the following correlations:

gate count <=> cell count

gate count <=> number of flip flops

number of flip flops <=> sequential depth

This leads to the following covariance matrix for these three parameters:

gates:	μ_{11}	μ_{21}	μ_{31}	0
cells:	μ_{12}	μ_{22}	0	0
flip flops:	μ_{13}	0	μ_{33}	μ_{43}
sequential depth:	0	0	μ_{34}	μ_{44}

where μ_{ii} is the variance σ_i^2 of the single parameter i, and μ_{ij} is the covariance of parameter i and parameter j. This is a special case of the covariance matrix, where the parameters are correlated in pairs. In this case the method can be simplified as follows:

$$
\begin{aligned}
X_{Gate} &= \mu_{Gate} + X_{0Gate} \cdot \sigma_{Gate} \\
X_{Cell} &= \mu_{Cell} + (X_{0Gate} \cdot \rho_{Cell} + X_{0Cell} \cdot \sqrt{1-\rho^2_{Cell}}) \cdot \sigma_{Cell} \\
X_{FlipFlop} &= \mu_{FlipFlop} + (X_{0Cell} \cdot \rho_{FlipFlop} + X_{0FlipFlop} \cdot \sqrt{1-\rho^2_{FlipFlop}}) \cdot \sigma_{FlipFlop} \\
X_{SeqDepth} &= \mu_{SeqDepth} + (X_{0FlipFlop} \cdot \rho_{SeqDepth} + X_{0SeqDepth} \cdot \sqrt{1-\rho^2_{SeqDepth}}) \cdot \sigma_{SeqDepth}
\end{aligned} \quad (12, 13, 14, 15)
$$

This can be proven by using (10) and (11) for each pair of the correlated parameters.

In order to be able to generate the correlated random numbers, we must know the distribution functions X* per parameter, the mean value μ per parameter, the root to the variance σ per parameter, and the correlation factor ρ per correlation. X*, μ and σ are variables. The correlation factor was derived by the author using data about existing designs. The calculation of the correlation factors is described in appendix A.

6.4. General Sensitivity Analysis

A general study of the cost model is performed by the author in order to classify all parameters of the cost model concerning their sensitivity impact on the total cost value. This analysis is a special case of type 4 described in section 6.2., and it gives a general idea, which parameters must be estimated very accurately, even if the cost for this estimation is high, and for which parameters a rough estimate may fulfil the accuracy requirements concerning the resulting cost value. An outcome of this study may even be, that some of the parameters can be neglected for the cost evaluation. This fact would allow a simplification of the cost model by cutting out the effect of these parameters. The refined cost model would provide the same results with lower costs in data acquisition and test strategy planning. Data acquisition costs are reduced, because the number of parameters, for which data need to be provided, is reduced. Due to the reduced complexity of the refined cost model, calculation effort in terms of CPU time and therefore the related test strategy planning costs are reduced. A second point of cost reduction concerning test strategy planning efforts is the fact, that a reduced number of parameters and variables may lead to a reduction in the search space. If, for example, two test methods differ only in parameters, which are no more present in the refined cost model, then these two test methods are identical concerning their impact on the test

strategy planning process, which reduces the size in one dimension of the search space - the test method alternatives - by one element.

The sensitivity classification of the parameters will be performed by estimating the following characteristics of each parameter:

1. The mean value and the variance of the sensitivity of each parameter in a constrained space $D \in R$ of the input parameters:

$$\mu = \int_{-\infty}^{+\infty} C'(\mathbf{x}) \cdot f(\mathbf{x}) d\mathbf{x}$$
$$\sigma^2 = \int_{-\infty}^{+\infty} (C'(\mathbf{x}) - \mu)^2 \cdot f(\mathbf{x}) d\mathbf{x} \qquad (16, 17)$$

where μ is the mean, σ^2 is the variance, $C'(\mathbf{x})$ is the sensitivity of the total cost, and $f(\mathbf{x})$ is the probability density function of the input parameters of the cost model.

2. The maximum sensitivity of the total cost for each parameter in a constrained space $D \in R$ of the input parameters:

$$S_{max} = \max(C'(\mathbf{x})) \forall \mathbf{x} \in D \in R)$$

where $\mathbf{x}$ is the vector of the input parameters, and $C'(\mathbf{x})$ is the sensitivity of the cost model.

In this thesis all three characteristics will be estimated by performing a Monte Carlo simulation. Due to the fact, that parts of the cost model are test method dependent, i.e. the cost model is different for different test strategies, the sensitivity analysis was performed for three different representative test strategies. These are *no DFT*, *scan path* and *circular self test path*.

6.4.1. Estimation of Mean Value and Variance of Sensitivity

6.4.1.1. The Algorithm

The integrals to calculate the mean value and the variance will be estimated by a Monte

Carlo simulation as follows:

$$\mu = \frac{1}{n} \cdot \sum_{i=1}^{n} C'(\mathbf{x}_i)$$

$$\sigma^2 = \frac{1}{n-1} \cdot \sum_{i=1}^{n} (C'(\mathbf{x}_i) - \mu)^2 \qquad (18), (19)$$

where μ is the mean value, σ^2 is the variance, $C'(\mathbf{x})$ is the sensitivity, n is the number of simulations and $\mathbf{x_i}$ is the vector of random numbers for the input parameters of the cost model. The random numbers per parameter are uniformly distributed. The range of the distribution covers all typical values. They are defined in table 8.

Parameter Name	Abbreviation	Lower Limit	Upper Limit
Number of cells	cells	3	64000
Complexity exponent	cexp	0.8	1.0
Number of gates	cgate	1,000	100,000
Labour cost rate	costrate	500 $	5000 $
Performance complexity	cperf	1	3
CPU time	cputime	0	1,000 h
Design centre cost rate	descentrate	10,000 $	40,000 $
Computer equipment rate	equrate	25 $	1000 $
Designer's experience	exper	0	100
Required fault coverage	fcreq	70%	100%
Average number of faults per gate	fpg	2	5
Constant factor concerning designer's productivity	kdes	1	2
Constant factor concerning total productivity	kp	70,000,000	90,000,000
Manual test generation time per fault	mtgtime	0.05 h	1 h
Number of flip flops	dffs	6	8,000
Design originality	or	0	1
Productivity of the CAD system	pcad	1	5
Percentage of design time an external design centre is used	percuse	0%	100%
Number test pattern for which test application cost increases	pms	64,000	640,000
Test application cost per step	pps	0	25 $
Production unit cost per gate	puc	0.5 $	2 $
Sequential depth	seqdepth	0	103
Production volume	vol	1,000	1,000,000

Table 8: Distribution characteristics of cost model parameters

The sensitivity $S_j(x_i)$ must be calculated for each input parameter. The sensitivity value is defined here as the relative difference in the total cost of a relative difference of the

analysed input parameter:

$$S_j = \frac{C'(x_j \cdot s) - C'(x_j)}{C'(x_j)} \quad (20)$$

where s is the sensitivity factor, which defines the increase of the parameter x_i as a percentage of x_j. The author has chosen this definition of sensitivity instead of the gradient dC/dx_j, because the gradient does not allow direct comparison between different parameters, because they are based on different measure units, and consequently the gradients are not comparable. For example, a gradient of 1 for the parameter "originality" is less sensitive than a gradient of 1 for the gate count. In both cases the meaning of a gradient of one is, that a variation of the parameter value by one unit will vary the resulting cost by one unit. This means, that a variation of the originality from 0.0 to 1.0 will have the same impact on the total cost as a variation of the gate count from 10,000 to 10,001 gates, if the gradient is constant for both parameters within the varied range. The relative difference of the parameter as defined here is much better linked to the original question of a sensitivity analysis, which is "how much does the variation of a certain parameter impact the total cost?". The sensitivity factor s will be set to 1.01, 1.1 and 1.2, which means a variation of the parameter by 1%, 10% and 20%. This allows to analyse the sensitivity for small, medium and large variations. For some parameters, especially those going into step like functions, e.g. the parameter "pin memory size", a larger variation may cause a significant increase in the sensitivity, where a smaller variation will lead to no sensitivity at all. The following example should illustrate this:

The test application costs are calculated by a step like function (see chapter 4). If the number of test patterns is 1000 and the value of the pin memory size is uniformly distributed between 500 and 1500, the probability, that a 1% increase of the pin memory size will lead to an increase in the test application costs is

$$p = \frac{(1000 - \frac{1000}{1.01})}{1500 - 500} = 0.0099 \quad (21)$$

If the increase of the pin memory size is 10%, the probability of a cost increase is

$$p = \frac{(1000 - \frac{1000}{1.1})}{1500 - 500} = 0.0909 \quad (22)$$

In the case of a 1% variation, a Monte Carlo simulation will result in test application increase in average every 100 simulations. For a 10% variation, this increase will be in average for every 11 simulations. Therefore a 1% variation will need more simulations as the 10% variation to achieve the same accuracy.

6.4.1.2. Estimation Error of the Monte Carlo Simulation

Monte Carlo simulations are estimates which converge to the exact value of the integral. The number of simulations needed to achieve a satisfying accuracy of the Monte Carlo estimate can be derived from the standard error. The standard error is defined to

$$\sigma_S = \frac{\sigma}{\sqrt{n}} \quad (23)$$

where σ_S is the standard error. If the variance converges to a certain value, the standard error will converge to zero. Figures 16 through 18 show the standard error as a function of the number of simulations for all parameters. The value printed is normalised to the root of the variance, σ, for 10,000 simulations. The simulation was performed with a sensitivity factor of 1.01 and "no DFT" as test method except for the parameter dffs, were the test method "scan path" was chosen. This exception was made, because the number of flip flops (dffs) does not affect the total cost for the "no DFT" test strategy.

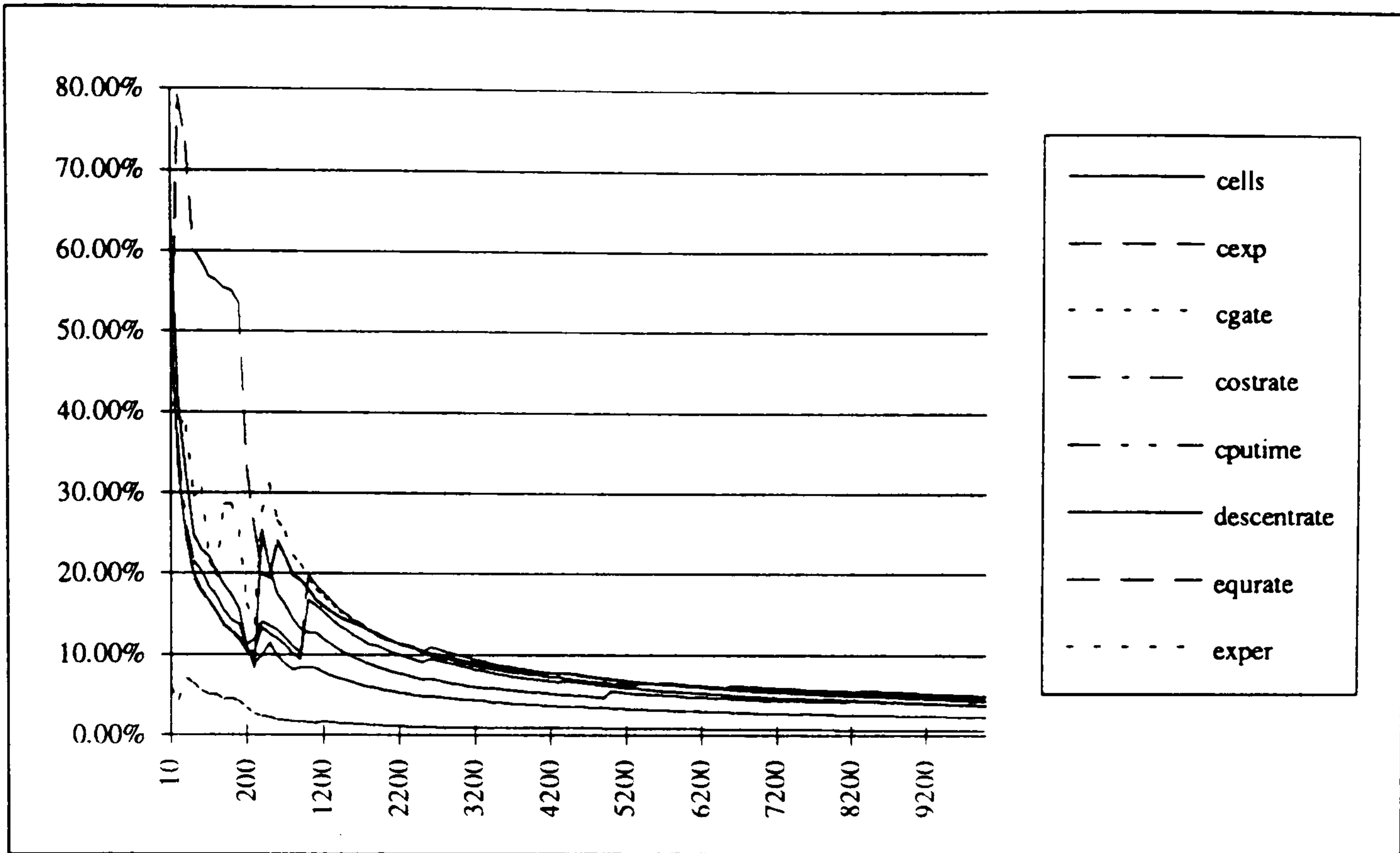


Figure 16: Standard error of Monte Carlo simulation for group 1

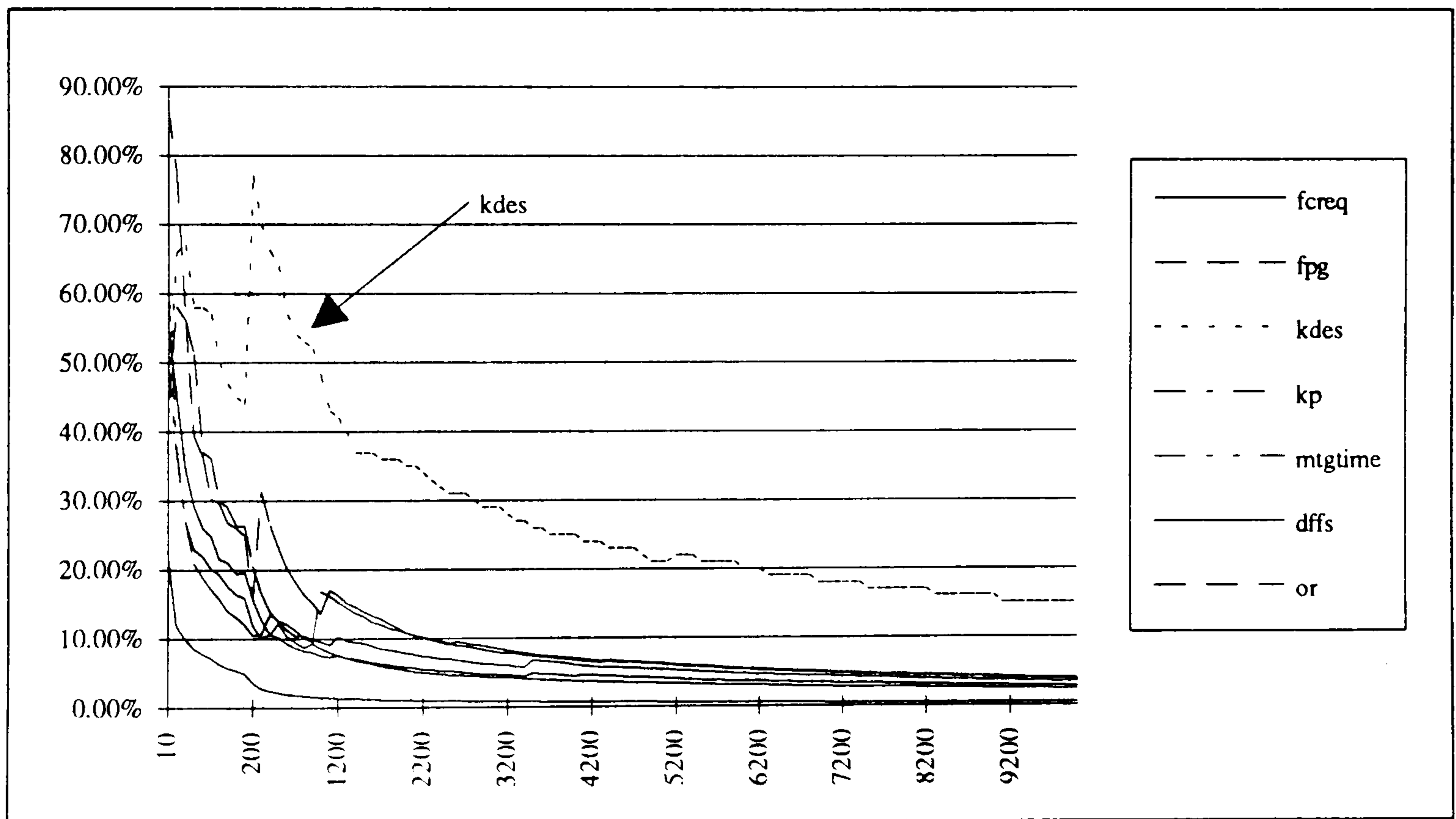


Figure 17: Standard error of Monte Carlo simulation for group 2

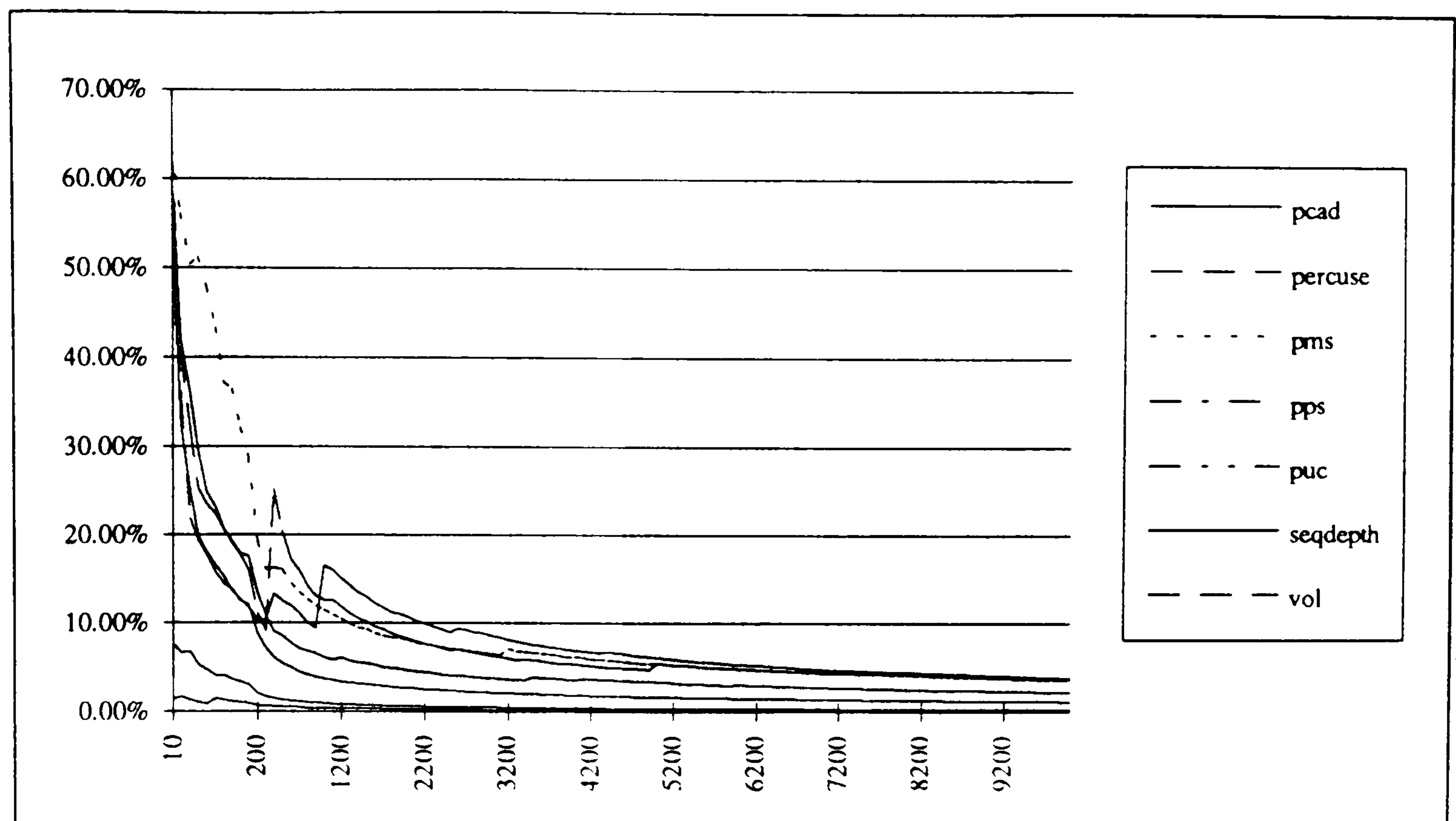


Figure 18: Standard error of Monte Carlo simulation for group 3

The figures show for all parameters except one good convergence to zero for the standard error. The value remains under 10% for 3,200 simulations. The exception is *kdes*. This parameter will need special attendance concerning its accuracy. However, for most parameters the standard error remains at a significant level even with 10,000 simulations. This means, that the Monte Carlo estimate of the mean value and the variance has some inaccuracy. Therefore a calculation of the confidence interval of each estimate will be performed. The confidence interval [Kre75] defines a range, in which the real value will be with a certain probability. This probability is called the *confidence coefficient*. The confidence interval can be calculated for the mean value, if the sample size, the variance of the sample and the confidence coefficient is given. The author has selected a confidence coefficient of 0.998, which means, that the probability, that the mean value will be in the confidence interval, is 99.8%. In figures 20 through 25 the confidence intervals of the mean values of the sensitivity are printed. 10,000 simulations were performed. In these graphs, the white square marks the upper bound and the black square marks the lower bound. These show, that the average confidence interval remains very small for 10,000 simulations in all situations. For most parameters the lower and upper bound are so closed, that the lower bound (black square) is covered by the upper bound. Therefore the author has selected 10,000 simulations as sufficient for this

analysis.

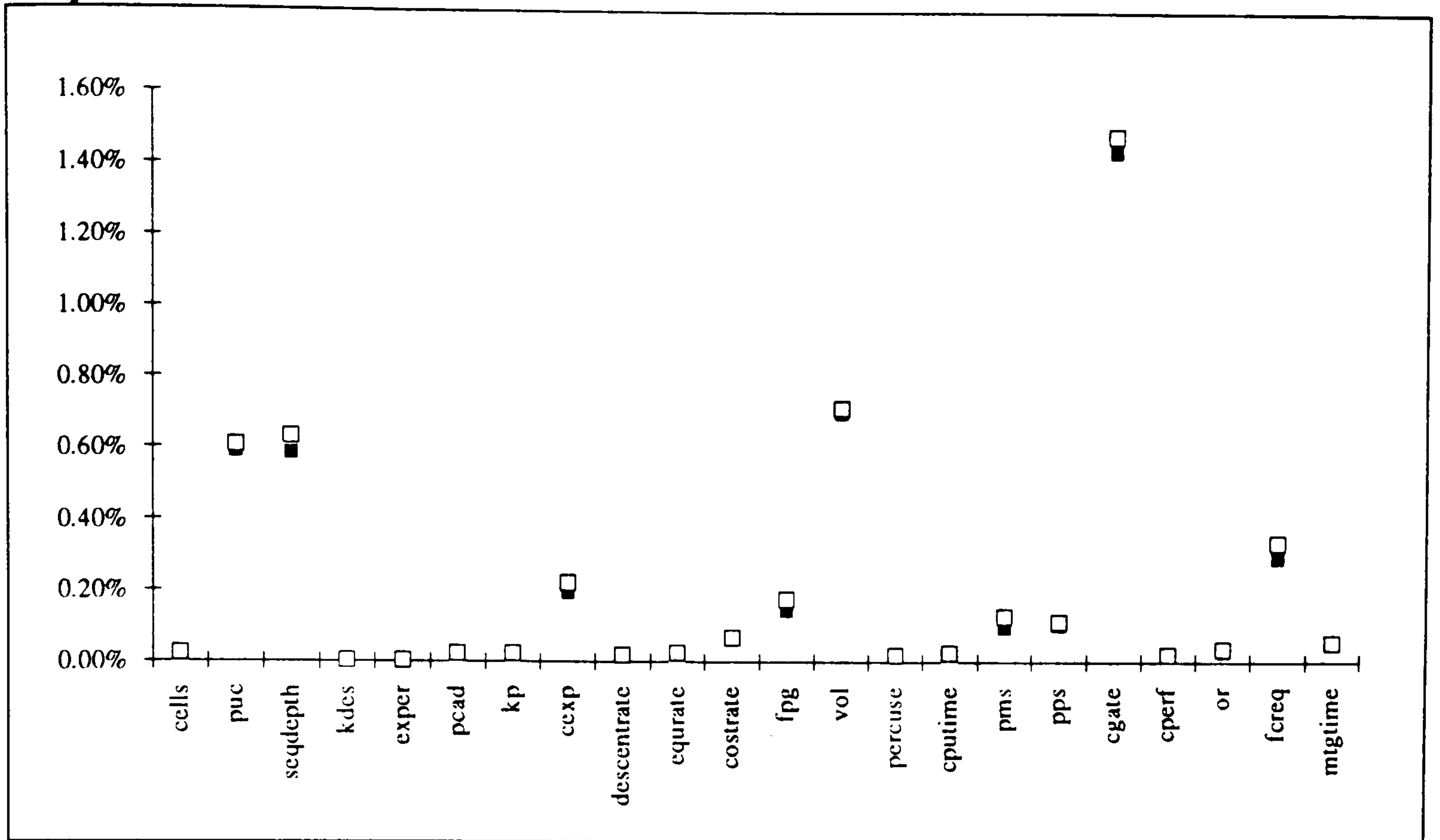


Figure 19: Mean sensitivity for a 1% variation and no DFT

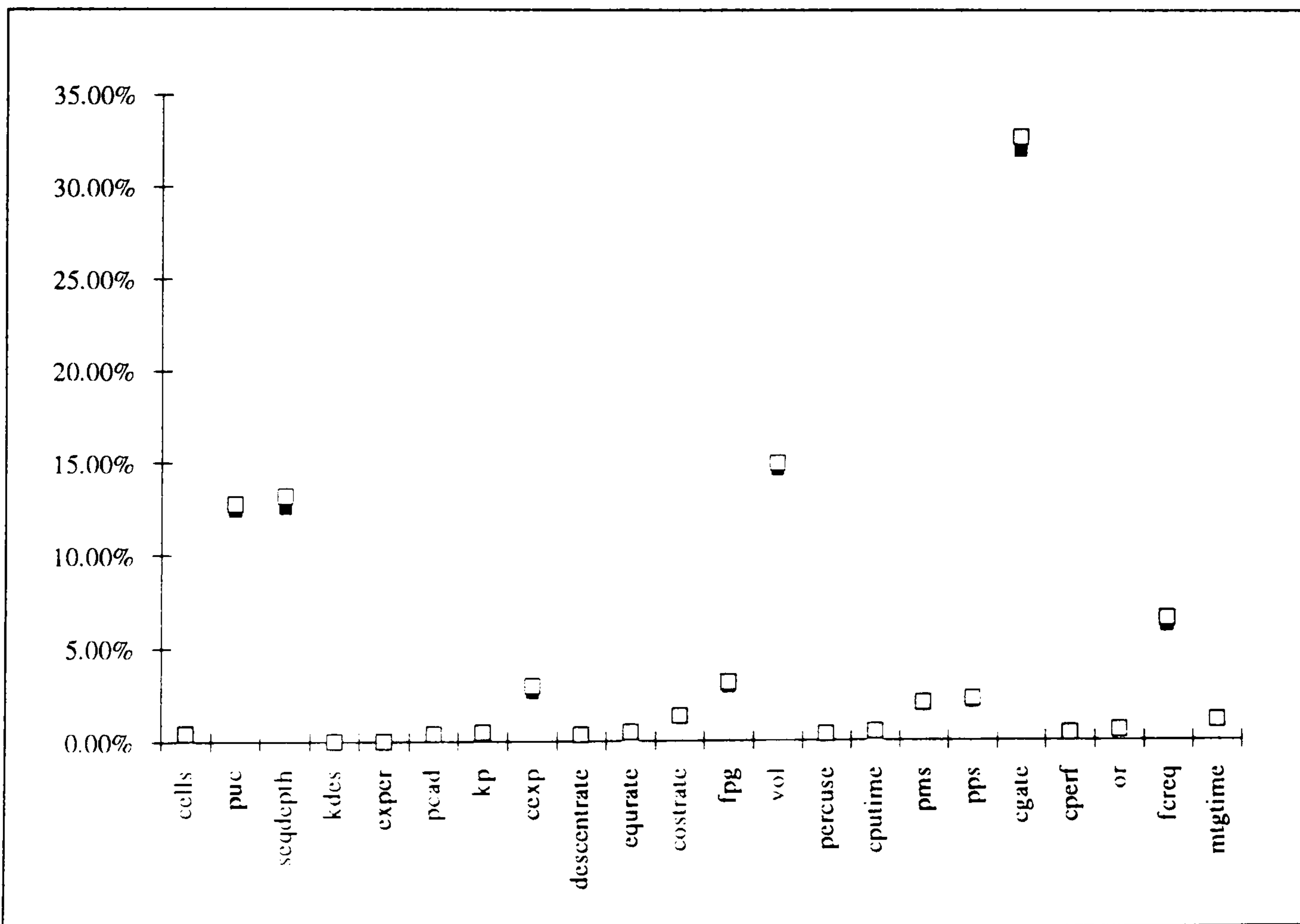


Figure 20: Mean sensitivity for a 20% variation and no DFT

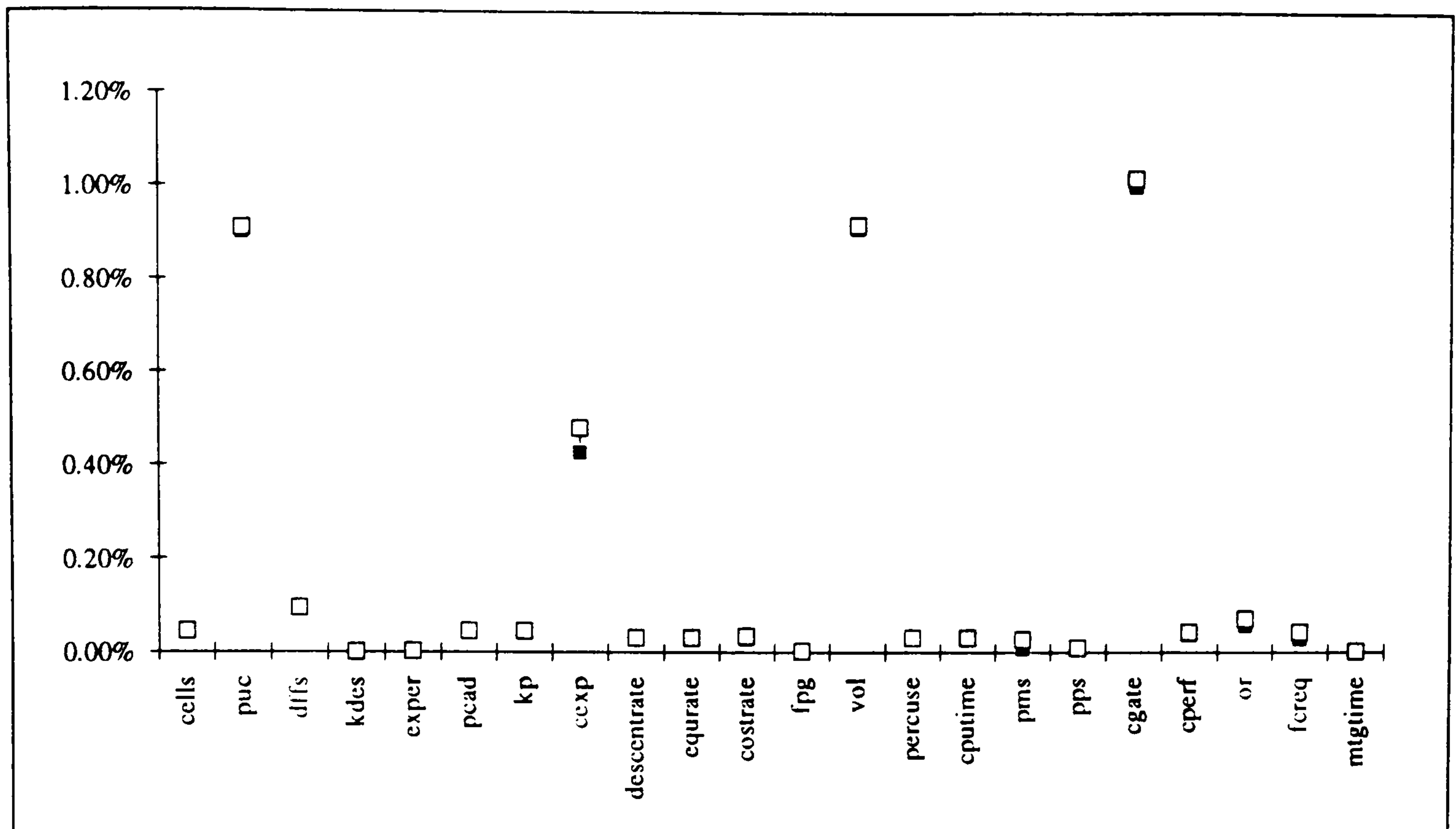


Figure 21: Mean sensitivity for a 1% variation and scan path

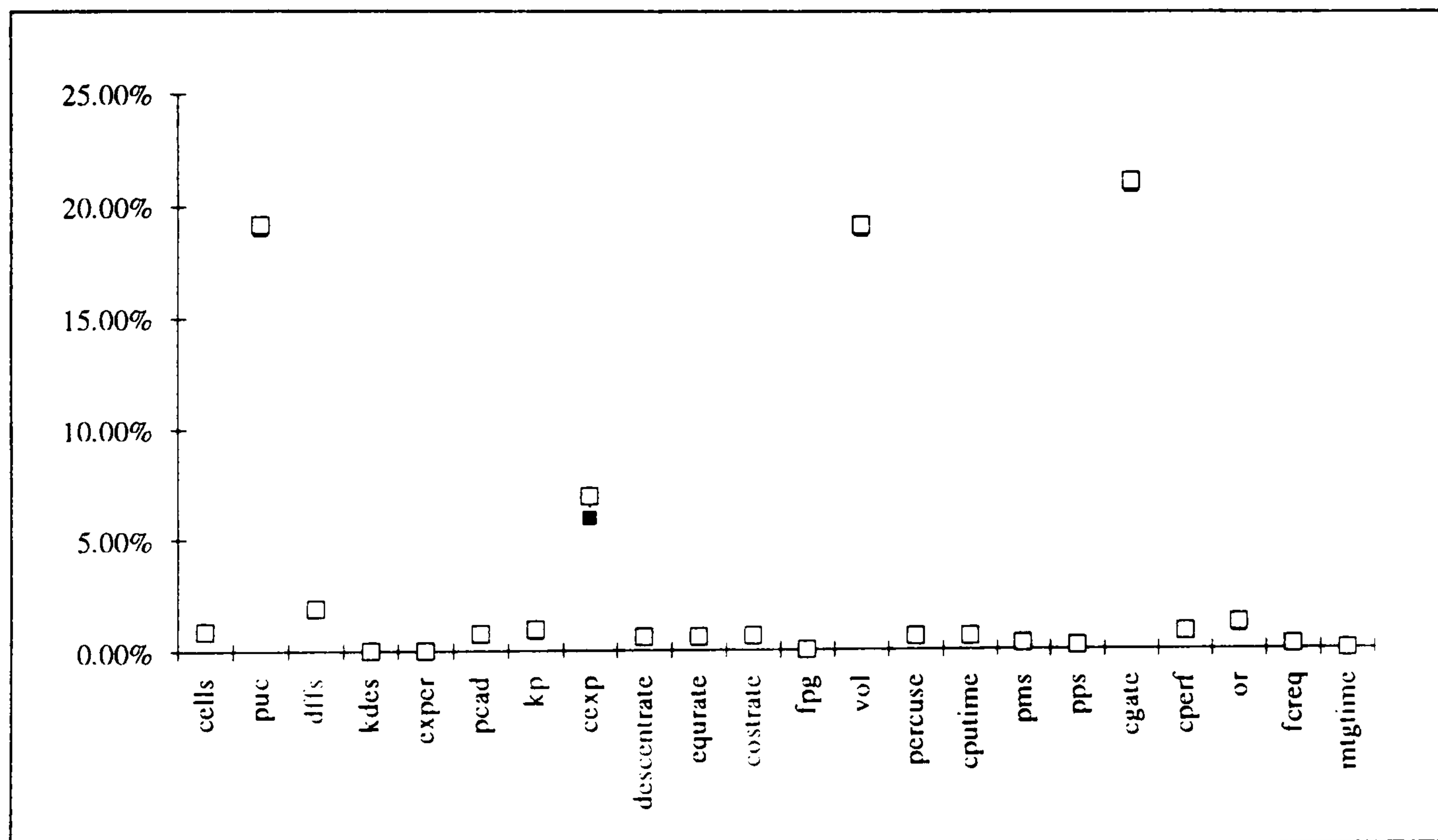


Figure 22: Mean sensitivity for a 20% variation and scan path

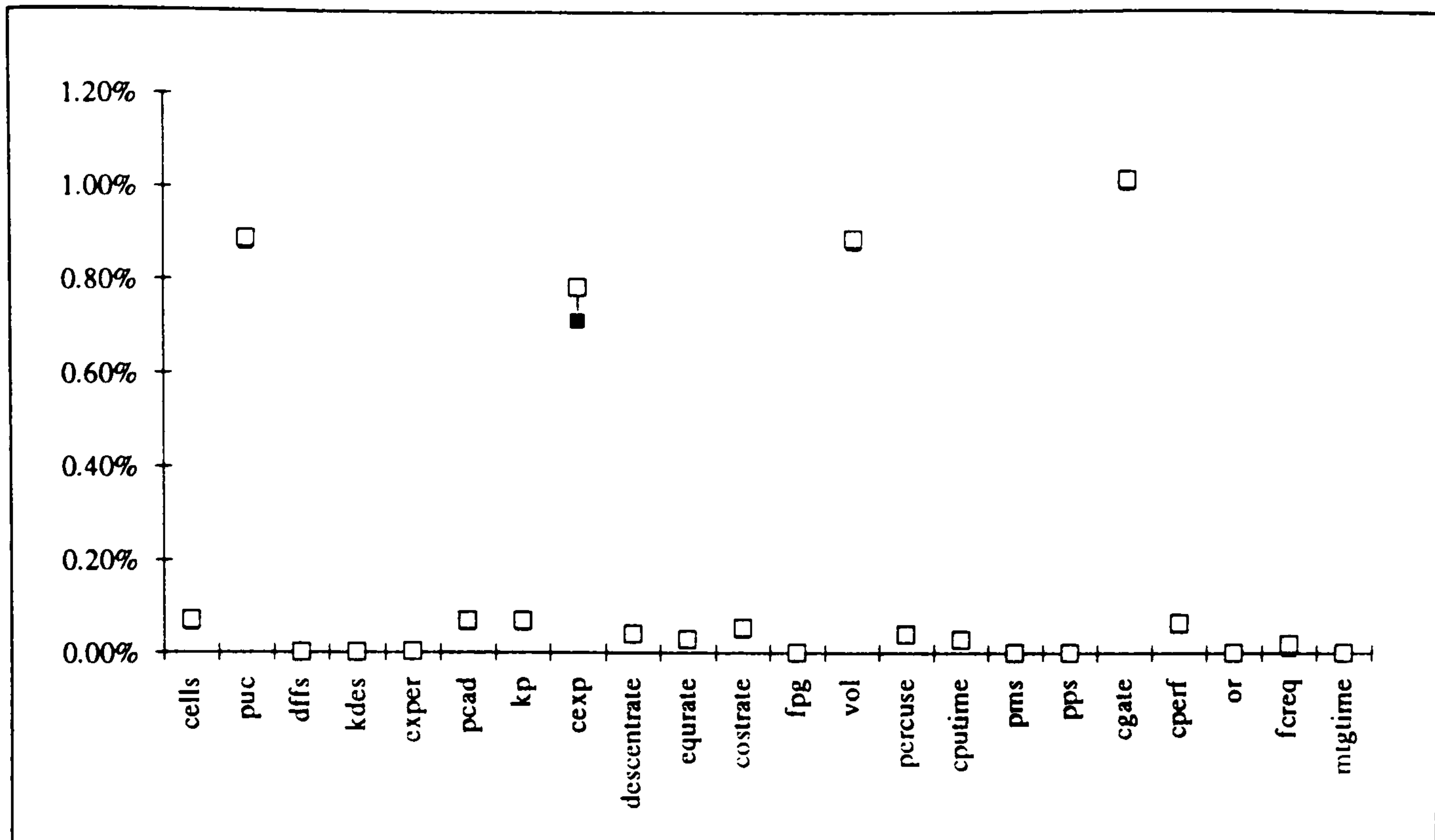


Figure 23: Mean sensitivity for a 1% variation and self test

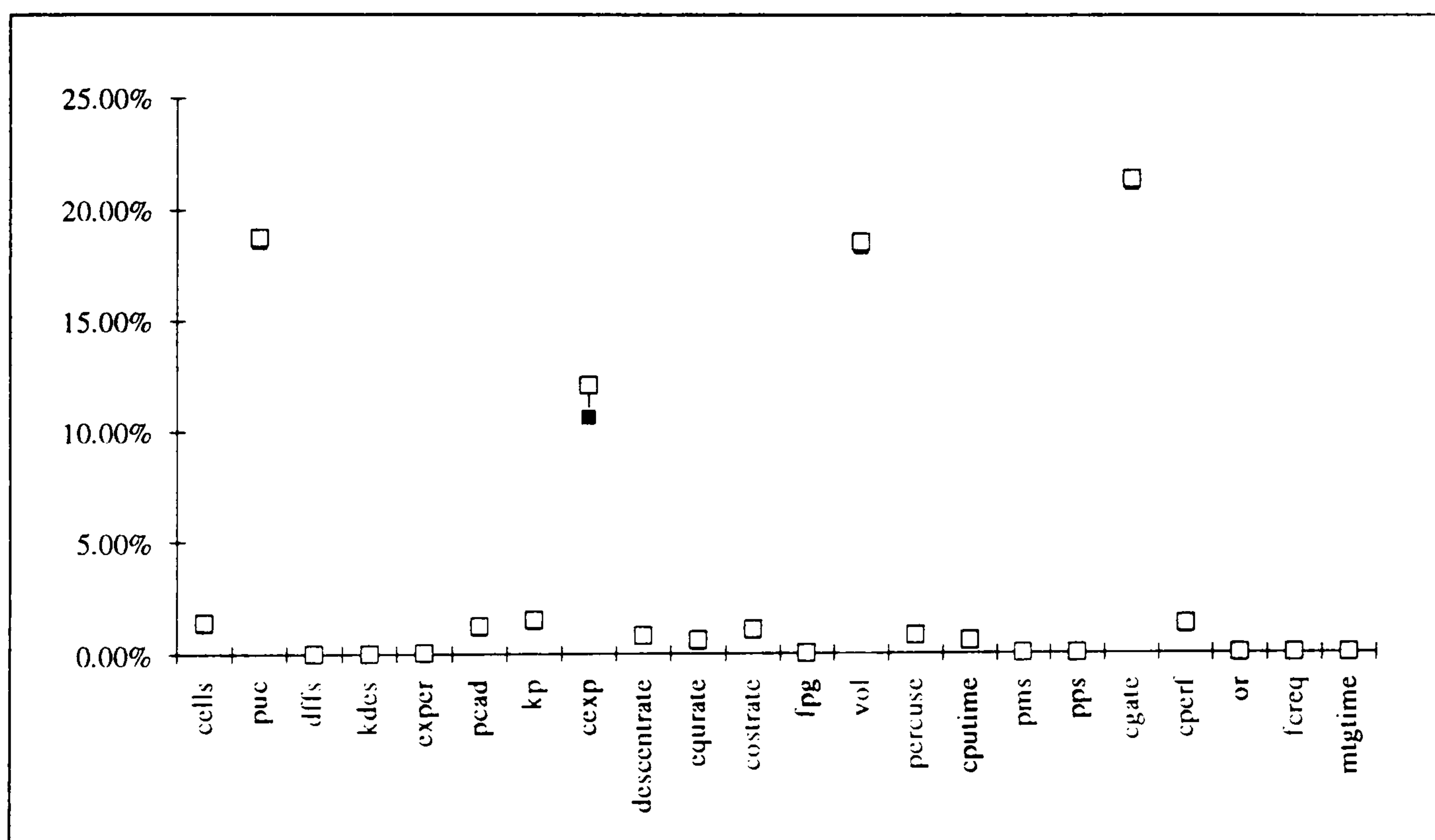


Figure 24: Mean sensitivity for a 20% variation and self test

6.4.1.3. Results

In figures 25 through 35 for each parameter the range is printed, in which the sensitivity will be in 99% of all cases. This range is defined as follows:

$$\text{probability (sensitivity} < \text{lower bound)} = 0.005$$

$$\text{probability (sensitivity} > \text{upper bound)} = 0.995$$

The range is marked by the vertical line and the mean value of the sensitivity is marked

by a small horizontal bar. We distinguish between the three cases no DFT, scan path and self test. For each case the parameters are classified as follows:

Class	no DFT	Scan Path	Self Test
High Mean, High Maximum	cgate, seqdepth	cgate	cexp,
High Mean, Av. Maximum	puc, vol	puc, vol	cgate, vol, puc
Av. Mean, High Maximum	cexp, fcreq, fpg, costrate, pms, pps	cexp	
Low Mean, Av. Maximum	cells, pcad, kp, descentrate, equrate, percuse, cputime, or, cperf, mtgtime	cells, or, fcreq, pms, cperf, cputime, pcad, kp, equrate, costrate, descentrate, percuse,	cells, pcad, kp, descentrate, equrate, costrate, percuse, cputime, cperf
Low Mean, Low Maximum	kdes, exper	dffs, kdes, exper, fpg, pps, mtgtime	dffs, kdes, exper, fpg, pms, pps, or, mtgtime

Table 9: Parameter classification concerning its sensitivity

See table 8 for the meaning of the parameter abbreviations. From this classification and figures 25 through 35 the following conclusions can be drawn:

- The parameters number of gates, sequential depth, production unit cost per gate and production volume are the most important parameters for the sensitivity and accuracy of the cost estimation.
- The *complexity exponent* becomes the most sensitive parameter for self test. This may be due to the fact, that this parameter affects only design cost and self test is the most design intensive test method here.
- The parameters constant factor concerning designer's productivity, designer's experience, number of cells, productivity of the CAD system, constant factor concerning productivity, design centre cost rate, computer equipment rate, percentage of design time an external design centre is used, CPU time, design originality, performance complexity and manual test generation time per fault do not affect the sensitivity of the total cost very much in most cases. Therefore, inaccurate data for these values may fulfil the accuracy requirements of the total cost. Especially the constant factor concerning designer's productivity, the designer's experience and the number of flip flops are candidates for a refinement

of the cost model, which will not use these parameters or will take an average value for them.

- Only one of those parameters, which purely affect the design cost, is important. This is the *complexity exponent*. This fact arises the question, whether the Monte Carlo simulation is dominated by the production volume dependent cost. The author has therefore performed a Monte Carlo simulation, where the production volume was varied between 1,000 and 2,000 (Low Volume). The results are presented in figures 34 through 35. They show, that the classification of the parameters is still the same, whereby the difference of the mean sensitivity between the parameters is smaller. The main difference between this case and a production volume varied over the full range is a larger variance and a higher maximum value for the parameter *complexity exponent*. This shows, that the assumption, which was made, does not follow.

The next question to answer now is, whether some of the "insensitive" parameters can be neglected such that the cost model can be refined. This question can be answered by determining the maximum sensitivity as a worst case. This analysis will be performed in the next section.

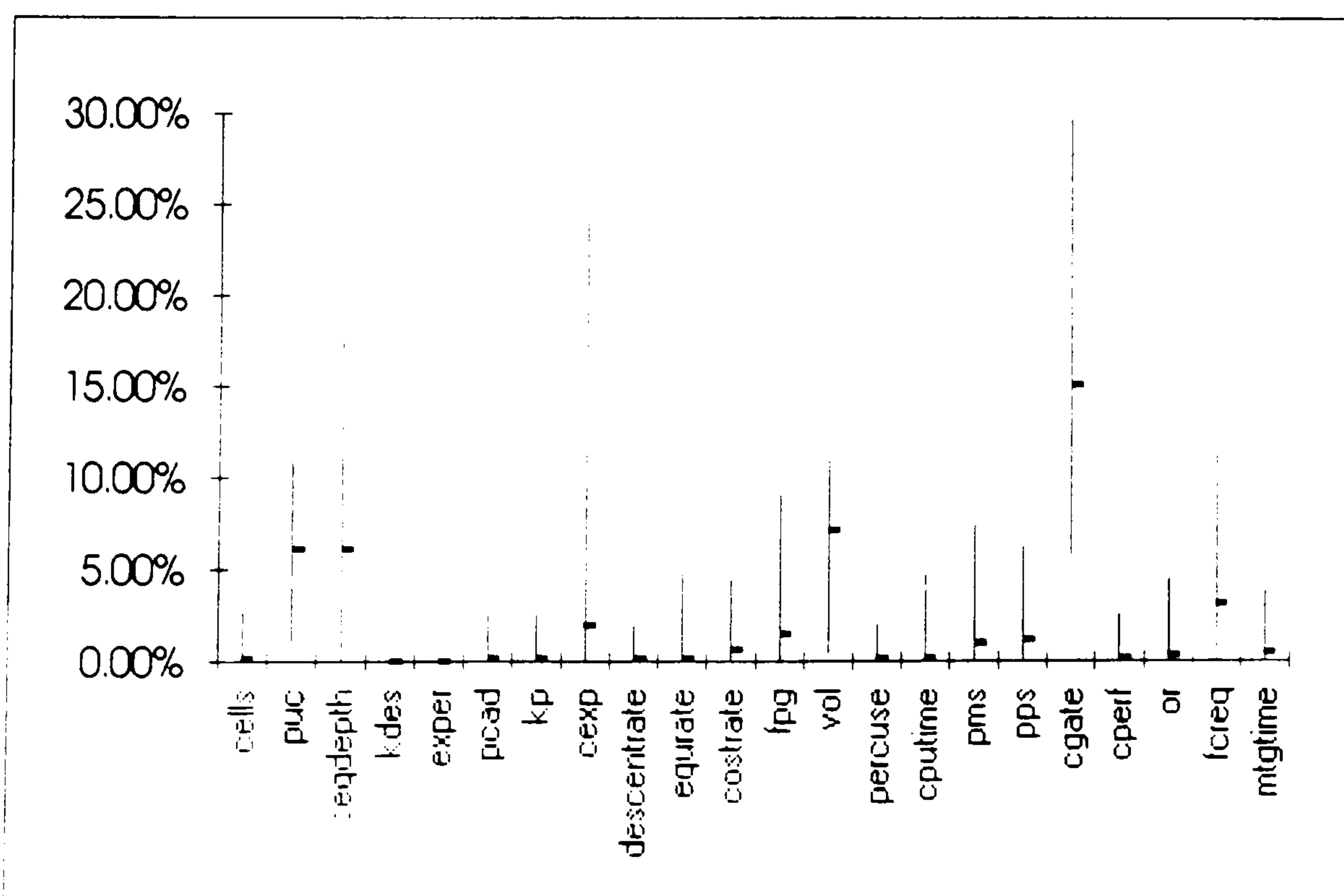


Figure 25: 99% range of sensitivity with a 1% variation, no DFT

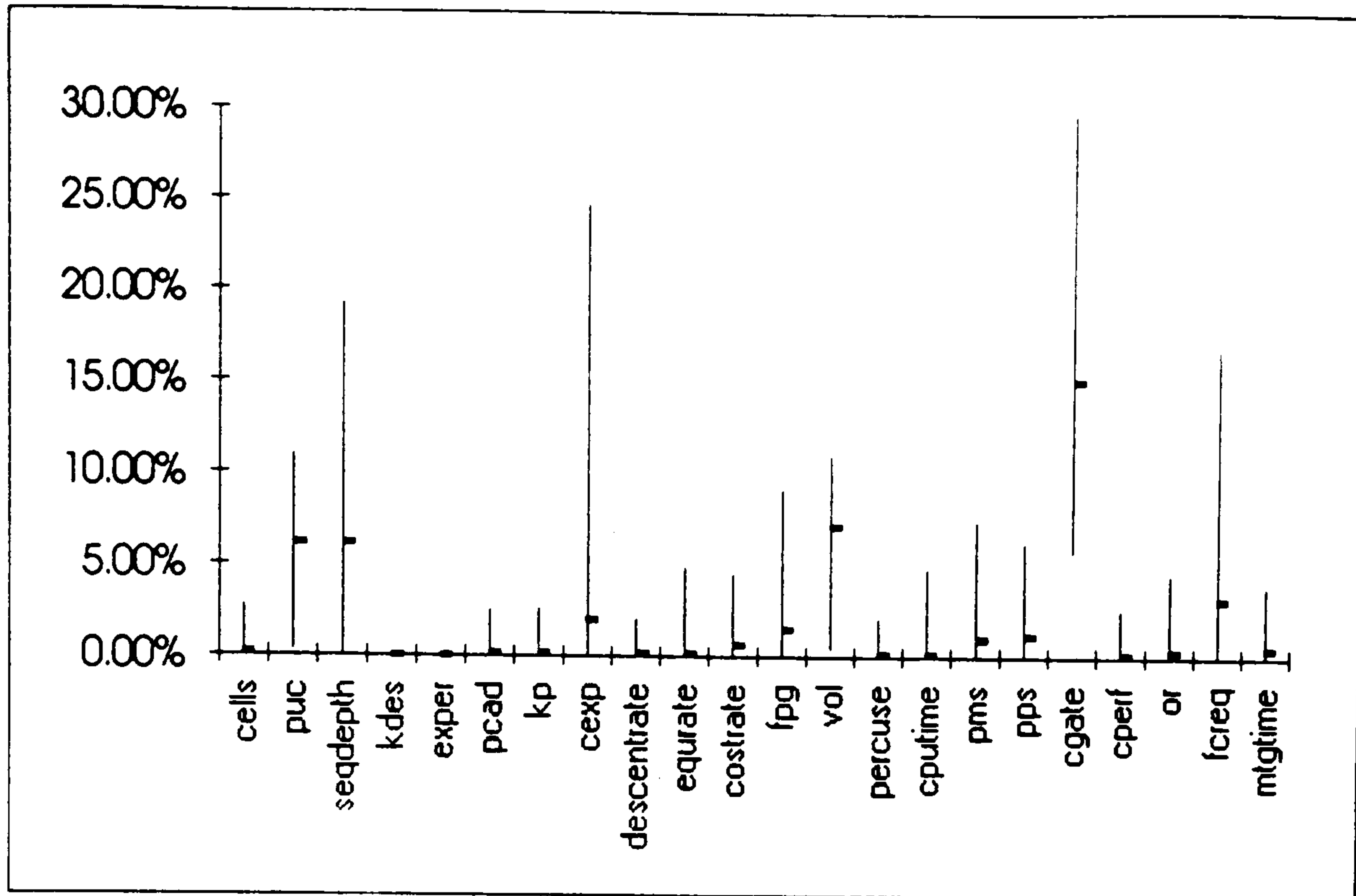


Figure 26: 99% range of sensitivity with a 10% variation, no DFT

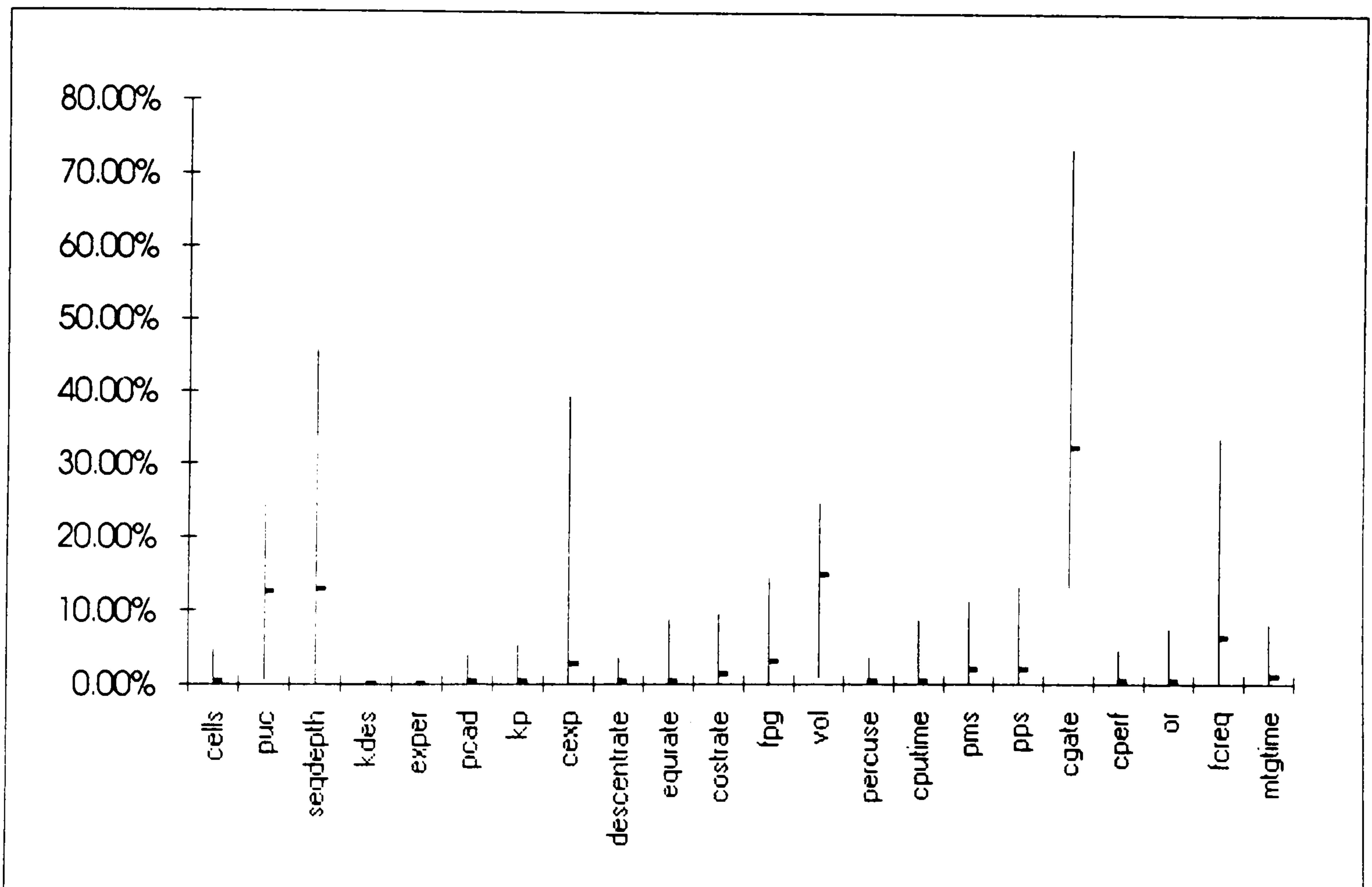


Figure 27: 99% range of sensitivity with a 20% variation, no DFT

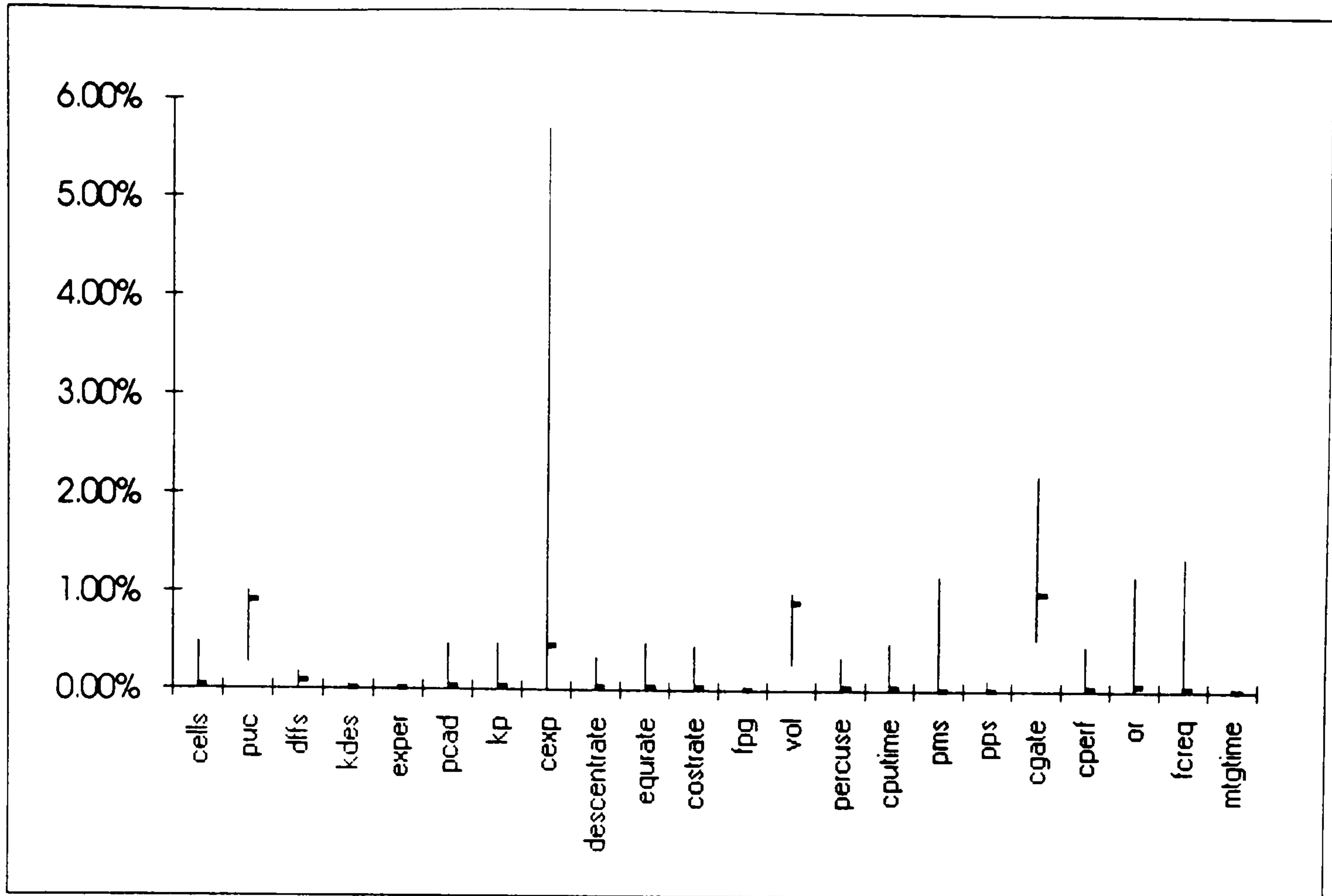


Figure 28: 99% range of sensitivity with a 1% variation, scan path

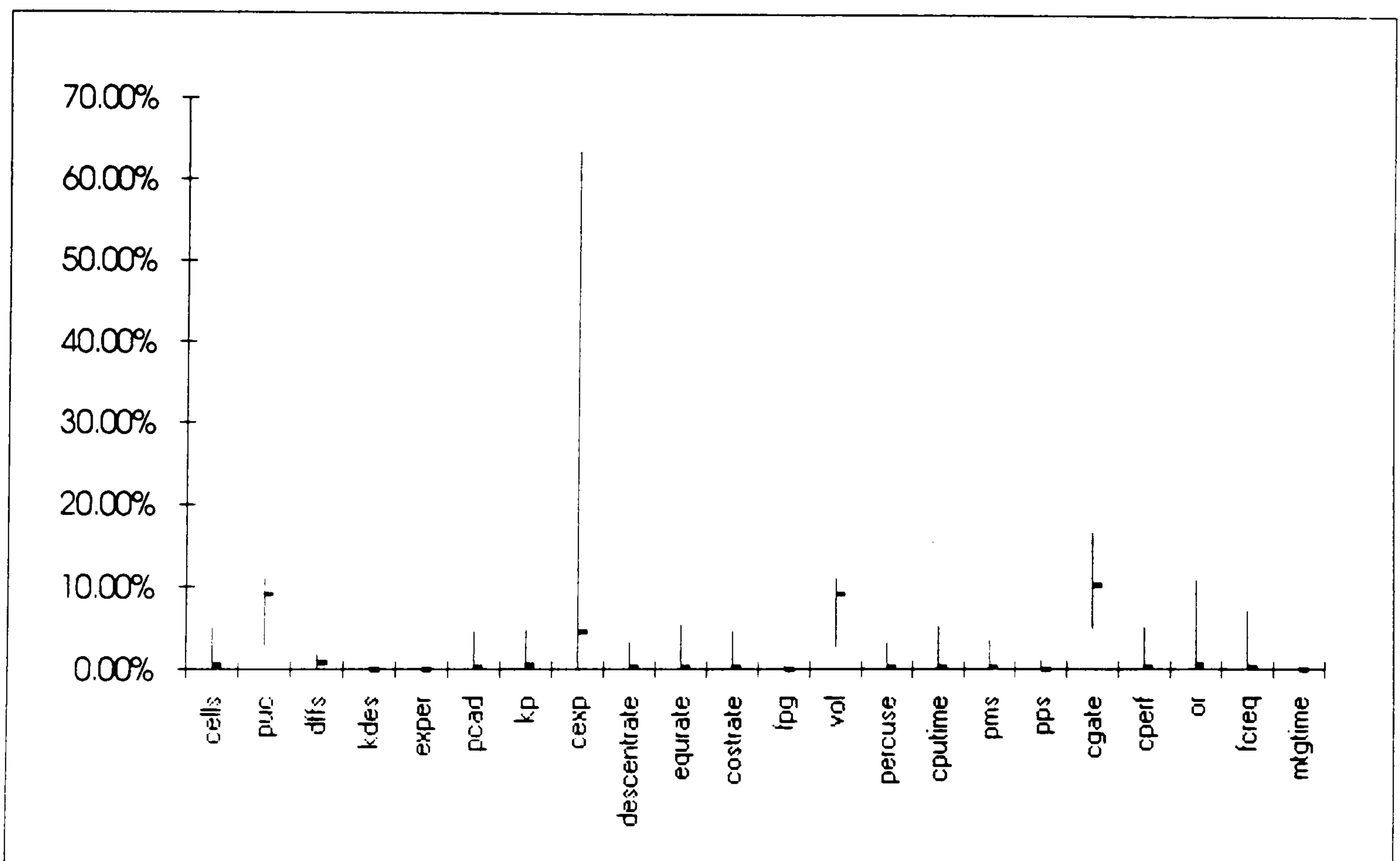


Figure 29: 99% range of sensitivity with a 10% variation, scan path

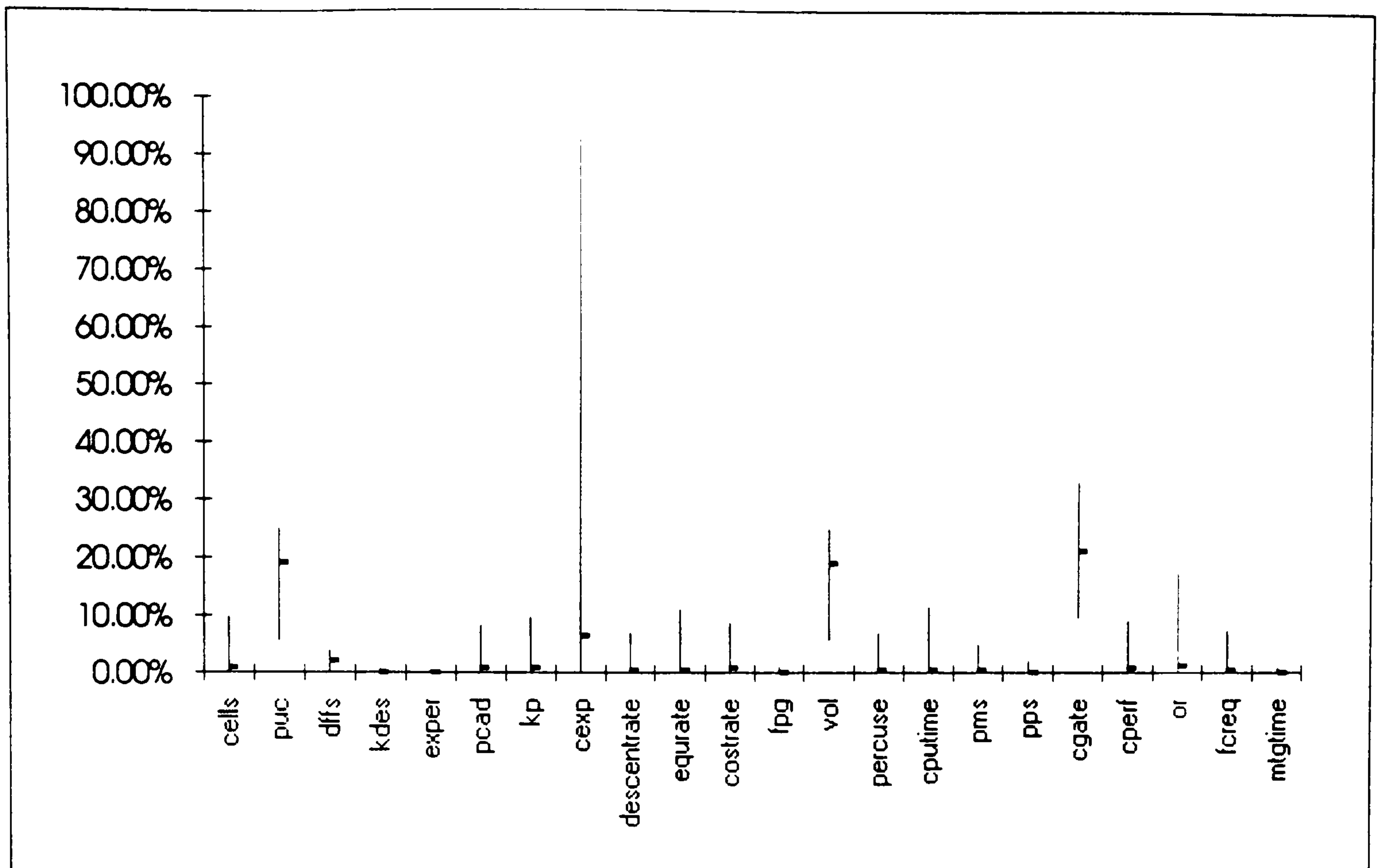


Figure 30: 99% range of sensitivity with a 20% variation, scan path

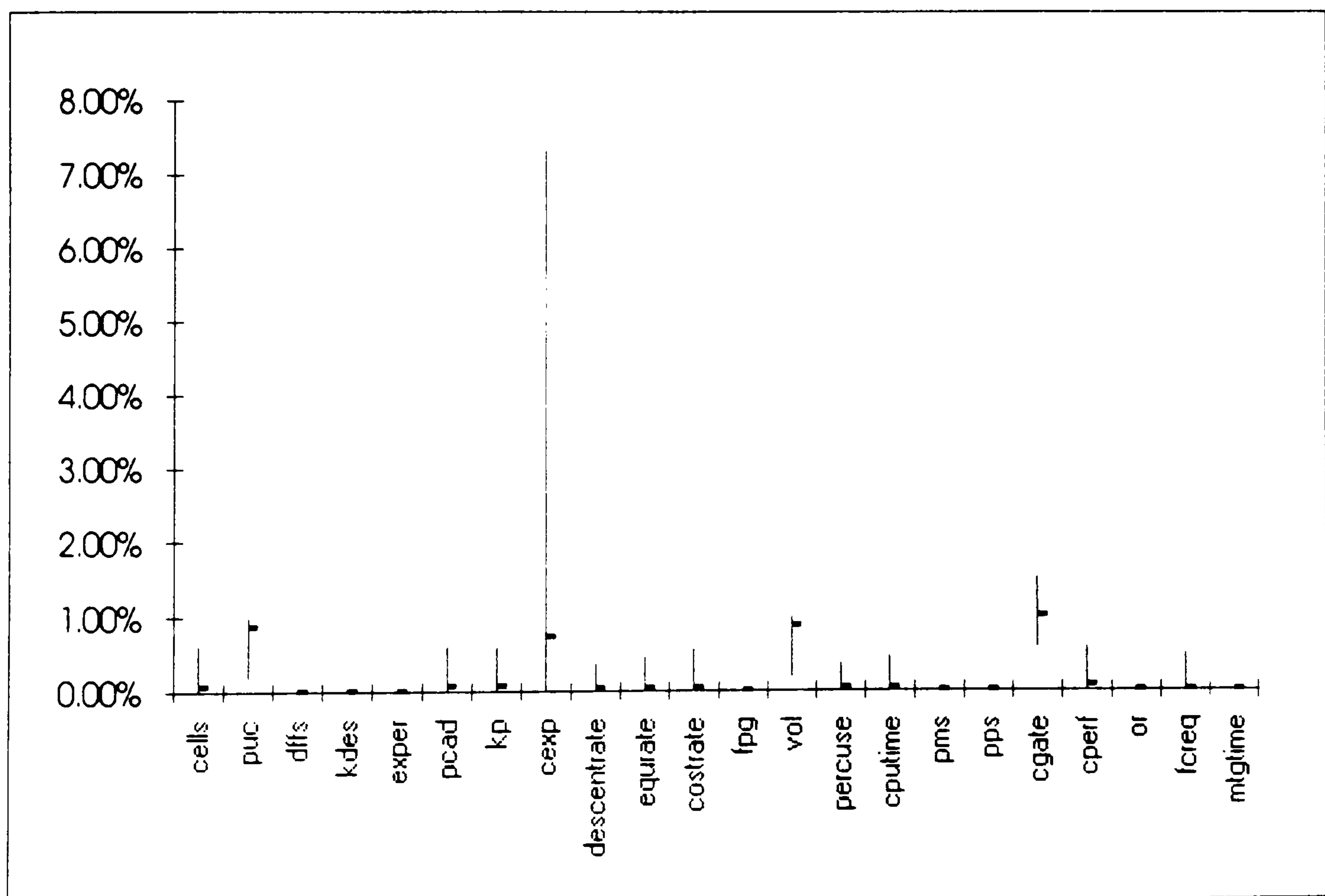


Figure 31: 99% range of sensitivity with a 1% variation, self test

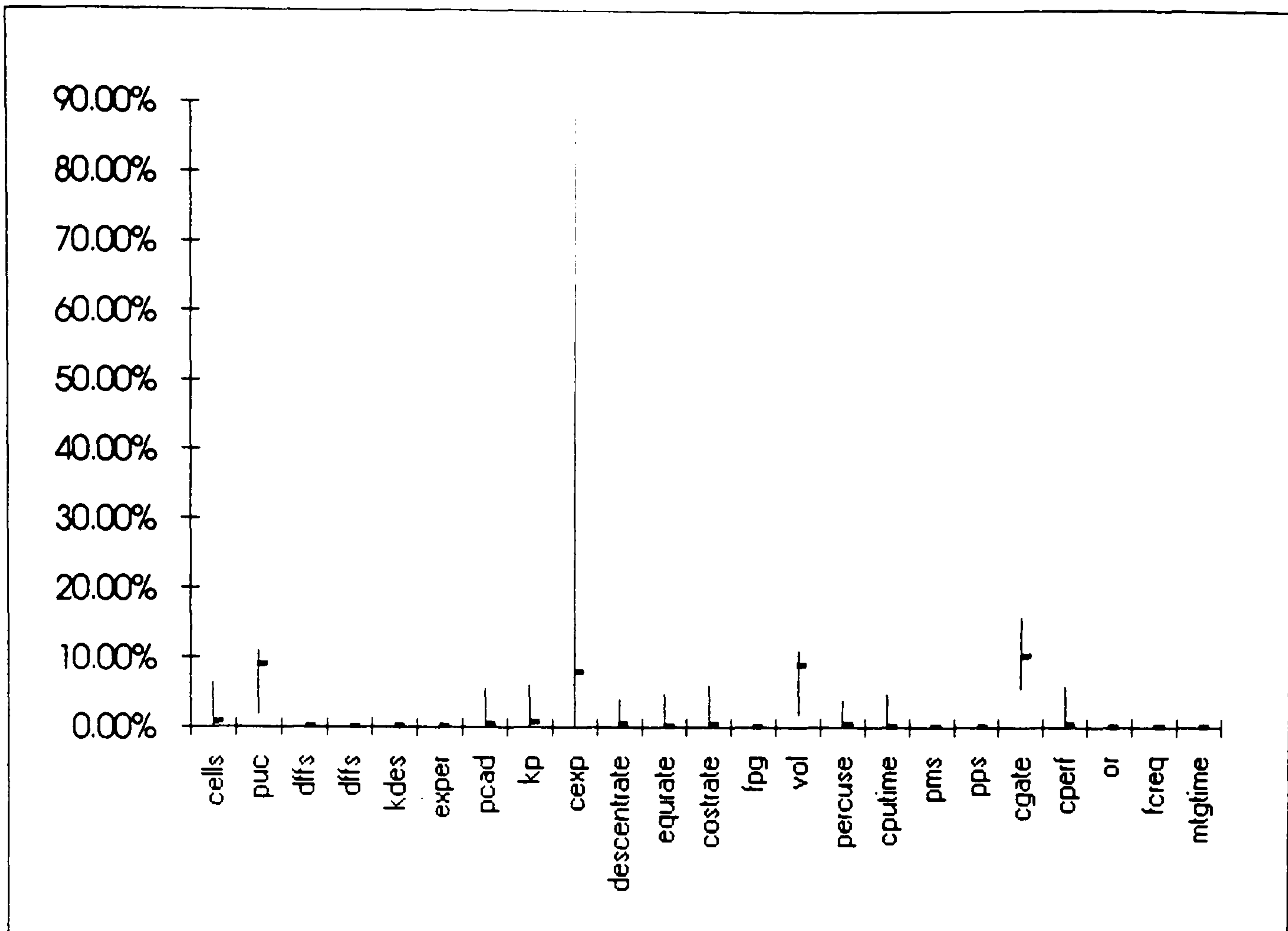


Figure 32: 99% range of sensitivity with a 10% variation, self test

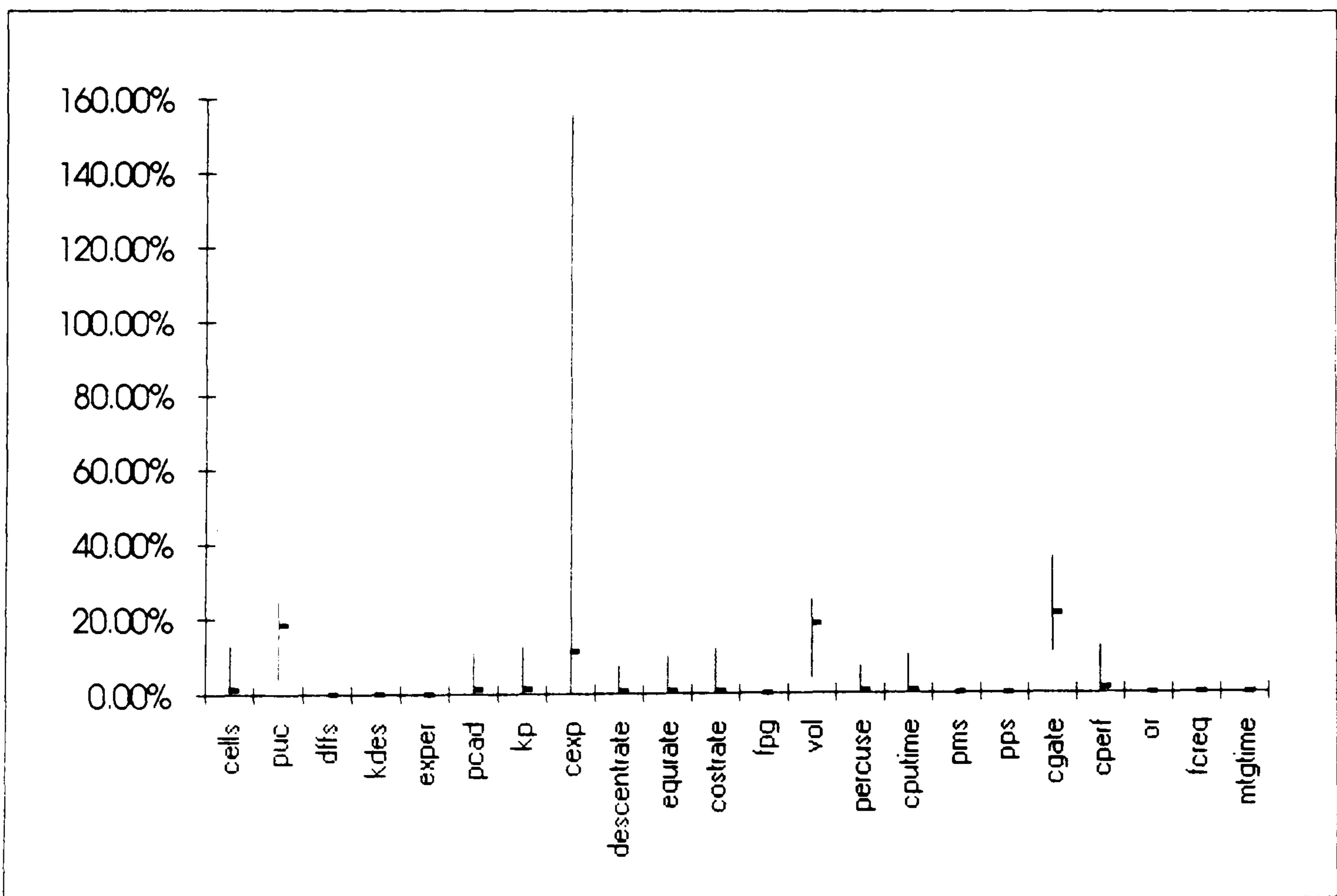


Figure 33: 99% range of sensitivity with a 20% variation, self test

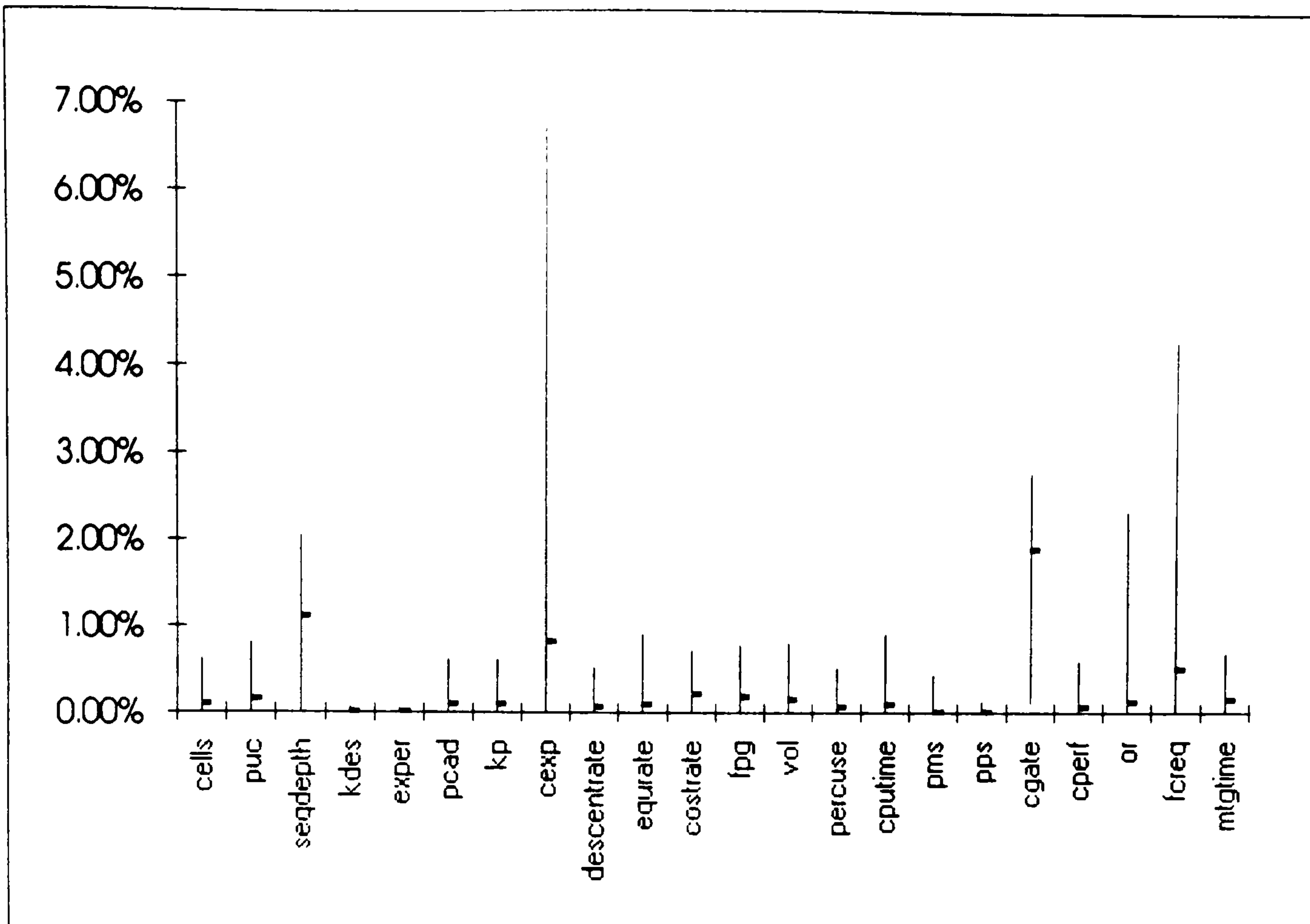


Figure 34: 99% range of sensitivity with a 1% variation, no DFT, low production volume

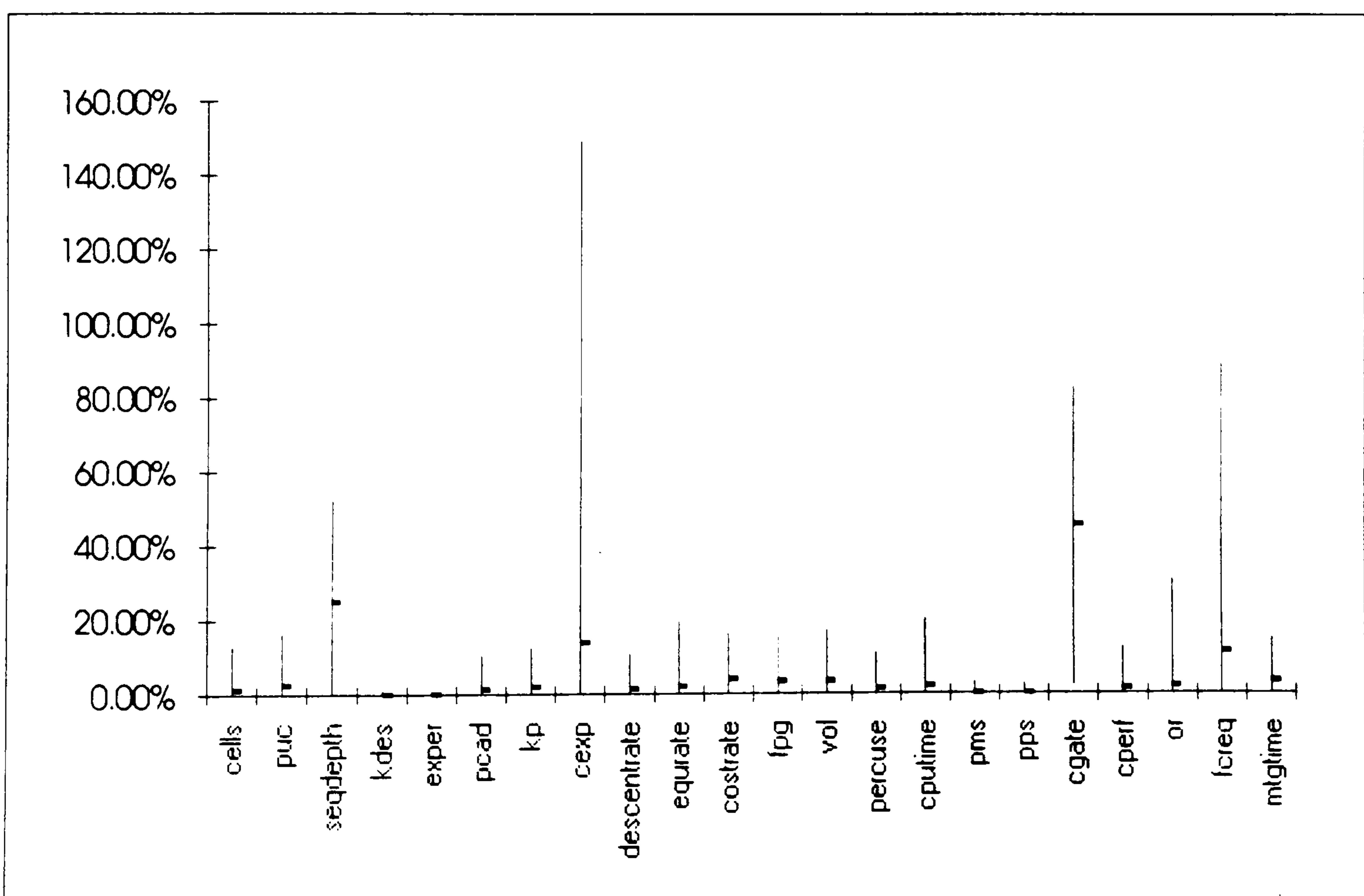


Figure 35: 99% range of sensitivity with a 20% variation, no DFT, low production volume

6.4.2. Estimation of the Maximum Sensitivity

As already mentioned in the previous section, the sensitivity to the total cost for some parameters remains negligible in at least 99% of all cases. This leads to the question,

whether the total cost is insensitive to these parameters at all. To answer this question, the maximum sensitivity for the parameters must be determined. If the maximum sensitivity is negligible, the cost model can be refined by taking out the insensitive parameters. The gains of this refinement have been discussed in the introduction of section 6.2..

In this section, the author will describe the methods he has developed in order to determine the maximum sensitivity and he will present and discuss the results.

6.4.2.1. The Algorithm

The author will now introduce the definition of the maximum sensitivity as it is used in this thesis. Therefore the following definition will be made:

- The *total cost difference for parameter i* is defined to

$$CD_i(\mathbf{x}) = C(\mathbf{x}, x_{i0}) - C(\mathbf{x}, x_{i1}) \text{ with } C(\mathbf{x}, x_{i0}) \geq C(\mathbf{x}, x_{i1}) \quad (24)$$

- The *relative difference for parameter i* is defined to

$$RD_i(\mathbf{x}) = \frac{C(\mathbf{x}, x_{i0}) - C(\mathbf{x}, x_{i1})}{C(\mathbf{x}, x_{i0})} \quad (25)$$

- The *maximum total cost difference for parameter i* is defined to

$$CD_{\max_i}(\mathbf{x}, x_i) = \max(CD_i(\mathbf{x}, x_i)) \forall (x_{i0}, x_{i1} \in D \in R) \quad (26)$$

  where D is the constrained search space of x_i as defined in table 10. This means, that CD_{max} is such that, if all parameter values beside x_i are kept constant, x_{i0} is the maximum and x_{i1} is the minimum of the function $CD_i(\mathbf{x}, x_i)$.

- The *relative maximum difference for parameter i* RD_{max} is defined to the relative difference of the maximum total cost difference.
- The *maximum sensitivity for parameter i* is defined to

$$S_{\max_i}(\mathbf{x}) = \max(RD_{\max_i}(\mathbf{x})) \forall (\mathbf{x} \neq x_i \in D \in R) \quad (27)$$

  This means, that the maximum sensitivity is determined by that combination of input parameters which leads to the relative maximum difference in the total cost

which can be caused by the variation of parameter i.

The author will use this definition of the maximum sensitivity for deciding, whether a parameter can be taken out of the cost model.

In order to determine the maximum sensitivity, we need to determine the relative maximum difference. If the function $CD(x_i)$ is monotone, the maximum and the minimum of CD is given by the upper and the lower limit of the constrained space of the parameter. Because this is not always the case, the author has selected the following approximation method to determine the maximum and minimum of CD:

```
calculate_CDmax()
  xmin is the lower bound of range
  xmax is the upper bound of range
  minind = maxind = 1
  while ((recursion depth less than 5) and (minind and maxind are not 0 or 10))
    Cmin = 0
    Cmax = maximum float number
    minind = 0
    maxind = 10
    for i from 0 to 10
      Cx = C(xmin + i•(xmax-xmin)/10
      if (Cx > Cmax)
        Cmax = Cx
        maxind = i
      else if (Cx < Cmin) then
        Cmin = Cx
      minind = i
      endif
    endfor
    if (minind and maxind are not 0 or 10) then
      xmin = xmin + max (0, min (minind, maxind)-1)•(xmax-xmin)/10
      xmax =  xmin + min (10, max (minind, maxind)+1)•(xmax-xmin)/10
      increment recursion depth
    endif
  endwhile
  RDmax = (Cmax-Cmin)/Cmax
```

This algorithm will give a good approximation of the relative maximum difference, if the sensitivity of the function does not vary too much, i.e. there are no "sharp peaks" in the function. This can assumption can be made for the cost model.

The maximum sensitivity can be determined by a Monte Carlo simulation as follows:

$$S_{max} = \max(\bigcup_{j=1}^{n} CD_{max}(\mathbf{x}_j)) \quad (28)$$

The values of **x** are sampled from uniformly distributed variates as defined in table 8. This method requires a large amount of simulations to achieve sufficient results, because we want to determine a specific element out of an infinite set of parameter combinations. The following example should illustrate this:

> The number of parameters is 23. If we divide each parameter range into four sub ranges, and if we want to combine all sub ranges for all parameters, the number of combinations is 4^{23}. There is no chance to run so many simulations.

Therefore the author has developed a technique to reduce the sample by keeping the accuracy of the result. This technique will be described in the following section.

6.4.2.2. Reduction of the Sample Size

There are several publications which are related to Monte Carlo optimisation ([Ant82], [Kje81], [Rub86]). All of them address the problem of reducing the sample size by using a technique called *importance sampling*. This technique exploits the random samples of the past for generating the random samples in the future. In [Ant82] and [Kje81] the importance sampling techniques closely linked to the problem to solve, which is different from the problem to be solved here. In [Rub86], the techniques presented are more general. The author has therefore implemented a different algorithm. In this algorithm the Monte Carlo simulation is divided into two phases:

1. In the first phase Monte Carlo simulation is performed with the input parameters uniformly distributed. In this phase we make an estimate of the values of the input parameters, which will lead to a high sensitivity. We call these values *maximum sensitivity values*.
2. In the second phase Monte Carlo simulation is performed with a distribution function for the input parameters, which is adopted to the maximum sensitivity values. In this phase the sensitivity values are updated with each simulation. This means, that in this second phase the parameter values are sampled in the region where we expect the actual maximum sensitivity value.

The algorithm is implemented as follows:

```
calculate_max_sens()
  for L times
    perform Monte Carlo simulation with uniformly distributed variates as input parameters and
determine RDmax
  endfor
  select M samples from all previous simulations which  cause the M maximum RDmax
  for N times
    perform Monte Carlo simulation with importance samples and determine RDmax.
    update the M maximum samples with the sample which just was calculated.
```

```
endfor
```

The importance samples are based upon the input parameter values of the M maximum samples, and they are calculated as follows:

```
importance_sampling()
  determine the distribution of the parameter values which are related to the M maximum samples.
  constrain this distribution by the lower and the upper limit of the parameter range
  generate a variate from this distribution.
```

The generation of a variate based on the determined distribution function was implemented by applying a table method which is similar to the Marsaglia table method. Instead of using the table for a normal distribution, we take the table of the values for the M maximum samples.

The object in this method is to concentrate the distribution of the sample points in the parts of the parameter range that are of most importance instead of spreading them out evenly. Nevertheless, the L simulations with a uniform distribution function are needed to make a good estimate about which parts are important. If a certain parameter has influence on the maximum sensitivity, the related importance sampling distribution will converge to a certain value, which we expect to be the value for the maximum sensitivity. The speed of the convergence and therefore the sample size and the risk of running into a local maximum is driven by the values L and M. A high value for M will cause slow convergence but a low risk of running into a local maximum and vice versa. A higher value for M will lower the risk of running into a local maximum but will increase the sample size and therefore computer runtimes. The author has performed some experiments with different values for L and M to find a good trade-off. In the next section, the values for L, M and N will be presented and the maximum sensitivity values for the parameters which are expected to be insensitive will be presented and discussed.

6.4.2.3. Results

The author has found M=10 and L=300 to be a good trade-off between fast convergence and a low risk of running into a local maximum. The total number of simulations L+N

was set to 1000.

The maximum sensitivity was determined for the following parameters:

performance complexity, design centre cost rate, designer's experience, average number of faults per gate, constant factor concerning designer's productivity, constant factor concerning total productivity, manual test generation time per fault, number of flip flops, productivity of the CAD system, percentage of design time an external design centre is used.

The analysis was made for the test strategies *no DFT*, *scan path* and self test. The resulting maximum sensitivity in percentage of the related maximum cost is shown in figure 36. This graph shows, that the maximum sensitivity exceeds 20% for all parameters, which is by far too high to neglect. This means, that for **all** parameters situations may occur, where a variation of the parameter within the range defined in table 8 can lead to more than a 20% variation of the total cost. A simplification of the cost model in general by removing parameters is therefore not possible.

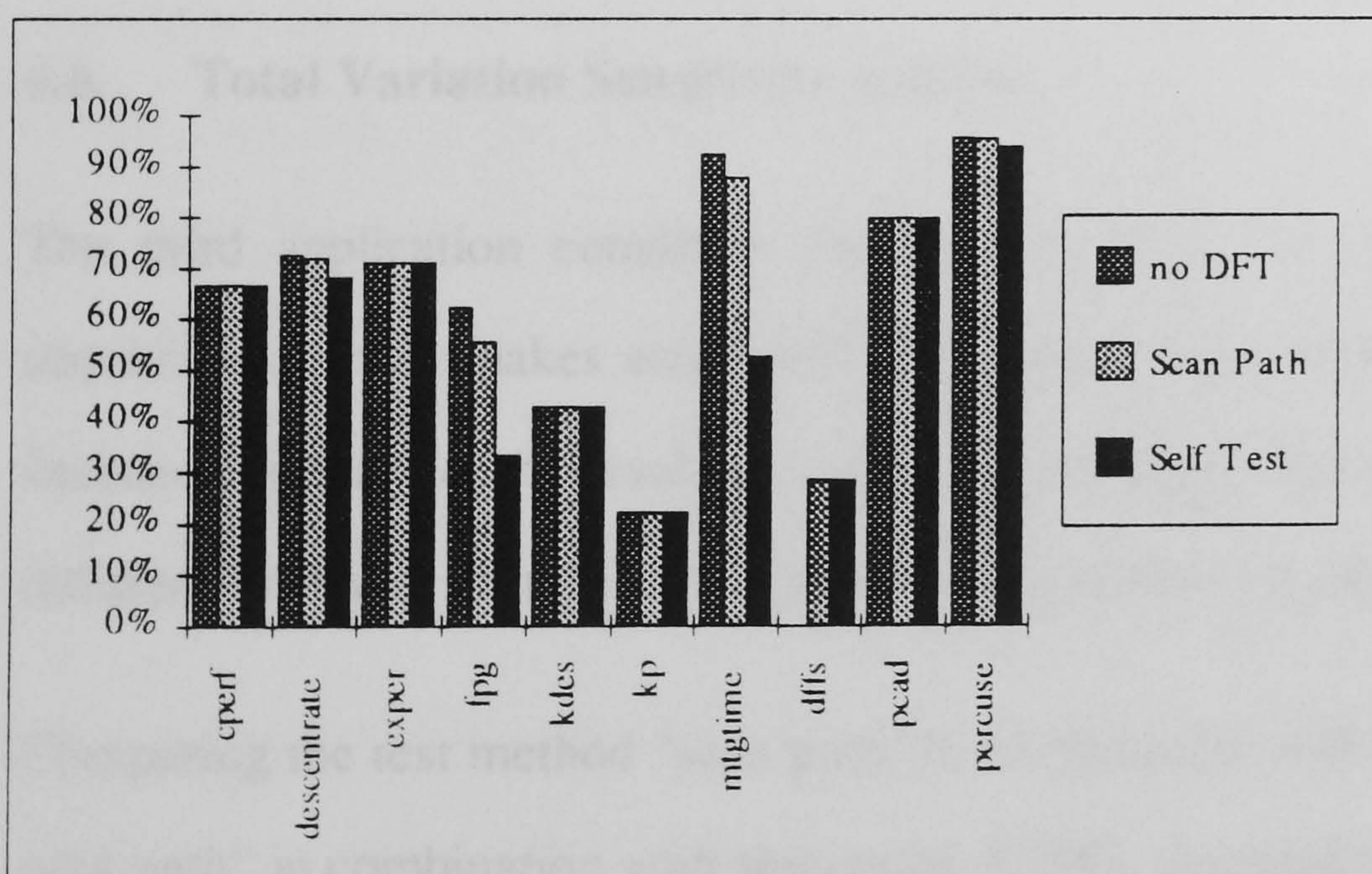


Figure 36: Maximum sensitivity values

6.5. Iterative Sensitivity Analysis

The second application of sensitivity analysis studied in this chapter is identical to type 4 mentioned in section 6.2.. This application leads to a test strategy planning procedure in iterative steps as follows:

1. Make a rough and inexpensive estimate of the input data.
2. Perform test strategy planning.
3. Perform a sensitivity analysis in order to find out, which parameters need to be defined more accurately. These are those parameters, which have a high sensitivity.
4. If accuracy of the result is OK, then the test strategy planning procedure is finished, else make more accurate estimates for the most sensitive parameters and go to step 2.

In the iterative sensitivity analysis, the same methods are used as for the general sensitivity analysis. The main differences are the iterative procedure and the different distribution of the parameter values. For the general sensitivity analysis, we used a uniform distribution for all parameters in the range of all possible values. For the iterative sensitivity analysis, we have inaccurate estimates for the parameters, which can be either a uniform distribution or a normal distribution.

6.6. Total Variation Sensitivity Analysis

The third application considers the fact that there is a certain cost limit for data acquisition, which makes estimated input data remain inaccurate. This means that the decision criteria, i.e. the resulting cost value, is subject to inaccuracy. This inaccuracy is different from test strategy to test strategy. The following example will illustrate this fact.

Comparing the test method "scan path" in combination with combinational ATPG to "no scan path" in combination with sequential ATPG, the related test generation cost as part of the resulting cost are much more uncertain for "no scan path". The reason for that is the inaccurate parameter "sequential complexity", which affects sequential ATPG cost but not combinational ATPG cost. This fact may lead to a situation, where the resulting costs for the sequential ATPG approach are lower as those for the combinational ATPG assuming that the estimated value for the sequential complexity is exact, but that there is a high probability, that the costs will be much higher for the sequential ATPG approach,

considering the high probability that the sequential complexity will be higher than estimated.

In order to take the inaccuracy of the cost estimation into account, the author will replace the total cost value by the distribution function of the total cost value as the optimisation criteria.

6.6.1. The Algorithm

The distribution function F(x) will be used to determine with a given cost probability p_C the upper cost limit CL:

$$CL = F^{-1}(p_c) \quad (29)$$

The cost probability has to be defined by the user of the test strategy planner. This cost probability is defined as the probability, that the total costs will be less than the related cost limit, due to inaccuracy of the input data. The value for the cost probability represents the *risk of the cost estimation* and by that the risk of the decision. In other words, the risk of the decision is the risk, that the chosen test strategy is not the optimum. This type of sensitivity analysis application is an extension to type 1 in section 6.2.. There, the sensitivity analysis is applied to the "solved problem", which is the selected test strategy alternative. Here, the sensitivity analysis is applied to the selection procedure, which is the test strategy planning procedure.

The calculation of the cost limit is implemented by performing Monte Carlo simulation. If the n cost values resulting from n simulations are stored in a sorted list c[n] in rising order, the cost limit can be estimated by a linear approximation as follows:

$$CL = c[\text{int}(n \cdot p_c)] + (c[\text{int}(n \cdot p_c)+1] - c[\text{int}(n \cdot p_c)]) \cdot (n \cdot p_c - \text{int}(n \cdot p_c)) \quad \text{if } p_c < 1$$
$$CL = c[\text{int}(n \cdot p_c)] \quad \text{if } p_c = 1$$

where $\text{int}(np_C)$ is the integer of the product np_C. It represents the index in the list of cost values, which points to the cost value which is closest to the estimated

cost limit.

6.6.2. Determination of the Number of Simulations Needed

The accuracy of the determination of the cost limit as described in the previous section depends on the number of simulations for two reasons:

- The Monte Carlo simulation is an unbiased estimator for the distribution of the total costs. This estimator is subject to inaccuracy as long as the number of simulations is finite.
- The linear approximation is subject to inaccuracy which depends on the curvature of the distribution function in the region where the linear approximation is made, and it depends on the number of simulations.

The author will therefore study the impact of the number of simulations on the accuracy of the cost limit estimation in order to determine an appropriate number for the Monte Carlo simulations.

The techniques used in 6.4.1 to determine the estimation error cannot be used here, because they are related to the estimation of the mean value and the variance, where in this analysis we estimate a certain value of the distribution function. The author has decided study the convergence of the Monte Carlo simulation for analysing the estimation error and the number of simulations needed to keep the error under a certain limit. Due to being an unbiased estimator, we can assume, that the Monte Carlo estimation will converge to exact value by increasing the number of simulations. If the difference between succeeding estimations during the Monte Carlo simulation become very small, we take the related value as the exact value. Based on this exact value CL the relative estimation error for an estimate CL_{est} is given by

$$\vartheta_{est} = \left| \frac{CL - CL_{est}}{CL} \right| \quad (30)$$

The author has performed the convergence study for the test strategies *no DFT*, *scan*

path and circular self test path, and for the following cost probabilities:

99%, 90%, 75%, 60% and 50%.

The relative estimation errors are shown in figures 37 through 39. The author has selected the cost limit for 10,000 simulations as CL. The graphs show, that for cost probabilities of 90% and 99% the estimation error converges slower to zero than for cost probabilities of 50% or 60%. This is because of the larger gradient of the inverse distribution function $F^{-1}(p_C)$ for the high end and low end regions of the function, for which an argument difference affects the related cost limit more than for lower gradients. The relative error remains for all curves under 5% for simulation number greater than 200. The author proposes therefore a simulation number of **200** to perform Monte Carlo simulations for total variation sensitivity analysis.

Some representative results will be presented and discussed in chapter 8.

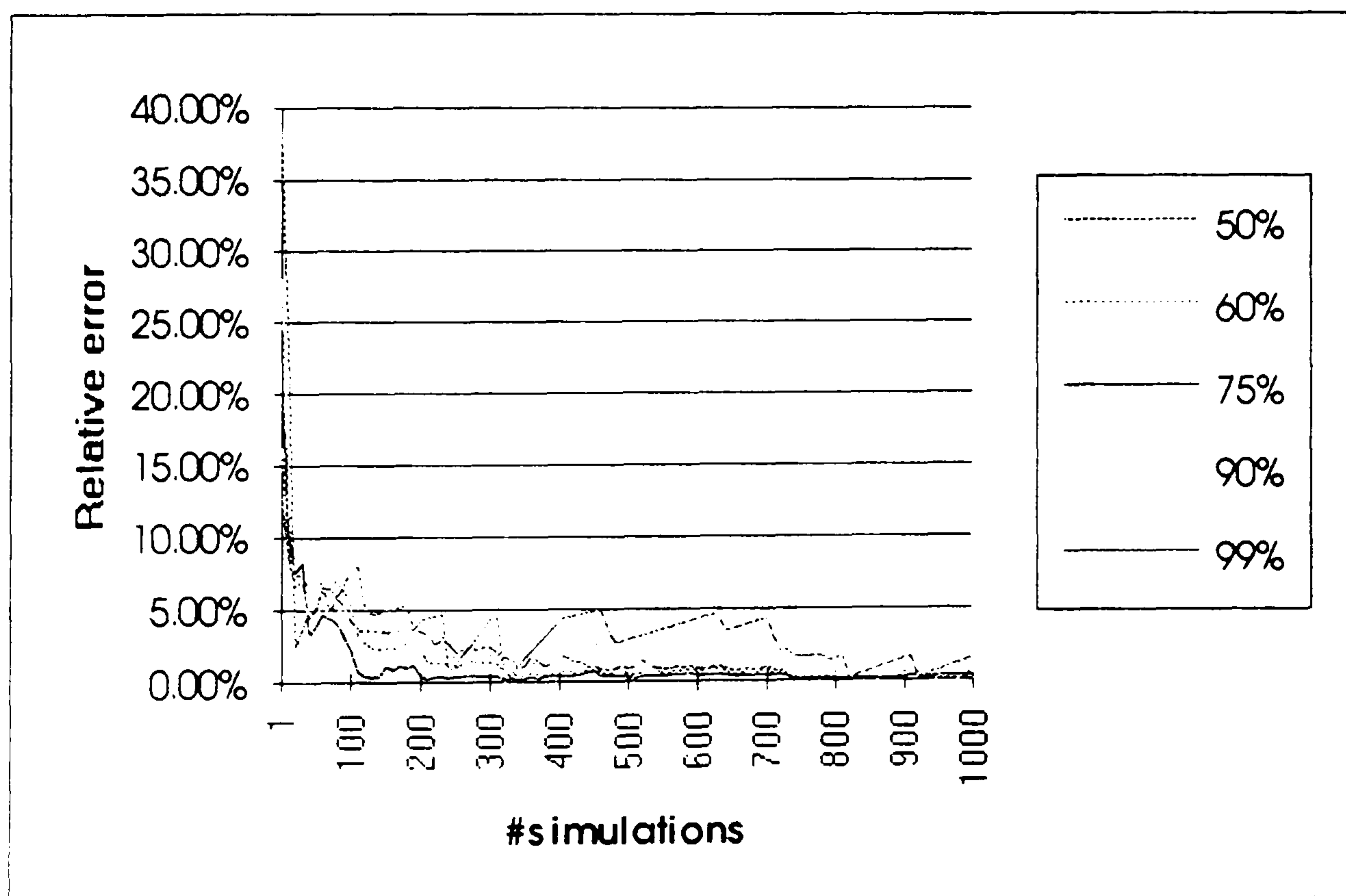


Figure 37: Convergence of Monte Carlo simulation for no DFT

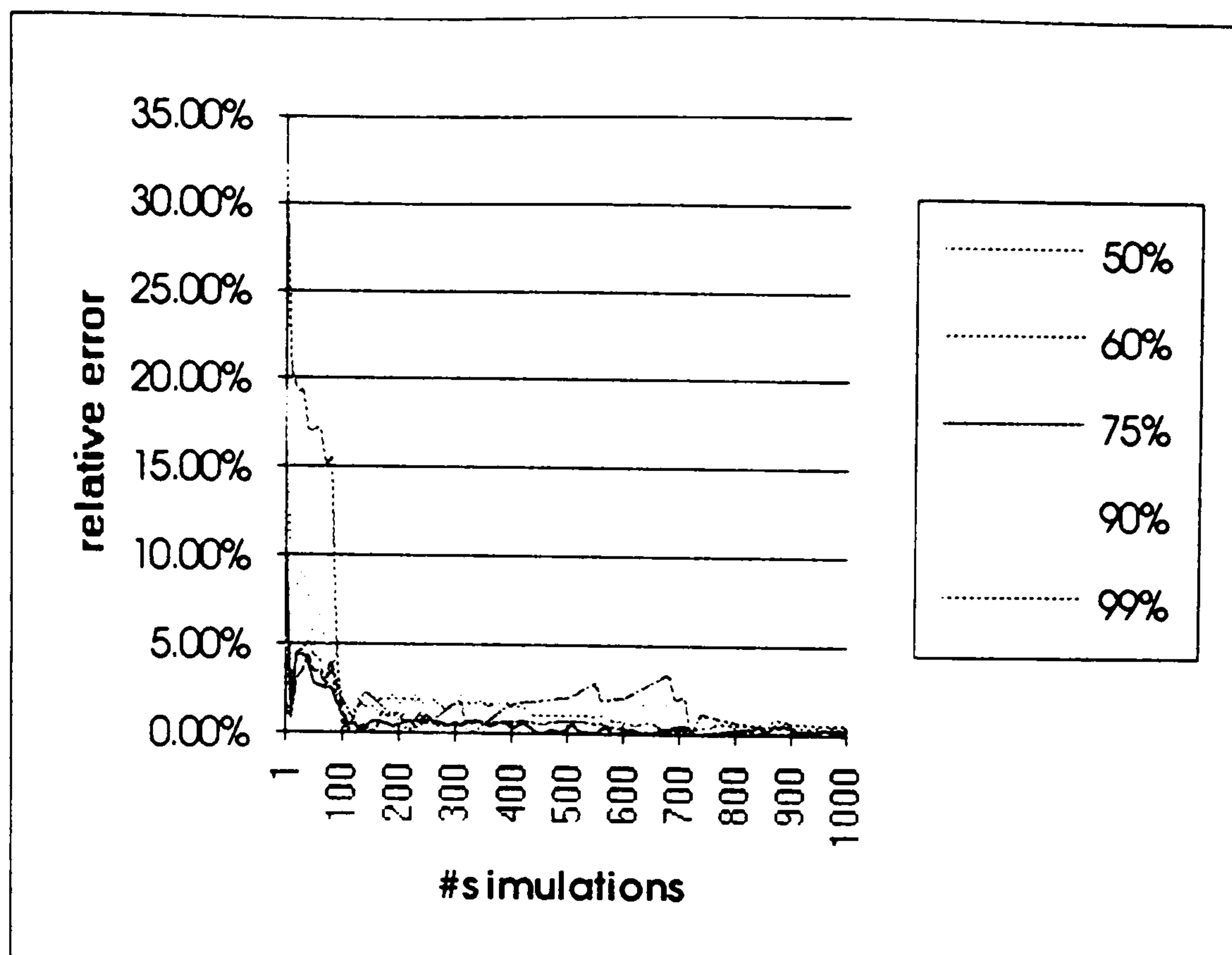


Figure 38: Convergence of Monte Carlo simulation for scan path

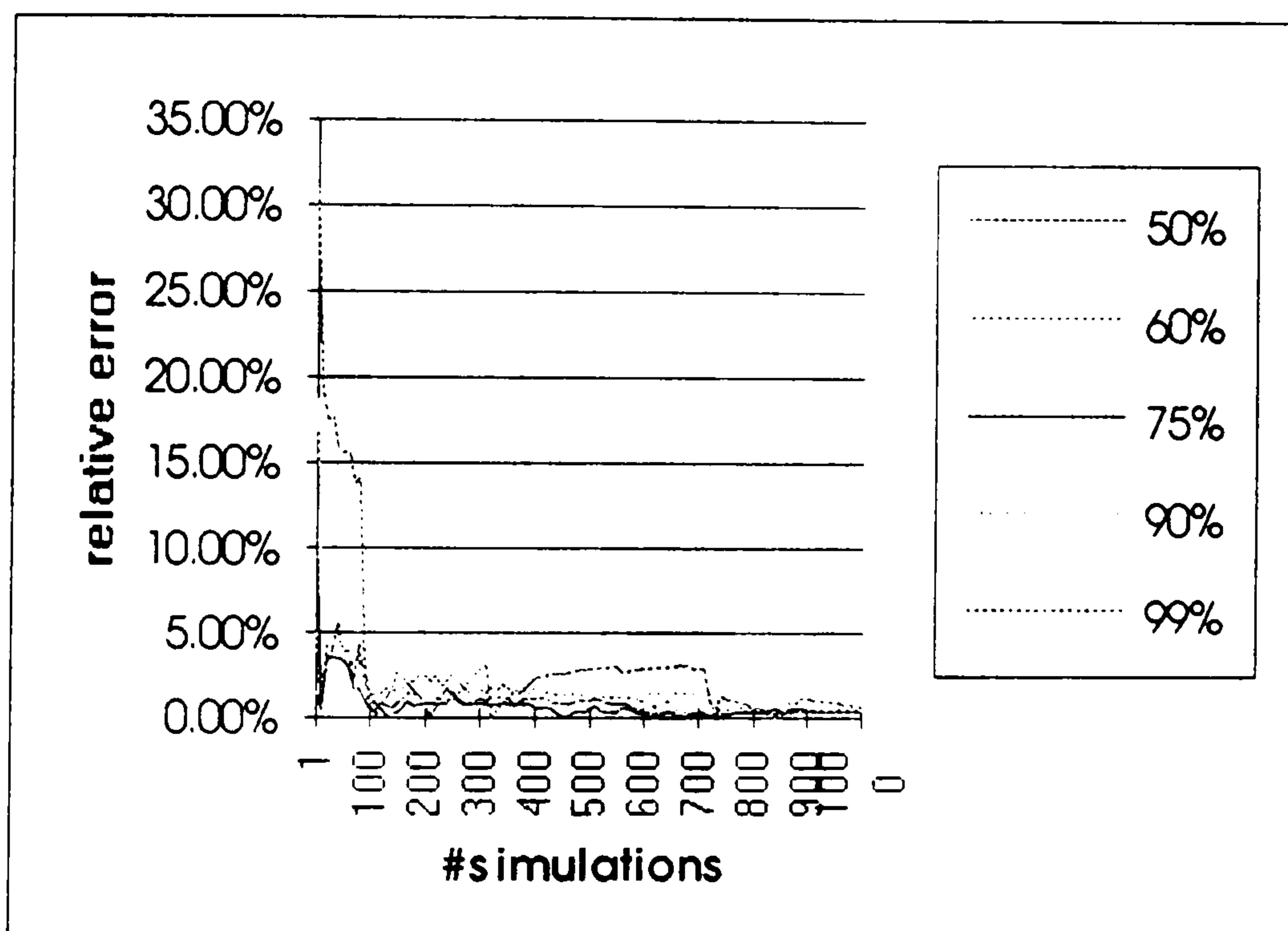


Figure 39: Convergence of Monte Carlo simulation for self test

6.7. Summary

In this chapter the author gave a literature survey on sensitivity analysis and presented the techniques he has developed to perform sensitivity analysis for three different applications:

- the *general sensitivity analysis* was performed to classify the cost model parameters concerning their sensitivity impact to the total costs.
- the *iterative sensitivity analysis* is used to make iterative cost parameter

estimations, which can reduce the cost for gathering data for the parameters.

- The *total variation sensitivity analysis* allows to perform test strategy planning with uncertain or inaccurate cost model parameters, which are described by their distribution. The optimisation criterion is based on the distribution of the total cost instead of an average value. The user of the test strategy planning system can select to take a basis for the system's decision for lower cost with a high risk to be exceeded or for higher cost with a low risk to be exceeded.

All applications are based on Monte Carlo simulations. This technique was introduced and adopted to the problems described in this chapter.

The general sensitivity analysis showed, that for each parameter configurations may occur, which lead to a maximum sensitivity of more than 20% of the total cost. But in most cases, the total cost is sensitive to only a few parameters of the cost model. These are the number of gates, the sequential depth, the production unit cost per gate, the production volume and the complexity exponent. Based on this outcome, the author proposes to perform economics based test strategy planning as follows:

- Provide values for the "important parameters" listed above.
- Perform iterative sensitivity analysis until the accuracy requirements are fulfilled.

This method can drastically reduce the effort spended in data gathering, because in most cases the accuracy requirements can be fulfilled by providing values only for a few parameters.

The cost model which was studied in this chapter calculates the costs related to the design and production of an ASIC. The methods were implemented for the test strategy planner for ASICs. But many design-for-testability methods produce cost savings in cost areas, which are not covered by the ASIC cost model, and which are related to the design and production of boards and systems. The next chapter will describe a test strategy planner which allows to plan and analyse the economics of test strategies for boards and systems.

Chapter 7

ECOvbs: A Test Strategy Planner for VLSI based Systems

The EVEREST test strategy planner for VLSI based systems, here called *ECOvbs*, was developed under the task "Test Economics Modelling" of the EEC funded ESPRIT project "EVEREST" in collaboration of Brunel University and Siemens-Nixdorf-Informationssysteme AG (SNI). The aim in this task was to develop and to evaluate test economics models and to use them in a test strategy planning system, whose development also was part of the project. Both the test economics modelling work and the development of the test strategy planner was performed for the application to VLSI test strategies and to VLSI based system test strategies. In chapter 5 the author described the test strategy planner ECOTEST, which targets test strategy planning for VLSIs. This chapter will present the test strategy planning system ECOvbs for VLSI based systems.

The tools developed in the EVEREST project are intended to be used in industrial environments by the partners of the project. This aspect, and the fact, that such a system had never been developed before (the reasons are discussed in the next section), were the basis of the following generic requirements for the system:

1. The system should take into account industrial practices and needs.
2. The system should therefore gain from the experience made in manual test strategy planning by quality assurance people.
3. The system should be integrated into an industrial environment.
4. Due to being a novel approach and a research work, the system should be a good compromise between the state-of-the-art in research and the applicability of the system in industry.

ECOvbs was designed and developed by SNI and Brunel University in an industrial/academic collaboration. Due to being employed by SNI, the author had good access to industrial data, experience and user requirements of such a system. By working

in a research environment in collaboration with an academic organisation, the author was able to design and develop a system, which fulfils the needs of industrial users, but includes features and techniques, which are completely novel and ahead of state-of-the-art. The specification of the work was based upon extensive talks to industrial users, a detailed research in literature, and many fruitful discussions between the partners in the task.

In this chapter, the author will first discuss the philosophy of the test strategy planner and the arising needs of such a system in industry. In the subsequent sections, an overview of the architecture of ECOvbs will be given, and its components will be described in detail.

7.1. Philosophy of ECOvbs

In the early 80's the manufacturing efficiency became the cornerstone in industry with the main objective to *build the highest quality product at the lowest possible cost*(see [Pyn86]). Pynn states, that "today's successful business requires an objective which takes both product quality and manufacture efficiency into account" [Pyn86], and based on this statement, he defines the following manufacturing objective:

> *To develop a production strategy which will achieve a product with superior quality manufactured at the lowest possible cost.*

One crucial part of the production strategy, especially in the field of VLSI based systems, is the test strategy. The test strategy is the essential factor to quality. And it affects nearly all cost areas of an electronic product.

Pynn [Pyn86] defines a test strategy as follows:

> *A successful test strategy is the optimum arrangement of various testers in the circuit board manufacturing process that will result in products of maximal quality at minimum cost.*

Based on this definition, the technical factors affecting the test strategy are the fault

spectrum, the process yield, the production rate or volume and the product mix, i.e. aggregation of active board types in a production line. The economical factors are the tester acquisition cost, the tester adaptation cost (fixture cost and test generation cost) and tester operation cost.

In this approach it is assumed, that for a given test strategy the technical factors are fixed, and a test strategy is purely built upon a mixture of test applications. A similar definition of test strategies is made in [Dav82]. This definition of a test strategy makes the test strategy planning task relatively simple for the following reasons:

- The only degree of freedom in the strategy planning is what test equipment should be used in which phase of the production cycle.
- Therefore the cost areas affected by a test strategy are only the cost related to the test equipment, as defined in the previous paragraph.
- Decisions about the test strategy can be made rather late. The first phase in the product life, which is affected by the test strategy, is the test generation phase, which in the past was done after the design cycle.
- There was no large multitude of test equipment. Automatic test equipment arised at the end of the 60's, and only a few companies shared the market on the test equipment.
- The rate of change in technology and therefore test technology was much lower than today.

These factors were the main reasons, why there was no need for a test strategy planning tool. The task of test strategy planning was confined to the selection of a new test equipment or the proof, whether the existing tester scene is appropriate for the new product.

Today the situation has completely changed:

- People know that quality and production costs are mainly determined during the specification and design phase of the product. Reinertsen claims, that "decisions

on technology, architecture and physical structure, which typically occur during the first few months of the design process, determine 90% of the product cost and the majority of its capabilities" [Rei83]. A similar statement is made by Szygenda: "...although the system design phase of product development represents only about 15% of the product's cost, it has a 70% impact on that product's operation and support cost" [Szy92].

- Today, design methods are available to control the production and the quality. Design for testability (DFT), design for manufacture (DFM) or concurrent engineering are the keywords in this field.
- Because of rapid technology changes, the product life time becomes shorter, and this fact and an increasing competition in the market leads to enormous price pressures in the electronics market. This arises the need for low production cost right from the beginning of the production.

For these reasons, a today's test strategy includes, beside the usage of test equipment, the design-for-test and design-for-manufacture methods, and therefore the author's definition of a test strategy (section 3.2) is different from that used by Pynn.

A test procedure consists of the provision of the test equipment and the environment to utilise it, the adaptation of the test equipment to the device under test, i.e. the hardware adaptation and the test program generation, and the test application. The design methods are those which facilitate the realisation of the test procedure (DFT) and those which support the avoidance of errors, or, more specific, which increase the yield in the production (DFM).

The optimum test strategy can be defined as follows:

An optimum test strategy is a test strategy, which meets all given product constraints and the system's final quality requirements by minimum total cost.

Based on this definition and the changing technology, design, production and market around electronic systems, the determination of the optimum test strategy and therefore the test strategy planning task is much more complicated than an approach based on a test strategy as defined by Pynn [Pyn86]:

- The test is now completely integrated into the design process. Therefore the decision on the test strategy must be made very early in the specification phase of the product. This makes the derivation of the test strategy factors, such as the prediction of yield, fault coverage, production cost or test adaptation cost much more complicated and uncertain. Especially in an environment with quickly changing technologies, it becomes very difficult to make the estimation of the factors, because we cannot gain from data of previous products.
- The estimation of the total cost is much more complicated as the estimation of the economic factors as defined by Pynn, which are purely related to the test procedure. Today's test strategies affect cost areas such as design cost for different design alternatives, component procurement, cost of the production process, test generation cost or test-, diagnosis- and repair cost in the production or in the field. All these costs have to be determined in a product phase, where factors such as the production yield, production volume, design complexity, fault coverage of a test or the fault spectrum cannot be measured and therefore are based on estimates.
- The multitude of test strategies is increased by another dimension, which is the design option.
- Factors such as the fault coverage per test and the production yield are continuous parameters of the test strategy, which need to be optimised per test strategy to analyse. This optimisation process (which is the optimum mix of yield and fault coverage in order to achieve a given product quality?) can be on its own an extremely complicated task.
- The planning of test strategies will need organisational changes and changes in responsibilities in most of the companies, because the implications of a test

strategy are no more limited to the production people (see also [Dav92]). The implementation of a test strategy as defined here involves most parts of the company: engineering, manufacture, finance, service, purchase and even marketing and sales (e.g. to determine what quality level the market requires).

All these aspects necessitate a structured support for the test strategy planners. This support can be given by a software tool, which supports the user in

- providing all factors which are needed to determine the optimum test strategy,
- evaluating the test strategies concerning its economics and compliance to quality and design constraints,
- optimising the parameters of a test strategy, and
- selecting the optimum test strategy.

The development of such a tool was the objective of the test economics task in the EVEREST project. The author has designed and developed this tool in collaboration with Brunel University. This tool, which is called ECOvbs, will be described in this chapter.

A test strategy for VLSI based systems includes test procedures at all levels of integration. Therefore the test of the components and the related test strategy is part of the test strategy of the entire system. Also, for several DFT methods, which are implemented at component level, or which are used for component level test, gains are achieved for board level or system level test. In particular, boundary scan is implemented in the component for supporting the board level test. Other examples are the scan path technique, which was originally used and demanded by the system test people for diagnostic purposes ([Sed92]), or built-in self test techniques, which are used for board test and diagnostics. These aspects make the test strategy planning task hierarchical, and the test strategy planning system for VLSIs, ECOTEST, can be integrated into ECOvbs for the test strategy planning task at component level for the VLSIs.

Due to good progress which the author made with object-oriented programming in C++,

ECOvbs was implemented in C++. This allowed the reuse of software components, which the author had developed for ECOTEST, such as the cost model calculator or the command handler. It is very easy to integrate C-components in C++, which was important, because the experience in C is much more common than in C++, and which allows to use code generators such as LEX or YACC. Another advantage of C++ as an object oriented language is, that the performance of the executable program is much better than implementations in other object-oriented languages, such as Smalltalk.

As this was a collaborative project, it is important to indicate the areas of the author's contribution. The cost model evaluator and the interface to the cost model text files were taken from the author's implementation of the C++ version of ECOTEST. The author has added some calculation functions. The interface to the test method descriptions was specified and implemented by the author as well as the test strategy planning functions, the test strategy evaluation functions, and the user interface. The design data interface and the test method evaluation functions were implemented by Brunel University, but the author was involved in the specification of these parts, especially because of his industrial experience about design data interfaces. The overall conception and specification of ECOvbs was made by the author.

7.2. System Overview

ECOvbs will provide the user with an estimate of the costs involved in using a particular test strategy on a given electronic design. The user can look at cost-quality trade-offs and can acquire an understanding of the fault spectrum of the electronic system at each stage of testing.

In order to provide all the support needed to make test strategy decisions. ECOvbs comprises of the following features:

- A series of cost models are used which describe the cost structure for all cost areas which are affected by test strategies.
- Cost parameters are linked across the cost models.

- The cost calculations are fully parametrised instead of using rules of thumb.
- ECOvbs utilises a user-supplied design description of a board.
- The test method descriptions and the test equipment descriptions are provided as text files so that they are available for general usage.
- The test strategies can be set up by the user consisting of one or more test stages, which can test different parts of the electronic system, or which are specialised to detect different fault types.
- Test clusters can be defined in order to apply specific test stages only to parts of the electronic system.
- The defined test strategies can be stored for later reference.
- ECOvbs automatically generates the test strategy specific cost models and creates the linkage of the cost models.
- An integrated verification function checks the correctness and completeness of test method descriptions, the applicability of the test methods to the design, and the consistency of the test strategy.
- Test strategies can be varied by deactivating particular parts of the test strategy.
- Based upon the defined test strategies, the ECOvbs system allows to evaluate test strategies in parallel in order to make direct comparisons of the cost components.
- There is considerable flexibility in looking at the costs at each test stage.
- ECOvbs calculates the fault spectrum after each test stage.
- ECOvbs provides a function, which graphically shows the fault spectrum and the total costs per test stage.
- The results can be stored in files for printing and for later reference.
- The user interface provides general commands to execute macros or to administrate the data in addition to the commands to execute the functions which are listed above.
- Extensive help support is provided in order to facilitate the usage of the system.

Figure 40 shows the outline of ECOvbs.

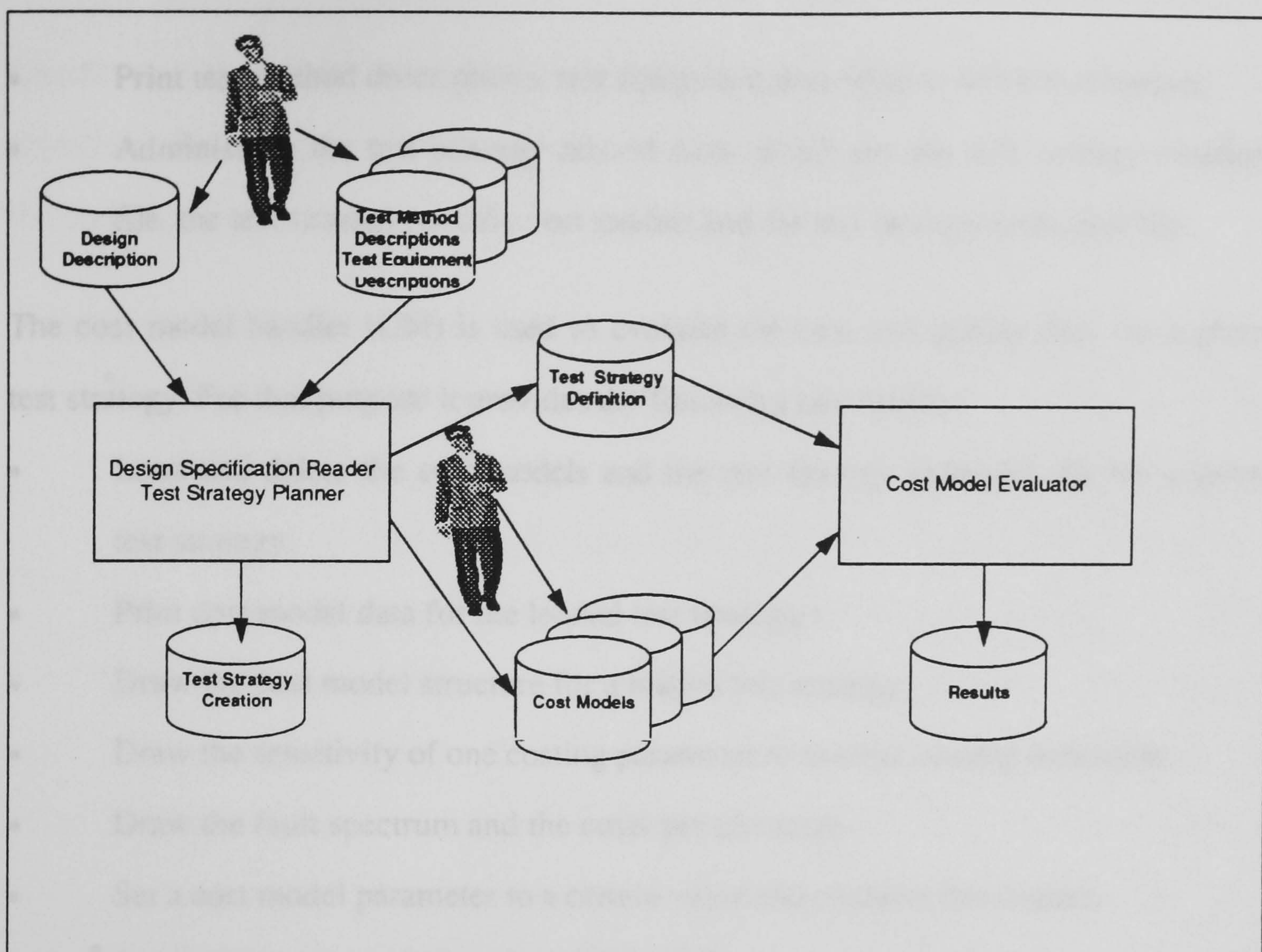


Figure 40: Architecture of ECOvbs

The design specification reader (DSR) handles the design data. It allows to read and to show the design specification. The read function checks the design description about correctness.

The test strategy planner (TSP) is used for managing the test strategies. Based on the design description and the test method / test equipment descriptions, it provides the user with the following functions:

- Read the test method and test equipment descriptions.
- Read and write the test strategy creation file.
- Create test strategies.
- Generate the test strategy specific and design specific cost models and the test strategy definition file.
- Verify the test methods and test strategies.
- Activate and deactivate test stages of a given test strategy.

- Print test method descriptions, test equipment descriptions and test strategies.
- Administrate the test strategy related data, which are the test strategy creation file, the test strategy specific cost models and the test strategy definition file.

The cost model handler (CM) is used to evaluate the cost and quality data for a given test strategy. For that purpose it provides the following commands:

- Load and delete the cost models and the test strategy definition file for a given test strategy.
- Print cost model data for the loaded test strategies.
- Draw the cost model structure for a loaded test strategy.
- Draw the sensitivity of one costing parameter to another costing parameter.
- Draw the fault spectrum and the costs per test stage.
- Set a cost model parameter to a certain value and evaluate this impact.
- Reset all cost parameters to its initial values.

When quitting ECOvbs, all the information about the test strategy (design requirements, test stages, main costing data) is stored in a result file.

7.3. The Design Description

The design description provides all the data which are needed about the board design in order to create test strategies for the board, to verify their applicability, and to make economic evaluations of the test strategy. The design description is provided in ASCII files with a special syntax in a hierarchy, which follows the hierarchy of the board.

ECOvbs should be used in an early stage of the design, because it may impact major design decisions. In this early phase of the design, a netlist of the board is not available. The design is mostly described by a specification paper. Therefore no standard format exists, which contains the data needed in ECOvbs in order to describe a design. For that reason, Brunel University has specified a special format which provides all information, which is needed about a design for ECOvbs. The author has contributed to this

specification by defining the contents and hierarchy concept of the design description. Especially the classification of the components and the DFT methods was performed by the author.

A design description consists of the following data files:

- The board description contains a list of components, production data such as the production volume, number of solder joins, defect rates for the production process, test complexity data, repair cost data, design effort data, test cluster data, production prepare cost and a list DFT methods it supports. A board description can be defined in several DFT alternatives. This means, that the data of the board description differ, if different DFT methods are implemented. For example, a board including boundary scan may contain one additional component compared to no boundary, which is the boundary scan controller. So, the list of components and therefore the board description is different from the no boundary scan alternative.
- The component description is provided per component type. It contains the definition of the component type, the number of components which are used for this board, the mounting type and a list of DFT alternatives. Each DFT alternative is a different version of the component, which provides different DFT features. The DFT alternatives differ in the DFT types they support, the component cost, defect rate and complexity. One DFT alternative can support several DFT types. The complexity information depends on the type of the component and it is different for functional components, edge connectors and bare boards.

There are no netlist data provided, because they are not available at this stage of the design. The DFT alternatives per component are different options of the same component. These options are available for standard components through different versions offered by the supplier, or by implementing an ASIC or a VLSI component with different DFT techniques.

For each DFT alternative of a component, the following data must be provided: price of the component, defect rate in DPM (defect per million).

The following data must be provided for functional components, but the meaning depends on the component type: complexity in gates or bits, number of pins, design effort if the component is designed in-house.

For edge connectors the following data is provided in addition to the price and the defect rate: number of pins.

For the bare board the following data are provided in addition to the price and the defect rate: number of test pads, number of test nodes, number of layers, number of sides, minimum wire separation.

The component types and the DFT types used to classify the components and DFT alternatives are also used in the test method descriptions in order to define the DFT requirements. The author has defined the following component classes:

> analog IC, passive IC, resistor, capacitor, PLA, RAM, ROM, micro processor, ASIC, bare board, edge connector.

To specify the supported DFT type of a component or the whole board, the author has defined the following DFT classes:

> no DFT, scan path, self test, boundary scan, board self test, in circuit test.

The test cluster data allow to define test clusters which are a subset of components of the board. A test cluster then enables to define a test stage as part of a test strategy, which applies the related test method only to the components of the test cluster.

The design description reader parses the files of the design description and creates and fills a data structure, which contains all the design data. Based on these data, test strategies can be defined, if the test method descriptions and test equipment descriptions are available. These descriptions are presented in the next section, and the creation of

test strategies is described in following section.

7.4. The Test Method Descriptions and Test Equipment Descriptions

The test method descriptions and the test equipment descriptions provide all the information about a test method or test equipment, which is needed to verify its applicability to the design, to generate the test strategy data, and to generate the cost models. In addition, a test method description defines, which test equipment are applicable, and vice versa. A test stage in ECOvbs is defined by a combination of a test method and a test equipment.

The information needed to verify the applicability of a test method consists of construction requirements and Design-for-Testability requirements. The verification information of a test equipment consists of construction requirements, which are the same as for the test method. The construction requirements define a minimum wire separation of the bare board, a maximum number of network nodes which can be accessed, the need of test pads, the need for a single side mounted board and the maximum number of edge pins. The DFT requirements define which DFT types for a component type are required. The various types have been defined with the design description.

The information, which is needed to generate the test strategy and the cost models, relates to the DFT requirements. These define, which DFT alternative of the components can be selected. The selected DFT alternative affects many cost parameters, such as the production price of the board, or the defect spectrum.

The test method descriptions and the test equipment descriptions are separated, because a test method may be used in combination with different test equipment. One example is the test method *boundary scan*, which allows to use an IBM-PC like test system as well as an In-circuit tester. For this reason, the data about the test equipment and the test method are separated, and the user selects the appropriate test equipment for a test

method in order to define a test stage in a test strategy. For each combination of test method and test equipment, which should be taken into account, a cost model must be provided which defines all costs which are related to the test generation, test application, diagnosis and repair. A description of this cost model is given in the cost model section of this chapter.

The test method and test equipment descriptions are provided as ASCII files. The reading and interpretation of the descriptions was implemented by using the UNIX tools LEX and YACC. The rest of this section will describe the syntax and the meaning of the test method and test equipment descriptions.

7.4.1. Syntax of Test Method Descriptions

The syntax of the test equipment descriptions is as follows:

```
tmd_def = namedef test_equ_list dft_req_list test_obj_list constr_req t_nl ;
t_nl = {"\n"}
namedef = "NAME" longname t_nl
test_equ_list = test_equ_def {test_equ_def }
test_equ_def = "TEST_EQU" name t_nl
dft_req_list = dft_req_def {dft_req_def}
dft_req_def = "DFT_REQ" comp_class dft_class t_nl
comp_class = name
dft_class = name
stype_list = stype_def {stype_def}
stype_def = "STYPE" name t_nl
test_obj_list = test_obj_def {test_obj_def}
test_obj_def = "TEST_OBJECT" comp_class t_nl
constr_req = "CONSTR_REQ" t_nl
             "WIRE_SEP" ddouble t_nl
             "N_NODES" integer t_nl
             "N_PINS" integer t_nl
             "S_SIDE" yesno t_nl
             "TESTPADS" yesno t_nl
yesno = "Y" | "y" | "n" | "N"
longname = name {name}
name = (A-Z | "S" | "@") {(A-Z | 0-9 | "_")}
ddouble = ({0-9} "." 0-9 {0-9}) | integer
integer = 0-9 {0-9}
```

with the following meanings:

namedef: Definition of user name: match name is the name of the file.

test_equ_list: List of the test equipment, which can be used with that test method.

test_equ_def: Related test equipment, specified by its match name.

dft_req_list: List of the DFT requirements.

dft_req_def: Definition of DFT requirements by defining a DFT class per component type.

dft_class: Name of the DFT type. The available types are defined in design description section.

comp_class: Name of the component type for which the related DFT class is required. The available component types are defined in the design description section.

stype_list: list of production stages, in which the test may be applied.

stype_def: Definition of a production stage for test application.

test_obj_list: List of component types, which can be tested by this test method.

test_obj_def: Definition of a component type, which can be tested.

constr_req: Requirements on the construction of the board; these are minimum wire separation, maximum number of testable nodes, maximum number of pins, single side mounting required and test pads required.

7.4.2. Syntax of the Test Equipment Description

The syntax of the test equipment descriptions is as follows:

```
teq_def = namedef tm_list availability constr_req t_nl;
t_nl = {"\n"}
namedef = "NAME" longname t_nl
tm_list = tm_def {tm_def}
tm_def = "TESTMETHOD" name t_nl
availability = "AVAILABILITY" ddouble t_nl
constr_req = "CONSTR_REQ" t_nl
      "WIRE_SEP" ddouble t_nl
      "N_NODES" integer t_nl
    "N_PINS" integer t_nl
    "S_SIDE" yesno t_nl
      "TESTPADS" yesno t_nl
yesno = "Y" | "y" | "n" | "N"
longname = name {name}
name = (A-Z | "S" | "@") {(A-Z | 0-9 | "_")}
ddouble = ({0-9} "." 0-9 {0-9}) | integer
integer = 0-9 {0-9}
```

with the following meanings:

namedef: Definition of user name; match name is the name of the file.

tm_list: List of the test methods, which can be used with that test equipment.

tm_def: Related test method, specified by its match name.

availability: Number of available test equipment.

constr_req: Requirements on the construction of the board; these are minimum wire separation, maximum number of testable nodes, maximum number of pins, single side mounting required, test pads required.

7.5. The Cost Models and the Cost Evaluator

The cost model structure in ECOvbs is hierarchical. This means, that the cost model of a test strategy consists of a series of cost models describing the design parameters of the board as well as the non design dependent aspects such as the labour rates, and one cost model per test stage, which describes the test dependent parameters such as test generation cost, the test/repair loops or the fault coverage. The cost model structure can be compared to a hierarchical netlist structure. In the netlist, the components are connected to form the circuit. In the cost model for a test strategy, the specific cost models are connected to form the cost structure of a test strategy. This approach was chosen, because it eases the evaluation of the costing parameters, and because several equations (secondary parameters) in the cost model depend on the test methods used. In addition, this approach allows to evaluate modifications of a test strategy such as the affect of removing single test stages in a test strategy.

In the specific cost models, the parameters and equations are grouped together by the following aspects:

- Parameters, which belong to the same topic, such as design, production or test are grouped in one cost model.
- Parameters for which the data are provided by the same group of persons in a company. For example, the hourly labour rates for the different functions form a

cost model.

- Parameters which differ from test method to test method will be grouped in a cost model. This allows to configure the cost model for a test strategy (the global cost model) by connecting the test method cost models to the other cost models.

The cost models are connected by using parameter values from one cost model in the equation of a parameter of another cost model. This approach of cost modelling is generic and extremely flexible. It can be used for any other application in the field of cost modelling and cost evaluation.

The following section will describe this cost modelling mechanism in detail. In section 7.5.2 the cost models as they are used in ECOvbs will be described.

7.5.1. The Cost Modelling Technique

A cost model in ECOvbs comprises of the following features to describe the costing relations of the parameters:

- A parameter is defined by the parameter name and an assignment to it of either a term or a value. If a value is assigned, the parameter is called *primary*, if a term is assigned, the parameter is called *secondary*.
- A term is composed of parameters and operators. The operators are grouped into algebraic, comparison, functions and others.
- The following algebraic operators are implemented: plus, minus, product, division and modulo division.
- The following comparison operators are implemented: equals, greater than, less than.
- Some functions, which are specific to the cost models for VLSI based systems, are hard coded to be used for the terms. This reduces the complexity of the cost models, because the implementation of a complex function is replaced by a function call. But more important is the reduction in the calculation time. This is achieved by the hard coded implementation instead of interpretation of the

function. The calculation time is reduced by a factor ten in average. The following functions have been implemented: poisson distribution and the logarithm with the base of ten.

- In addition, the following operators have been implemented: power, factorial and conditional assignment.
- The equations are nested by the terms "BEGIN <user friendly cost model name>" and "END". The user friendly name is used in combination with a match name to refer to the cost model.
- The syntax provides features to access parameter values from other cost models in order to build an equation. This cost model connection can be static, i.e. the cost model from which the parameter is accessed, is fixed, or it can be dynamic, which means, that the cost model, from which the parameter is accessed, is test strategy dependent and is therefore defined in the main cost model of the test strategy. The following example should illustrate this: The design cost depend on the hourly labour rate for designers. These two parameters are defined in different cost models, but the connection is static, because the labour rate is independent from the test strategy. The fault spectrum after a test depends on the fault spectrum of the previous test stage. But the previous test stage depends on the test strategy, and hence this type of connection is dynamic.
- In addition to the parameter descriptions (i.e. the equations), a cost model may consist of a cost model reference. This feature is needed for the main test strategy cost model in order to specify, which cost models are included (depending on the test stages), and in which order they are calculated.
- In order to allow the same cost parts to be calculated in a loop, such as the test repair loop, the cost modelling approach of ECOvbs provides an appropriate syntax.

A static cost model connection is made by providing both the cost model name and the parameter name. A dynamic cost model connection needs to be specified at two

locations. The first location is the using cost model, where the value of the parameter will be accessed. The second location is the cost model, in which the providing cost model for the parameter to be used will be defined. At the first location, the parameter name to be used and a type classification of the connection must be specified. At the second location, the using cost model, the providing cost model and the type of the connection has to be defined. This specification will perform all dynamic connections between the providing and the using cost model such, that all parameters, for which its name and the specified type matches, are connected.

The following example should illustrate the connection mechanism. There are the following three cost models:

```
BEGIN CMprovide
a = 5
b= 3
c = 6
d = 2
e = $*e.e
END
```

```
BEGIN CMuse
a = $CMprovide.b
b = $*typex.a
c = $*typex.c
d = $*typey.d
END
```

```
BEGIN CMmain
a = 2
$CMprovide.e = a
$DMprovide
$CMuse = CMprovide(typex)
END
```

CMprovide is the providing cost model, CMuse is the using cost model, and CMmain is the main cost model. The using cost model consists of 4 parameters, for which parameter values are assigned from another cost model. The assignment to parameter *a* consists of a static link. The $ character is used as an identifier, that the following name is a cost model name. The assignment to *a* means, that the value of parameter *b* in the cost model CMprovide is assigned to parameter *a* of the cost model CMuse. The assignments to the

parameters *b* through *d* are dynamic connections. The meaning of the syntax for parameter *b* is as follows:

> The value of parameter *a* in a cost model, which is not defined yet, will be assigned to parameter *b* in the cost model CMuse. The type of this connection is *typex*. This type classification is used, when the connecting cost model is defined.

In the main cost model, the assignment terms of the parameters *b* and *c* in CMuse are connected by the assignment "$CMuse = CMprovide(typex)", which means, that all dynamic connections in cost model CMuse, which are of typex, and for which the connecting parameter exists in CMprovide, the related value is assigned. Then the cost model CMuse is calculated. The specification of $CMprovide in the forth line of the main cost model means, that CMprovide has to be calculated. The second line in CMmain provide a forward connection, which connects the parameter *a* of CMmain to the parameter e of CMprovide.

The loop feature can be used to model the costs of cyclic processes such as the test/repair loop. This means, that a certain costing parameter depends on itself. This costing parameter is initialised, is calculated n times, and the value after n calculations is used as the resulting cost value. An example therefore is the calculation of the fault spectrum of a test stage after the test has been performed. At the beginning of the test, the devices going to test have a certain fault spectrum, which depends on the previous test or manufacture stage. After a test application, some devices are good, some are bad. The fault spectrum of the 'good'/passed devices is now different from the fault spectrum before the test. The same is the case for the bad devices, for which the defect, which was detected, has been repaired. These devices are subject to be tested again. This test/repair loop is repeated until all devices pass the test, or until the devices have been tested n times. The cost model syntax for the loop feature is similar to the loop feature of the programming language C.

The basic cost modelling technique was adopted from ECOTEST. See chapter 2 for a

detailed description of the methods.

7.5.2. Description of the Cost Models

The cost models which have been implemented allow the user to evaluate a large number of test strategy alternatives. They can be easily amended by the user, or the user can add new cost models and link them to existing cost models, by using the techniques described in the previous section. In the following the author will describe the cost models which have been implemented for ECOvbs. The equations are listed in appendix B of this thesis.

<test_strat>: The test strategy cost model is the top level cost model in the hierarchical cost model structure. It contains all cost models which are used for a test strategy cost calculation, and it specifies the calculation order of the cost models and provides the linkage of the dynamic connections.

repair: This cost model contains the repair cost per defect type. These data are derived from the component cost plus a replace cost for component defects, and the repair cost for manufacture defects. These data are used to calculate the actual repair cost at a test stage, based upon the actual defect spectrum.

board_data: This model comprises of the board related data. These are the number of components per component type, the percentage of the board, which is combinational, pipelined, synchronous and asynchronous, the production volume, the maximum daily volume, the number of solder joins, number of test pads, number of nodes or nets, number of layers, number of sides on which components are placed. the minimum wire separation. the total number of component pins, the total number of gates. the total number of components. and the total number of digital component pins.

design_data: The design data cost model includes all parameters specifying the design efforts for design. layout. verification and prototype production. In addition, the prototype material cost. the number of prototypes to be produced. and the average number of redesigns expected.

prod_data: This cost model contains the following production related data: production prepare cost, bare board cost, total component cost and the number of components per assembly type.

def_spectrum: In this cost model the defects per defect type are specified. From that, the total number of defects per board and the production yield is determined.

All cost models which are described so far are design or test strategy dependent. Therefore they are generated by ECOvbs. This generation is based on the design specification data and the test strategy. The following cost models are common for all designs and test strategies. Only the usage of the test cost models depends on the test strategy. But the definition, whether a test cost model is used or not, is defined in the test strategy model. The next three cost models (iterations, labrates and assembly) contain input data, which are not design dependent, but which need updating when the related data change. The other cost models mainly consist of equations which are based on data from the cost models above.

iterations: This model lists the iterations factors for the various design phases [Mil91]. The iteration factors define the effort relation between the original design and a redesign.

labrates: In this file the user enters the labour rates for the various activities such as design, layout, verification or prototype production. The test related labour rates are stored with the test cost model, because they are test dependent.

assembly: This file comprises of the assembly cost per assembly type. The data are provided by the user.

prod_cost: Based upon the board data, the production data and the assembly data, this cost model calculates the total production cost of a board.

defect2fault: This file contains a matrix like table which allows to convert the defect spectrum into a fault spectrum. The difference between the defects and the faults is, that the defect is related to *what can be repaired*, and the fault is related to *what can be tested*. For example, a defect can be a defective digital component.

The fault related to that defect can be an open or short, a functional fault or a parametric fault. This distribution of the defect types is defined in this cost model. In addition, the total number of faults per board, which is the sum over all fault types, and the yield are calculated.

<comp_type>2fault: In addition to the cost model defect2fault, a cost model per component type is defined which contains the number of faults per component type. These cost models are needed for the incoming tests.

fault2defect: This cost model converts the fault spectrum into the defect spectrum, and it is needed to calculate the actual repair cost per test stage.

<test_cm>: This cost model consists of all parameters, secondary and primary, which are related to a certain test. This includes all parameters for test generation, test application, diagnosis and repair. In addition, the fault spectrum after the test is calculated, which is based upon the fault spectrum before the test application and the fault coverage of the test. Because a test stage is a combination of a test method and a test equipment, a test cost model has to be defined for every valid combination of test method and test equipment.

tot_cost: In this cost model the total costs are calculated which are based on the cost of the various stages of design, production and test.

The user can examine the cost models in detail during the test strategy planning process. By providing user friendly names and measuring units for the cost model parameters in separate files, the costing data are presented in a user friendly way. In addition, the user can define, which cost parameters are shown to the user and which are hidden. This technique allows to evaluate the economics of test strategies with different detailing levels.

7.6. Calculation of Fault Spectrum and Defect Spectrum

The terms defect and fault are often used for the same effects. However there is a difference in the meaning of a defect and a fault. Therefore the author will introduce the

definitions for these terms, as they are used in this thesis. Similar definitions are given in [Ben89]:

> *A **defect** is a discernible physical flaw which causes the device not working correctly under certain conditions.*
>
> *A **fault** is the effect, which is caused by a defect, and which can be measured by a test system.*

As an example, a broken wire of a bare board is defect, which will leads to an open fault.

The defects are related to the manufacture and repair phase, because defects come into existence during the manufacture process, and the defects are eliminated through a repair. But in most cases - beside gross defects - a defect cannot be identified directly. For these cases, test systems are used to measure the fault, which is related to the defect. If the defect should be repaired, a fault diagnosis is performed in order to isolate the fault, and to relate it to the actual defect.

Due to the complexity of today's electronic systems, and therefore the complexity of defects and faults, there are special test systems which are optimised for the detection of certain fault classes (see chapter 2). In order to predict the fault coverage of a test method, it is important to know the fault rate per fault type of the device under test. These data are known as *the fault spectrum.* The fault types which form a fault spectrum depend on the product mix. A typical fault spectrum consists of opens, shorts, static faults, dynamic faults, voltage faults and temperature faults.

Now the question remains, how to predict the fault spectrum of a system which has not been designed yet. In previous work this prediction was always made by taking the numbers from the systems, which are already in fabrication, and for which the fault spectrum can be measured. But this method implies two problems, which can make the prediction rather unreliable:

1. The faults, which are measured in a production process are not necessarily the faults which the device under test contains, because only those faults are

measured, which can be detected during the test. But the detection of a fault depends on the fault coverage of the test. Hence the measured fault spectrum depends on the fault coverage of the tests. This matter of fact was observed by the author by several talks to quality assurance people. In many cases their assumption was, that the total number of faults in a system is the total number of faults detected during all the test stages in the production.

2. If a system is based on a new technology, it definitely cannot be assumed, that the fault spectrum will be the same as for the system, which is based on an older technology. This is because the production process, the component defects and the composition of the system will be completely different.

For these reasons, a prediction of the fault spectrum by measuring the fault spectrum of the previous system can be very inaccurate, especially if a new technology is used.

For these reasons, the author has developed a novel method for predicting the fault spectrum of a system. This method is based on the conversion of the defect spectrum into the fault spectrum. The defect spectrum describes the number of defects per defect type. The defect types arise from a classification of the components and the production process in to several classes. The defect spectrum can be predicted by knowing the composition of the system, the defect rates of the components and defect rates of the production process. The conversion of the defect spectrum is based upon previous data. These conversion characteristics are independent from the technology.

7.6.1. Calculation of Defect Spectrum after Manufacture

The defect spectrum of a board describes the average number of defects per board and per defect type. The defects are classified into types by the following aspects:

- component type
- manufacture step type
- repair type

The classification into component and process types is because the different components and the different manufacture processes have different defect rates. A further classification into repair types is only needed, if for the same component type or the same process type different repair costs occur.

By knowing the defect rates per component and the defect rates per manufacture step, the total number of defects per defect type, which forms the defect spectrum is calculated as follows:

$$d_i = \frac{dpm_i \cdot n_i}{10^6} \qquad (31)$$

where d_i stands for the number of defects for defect type i, dpm_i stands for the DPM rate (defects per million) for defect type i, and n_i stands for the number of elements of type i.

In ECOvbs we have defined the following defect types:

1. Component types:

   digital ICs, analog ICs, passive ICs, edge connectors, PLAs, RAMs, ROMs, micro processors, ASICs, resistors, capacitors, bare board.

2. Manufacture types:

   pick and place, solder joint.

The defect types are defined in the cost models. Therefore it is very easy to add other defect types, or to remove any of the above.

7.6.2. Calculation of Fault Spectrum

The fault spectrum is derived from the defect spectrum by defining for each defect type, how a related defect is distributed among the fault types. We call this definition the *conversion matrix*. The dimension of this matrix are the number of faults (for each row) and the number of defects (for each column). Due to being a distribution of the whole, which is the number of defects, the sum over each column of the matrix must be 1. The fault spectrum can be derived from the defect spectrum by a matrix multiplication. The

defect spectrum and the fault spectrum are modelled as vectors **d** and **f**, and the conversion table is modelled as a matrix **C**. The fault spectrum **f** is derived from the defect spectrum **d** and the conversion matrix **C** as follows:

$$\mathbf{f} = \mathbf{C} \times \mathbf{d} \qquad (32)$$

This method of deriving the fault spectrum, together with the calculation of the defect spectrum, allows a very accurate prediction of the fault spectrum **before the system has been designed**. This has been proven by making this prediction for computer systems at Siemens. The real numbers, which were derived when the system was in production, differed only by 2%.

Using the defect rates of the components and the manufacture steps for deriving the defect spectrum and the fault spectrum, has another major advantage beside the accuracy of the prediction: it allows to calculate the impact of different component or manufacture qualities to the manufacture yield. This enables to optimise the mix of yield and fault coverage for a target quality by optimising the total cost and hence to define the optimum test strategy.

7.6.3. Calculation of Defect Spectrum for Repair

In ECOvbs the repair costs are calculated not by using an average costs per repair but by determining the type of the defect to repair, and by using a repair costs per defect type. This method of repair cost prediction is more accurate than using an average value, because repair costs can be quite different depending on what needs to be repaired. For example, an exchange of a defective VLSI component is much more expensive than the exchange of a defective resistor. In addition, this method allows to evaluate the impact on the repair costs for improved manufacture quality, or the impact of alternative test strategies. There may be a difference in the repair costs for different test strategies, if a defect - e.g. a defective component - is detected in a different manufacture stage. For example, an income test of resistors may not only allow to return low quality charges to

the supplier but may also significantly reduce the repair costs of the assembled board, if an exchange of components at board level becomes expensive.

In order to calculate the repair costs as described above, we must derive the defect spectrum from a certain fault spectrum. Based on the fault spectrum of the device under test before the test application, and the fault coverage of the test, we can derive the number of faults per fault type. But the repair costs rather depend on the defect type than on the fault type. Therefore the number of defects per defect type needs to be derived from the faults per fault type. If the repair cost per defect type is given, the total repair cost can be calculated as follows:

$$tr = \sum_{j=1}^{n} d_j \cdot r_j \qquad (33)$$

where tr stands for the total repair costs, d_j stands for the number of detected defects for defect type j, and r_j stands for the repair cost per defect for defect type j.

The defect spectrum of the detected defects, i.e. the number of defects per defect type, can be derived from the fault spectrum of the detected faults, the fault spectrum before the test and the defect-to-fault conversion matrix as follows:

$$d_{\mathrm{det}_j} = \sum_{i=1}^{n} f_{\mathrm{det}_i} \cdot \frac{f_{ij}}{f_i} \qquad (34)$$

where d_{detj} stands for the detected defects of the defect j, f_{deti} stands for the detected faults of type i, and f_i stands for the number of faults of type i before the test. f_{ij} stands for the number of faults of type i before the test, which are related to the defect type j. This value can be derived by using the defect-to-fault matrix as follows:

$$f_{ij} = d_j \cdot c_{ij} \qquad (35)$$

where d_j stands for the number of defects of type j before the test, and c_{ij} is the element in the jth column and the ith row in the defect-to-fault conversion matrix.

The values for f_{ij} are the same for all test methods, and therefore separate cost models, one per defect type, are provided in order to calculate these values.

7.7. The Test Strategy Planner

The test strategy planner creates a test strategy for the cost evaluator by creating the related cost models, creating a cost model connection file, which is itself a cost model, and verifying the applicability of the test stages and the consistency of the test strategy. The three following paragraphs will define the terms *test stage, test strategy* and *applicability*.

A *test stage* is a combination of a test method with an appropriate test equipment, which is applied at a certain production stage, which is defined by the user of ECOvbs. The production stage can be component, test cluster, board and system.

A *test strategy* is a combination of test stages which can be applied at several production stages. A test strategy requires the applicability of the test stages and the test strategy as a whole.

A *test stage is applicable*, if all DFT requirements and construction requirements of the test method and the test equipment are applicable. A *test strategy is applicable*, if all test stages of the test strategy are applicable, and if at least one DFT alternative per component type is available, which fulfils all DFT requirements of all test methods which are applied.

A test strategy is set up by the user. The system lists, which data may be entered, and checks the entered data about correctness. The data which need to be entered in order to define a test strategy are the following:

- Match name and user friendly name of test strategy.
- Number of test stages.
- Per test stage:
  - Match name and user friendly name of the test stage.

- Production stage at which the test should be applied.
- Test method
- Test equipment

After having entered the test strategy, it will be verified concerning its applicability. If the test strategy is applicable, the user is prompted to decide whether the test strategy should be generated. The test strategy generation performs the following actions:

1. Selection of a DFT alternative per component and for the whole board. If more than one option is applicable, the user is prompted to select one.
2. Generation of the test strategy description file and the test strategy cost model. The contents of these files will be described later.
3. Generation of the test strategy dependent cost models. These cost models are board_data, def_spectrum, design_data, prod_data and repair. Its contents are described in section 4. This generation includes the calculation of several parameters:
   a. The calculation of the DFT alternative dependent parameters.
   b. The calculation of other summarising parameters such as the total number of gates, or the total number of digital pins.
   c. The calculation of the DFT alternative parameters per test stage. These values allow to calculate the cost per test stage. An further description will be given in the following paragraph.

In order to calculate the cost, which are directly related to a test stage, the incremental cost for applying the test stage must be determined. These are the cost for test generation, test application, diagnosis and repair, and the incremental cost for design and production. The incremental cost for design and production occur, if a test method requires a certain DFT method, which requires a different DFT alternative for some of the components or the whole board. These incremental cost is determined as follows:

- For each component and the whole board, the DFT requirements of the test method under consideration are removed.

- For each component this DFT alternative is selected, which has the lowest component cost.
- For the whole board this DFT alternative is selected, which has the lowest total component cost. The cost per component is taken from the DFT alternative with the lowest cost.
- All cost model parameter values are derived.
- The DFT requirements of the test method under consideration are set and the user selected DFT alternatives are chosen.
- The incremental costs are derived from the difference of the cost model parameters before setting the DFT requirements and after setting the DFT requirements.
- Each DFT dependent input parameter in the test strategy dependent cost models will be composed of the basic value - i.e. the value without any DFT requirements - plus the incremental value per test stage. The incremental value as the term of a sum is multiplied by a Boolean parameter, which allows to switch the term on and off. By this method, all costs, which are related to a certain test stage, can be removed by setting the related Boolean parameter to zero.

The test strategy cost model includes for each test stage one parameter which is connected to all Boolean parameters of the other cost models, which are related to the same test stage. This allows to set or to reset the costs related to the test stage by setting this single parameter to one or to zero.

In addition to the test strategy dependent cost models, two files are generated. The first file contains a list of all cost models used by the test strategy. This list needs to be ordered so that a cost model which accesses a parameter from another cost model is listed after the related cost model. The other file contains all the setup data of the test strategy which are entered by the user. This file can be read by ECOvbs to set up a test strategy. By modifying this file, the user can modify the related test strategy.

When a test strategy is set up and the cost models and other test strategy files are generated, the economics of the test strategy can be evaluated. The evaluation methods will be described in the next section.

7.8. The User Interface

The user interface is implemented alphanumerically for character I/O and by X11 for graphical drawings. ECOvbs comprises of a command handler system including four different command handlers:

- The ECOvbs handler includes general commands for data handling and maintaining the handler.
- The DSR handler provides several commands to modify and access design data.
- The TSP handler provides commands to define and verify test strategies.
- The CM handler includes all commands to read test strategies and to evaluate their economics.

A command handler consists of several commands. Every handler is provided with all commands of the ECOvbs-handler. A command is composed of the command name and arguments, which are classified into optional and required arguments. The help command, which is provided for all handlers, prints a list and short description of all commands. A command name can be entered in an abbreviated form. You must enter at least the first character of the command (e.g. you can enter 'h' instead of the full name 'help' to call the help command).

To enter a new handler the command 'enter' can be used or the handler name can be entered (e.g. by entering 'tsp' you enter the TSP handler). If a command of another handler is to be called, either the accompanied handler can be entered and the command is called or the handler name is entered with the command name as argument (e.g. entering 'cm print' from the TSP handler executes the command 'print' of the CM handler). The user interface is provided with help text. For every input requested, a default value is given, which is printed together with the question string. All data entered

are checked about correctness.

In the following section the commands will be described. The meta-language of a command is as follows:

command-name {<required argument> | [<optional argument>]}*,

where [] means, that the argument is optional.

7.8.1. The User Commands of ECOvbs

7.8.1.1. The ECOvbs handler

The ECOvbs-handler provides several general commands. These commands can also be called from any other handler. The ECOvbs-handler is entered when you start ECOvbs.

alias [<alias-name>] [<command-name>]

Alias lets the user define a new command. Whenever a new command is entered, the related command is executed. If no argument is specified, a list of defined aliases is given. If one argument is defined, the related alias-command is deleted. If two arguments are defined, the specified definition is added or replaced.

enter [<handler-name>]

This command allows the user to set a handler. If the user wants to specify commands attached to this handler, he may omit the handlers name from the command line. The commands of the ECOvbs handler are available with all handlers. By omitting the optional handler-name a list of available handlers is printed.

shell [<shell-command>]

This command lets the user execute a shell commands or execute into the shell without terminating the execution of ECOvbs. If a <shell-command> is defined, it will be passed to the shell for execution but you will stay in ECOvbs.

echo [<string>]

This command prints messages on screen. It is primarily useful for scripts. (see read command).

read <file-name>

This command reads and executes commands from a file.

help [<command-name>]

This command prints a list of available commands (and aliases) together with a brief comment what these do. If a command-name is supplied, a more explicit help text about the defined command is printed.

mod_startup

This command allows to modify the startup data and to store these data in the file paths.startup. The startup data are the default design, the path to project data, the path to library data, the project name and the library name.

quit | Ctrl-Z

The user can terminate the execution of this program by either typing 'quit' or holding the key labelled <Ctrl> down and pressing 'z' at the same time.

Ctrl-Q

The user can interrupt the execution of ECOvbs or a command by typing ctrl-q. ECOvbs then asks to return to shell (i.e. terminate ECOvbs), return to command-handler (i.e. terminate the execution of the actual command) or to continue.

7.8.1.2. The DSR handler

The design specification reader (DSR) includes all commands needed to modify and to

access the design data. The following commands exist:

load

This command reads and checks the board description files as defined in the startup paths.

print [<option>] [<component name>]

Print board data. The user specifies the type of data to be printed by the following options:

-all: print all board data.

-board: print all board related data.

-comp [<component name>]: print component related data specified by <component name>. If the argument is omitted, all components are listed and the user is prompted to select one. The default option is -all.

7.8.1.3. The TSP handler

The test strategy planner (TSP) includes all commands to define and verify test strategies. The following commands are implemented:

ts_print [<test strategy name>]

Print the test strategy specified by the argument. If the argument is omitted, all test strategies are listed and the user is prompted to select one.

create <test strategy name>

Create a new test strategy and verify its applicability. The match name of the test strategy is defined by the argument.

verify

As in the create function, but only the verification step is taken. If the argument is

missing, the system lists all test strategies and asks the user to select.

ts_delete [<test strategy name>]

Delete the test strategy specified by the argument. If the argument is omitted, all test strategies are listed and the user is prompted to select one.

tmd_print [<tmd_name>]

Print test method description data of test method <tmd_name>. If <tmd_name> is omitted, the test method names are listed and the user is prompted to select a test method description.

teq_print [<teq_name>]

Print test equipment description data of test equipment <teq_name>. If <teq_name> is omitted, the test equipment names are listed and the user is prompted to select a test equipment description.

activate [<test stage>] [<test strategy>]

Activate test stage <test stage> of test strategy <test strategy>. Activation of a test stage means, that all costs related to the test stage (DFT cost, test cost) are considered for the cost calculation. All test stages are active when loading a test strategy. If the command 'cm reset' is applied, all test stages are activated. If <test stage> is omitted, the test stage names are listed and the user is prompted to select a test stage. If <test strategy> is omitted, the test strategies are listed and the user is prompted to select one.

deactivate [<test stage>] [<test strategy>]

Deactivate test stage <test stage> of test strategy <test strategy>. Deactivation of a test stage means, that the costs related to the test stage (DFT cost, test cost) are omitted for the cost calculation. All test stages are active when loading a test strategy. If <test stage> is omitted, the test stage names are listed and the user is prompted to select a test

stage. If <test strategy> is omitted, the test strategies are listed and the user is prompted to select one.

7.8.1.4. The CM handler

The cost model (CM) handler provides all commands to read test strategies and to evaluate their economic impact. The following commands are implemented:

load [<test strategy_name>]

Load the test strategy specified by the argument. If the argument is omitted, all test strategies are listed and the user is prompted to select one.

delete [<test strategy_name>]

Delete the test strategy specified by the argument. If the argument is omitted, all test strategies are listed and the user is prompted to select one. In contrast to the delete command of the TSP handler, the deletion will only delete the internal data structures of the cost models related to the test strategy, and it will cut the test strategy out of the list of test strategies.

draw_cm [<test strategy_name>]

Draw cost model structure of test strategy <test strategy name>. If <ts_name> is omitted, the user is prompted to select one.

print [<cost model_name>] [<test strategy_name>] [<test strategy_name>] [<test strategy_name>] [<test strategy_name>]

Print all user parameters of the cost model specified by the argument. If all arguments are omitted, all cost models of the test strategy loaded last are listed and the user is prompted to select one. If no test strategy is specified, the data are listed for the test strategy loaded last. If one or more test strategies are defined, the cost model parameter values are listed for all test strategies specified. The maximum number of test strategies

to be listed in parallel is four.

sens

- Draw the impact of the variation of one parameter to another parameter (so called 'sensitivity analysis') graphically. The user is prompted during the execution of the sens function for the following data:
  - list of test strategies to analyse
  - cost model which includes the variation parameter
  - variation parameter
  - cost model which includes the resulting parameter
  - resulting parameter
  - range of variation parameter

draw_flt_spec [<ts_name>]

Draw fault spectrum and related cost of test strategy <ts_name>. The fault spectrum and the related cost are drawn for all test stages. If <ts_name> is omitted, the user is prompted to select one.

set [<cost model>] [<parameter>] [<value>] [<test strategy>]

This command enables the user to vary single parameters by modifying the accompanied value. The parameter is specified by the arguments <cost model>, <parameter> and <test strategy>. The value is specified by the argument <value>. If all arguments are omitted, all cost models of the test strategy loaded last are listed and the user is prompted to select one. If <test strategy> is omitted, the last test strategy loaded is selected. For every new value the cost model will be recalculated. This function is needed to evaluate the cost impact of test methods for various design possibilities. The user can use the print commands to see the effect of the new value. To retrieve the initial status of the cost model, the user has to execute the command 'reset' (see next command).

reset [<test strategy>]

The execution of this command recalculates the test strategy specified by the argument. This command may be used to reset the cost model values to the test strategy data in the case of having executed the command 'set'. If the argument is omitted, all test strategies are recalculated.

7.9. Summary

This chapter has described the economics driven test strategy planner ECOvbs. The author has discussed the need for such a system and has presented its philosophy. He has described the methods which were developed by the author and implemented in the system, and a description of the functions, which are provided to the user of the system in order to make economic evaluations, has been given. The hierarchical cost modelling concept, which is implemented in ECOvbs, has been introduced. ECOvbs is currently in use in an industrial environment, and it provides a useful advice tool for test strategy planning in the early stages of the design of VLSI based systems. In the next chapter, the author will present some results which have been obtained using ECOvbs, ECOTEST, and the spreadsheet software Excel, which was used for evaluating the cost models which have been described in chapter 4.

Chapter 8
Test Economics Evaluation

8.1. Introduction

The scope of this chapter is to present and to discuss the results from analyses using the test economics models which were described in chapter 4, using ECOTEST and using ECOvbs.

The life cycle cost models described in chapter 4 were used to study boundary scan test strategies. The results are presented and discussed in section 8.2.

ECOTEST was used for two types of experiments. The author has used the system for several large industrial designs in order to proof its applicability. The results which the author has achieved will be presented in section 8.3. The second experiment is related to performing test strategy planning with inaccurate data by using the techniques, which were described in chapter 6. For that experiments the author has used artificial but realistic designs in order to show some effects, and some industrial designs in order to proof the applicability of the methods.

In section 8.4 test strategy planning results for a large computer board by using ECOvbs will be presented and discussed.

8.2. The Economics of Boundary Scan

The author has implemented the test economics models which are described in chapter 4. as spreadsheets by using the MS-DOS spreadsheet software Excel and Lotus 1-2-3. He has received input data for the cost models from an industrial source (from SNI). The data were used to study the economics of implementing a boundary scan test strategy versus an In-circuit test strategy for the SNI data. The test strategies, the most important data and the results will be described and discussed in the following paragraphs.

This analysis was performed for a complex microprocessor based computer board. A special environment of the development and production of the board lead to the following simplifications of the test economics models:

- For the design phase, the computer equipment costs are included in the hourly labour rate of the designers. Therefore the related costs are set to zero.
- The cost for diagnosis and repair equipment was set to zero for the same reasons.

Appendix C presents the data for all parameters. The author will now describe the cost areas were major differences between the two test strategies occur.

The cost difference in design cost is mainly due to different development times. They are higher for the boundary scan test strategy, especially because there was no experience in implementing boundary scan. Another difference is in the test preparation cost due to high production cost for the bed-of-nails fixture for the In-circuit test strategy.

The production cost differ mainly because of different component prices, which are higher for the boundary scan components. This leads to an increase in the production cost of 700 DM or 14% for the boundary scan board.

The fault spectrum shows differences between the test strategies, due to the higher complexity of the boundary scan board. But these differences are negligible, so that there are no significant effects on the cost difference in test application. These are mainly due to the different test equipment costs. The test equipment for the In-circuit test is an expensive In-circuit tester, whereas the boundary scan test can be performed by an IBM PC with certain extensions, such as a special interface to the boundary scan edge connector, a serial to parallel converter of the test pattern, and software to operate the test and diagnosis.

In this case, the total board related cost is 10% higher for the boundary scan test strategy. If the decision would have been made only by considering the board related cost, the In-circuit test strategy would have been selected.

In our case we have considered also the costs on system level. These are the test application and the field costs.

By making use of the boundary scan features for diagnostics on system level, the system test costs are significantly lower for the boundary scan system. This matter of fact shows its main advantage in the field, where a fault diagnosis and repair can be performed in the field in many cases. For a system without boundary scan, the field diagnosis is very mostly limited to detecting the faulty board. This implies a replacement of the faulty board in the field and repair in a service centre or in the depot, which is in our case the production facility. This procedure makes a field breakdown much more expensive than a repair in the field.

Considering these system level costs for comparing the economics of the boundary scan test strategy versus the In-circuit test strategy makes the boundary scan test strategy more economical than the In-circuit test strategy by 24.365.527 DM versus 25.383.800 DM or by approximately 4%.

This economic evaluation shows the significance of the field costs in the economics of test strategies, and it has proven the arguments for using a life cycle cost model for the evaluation of test strategies.

The following graphs will show the effects of the variation of single parameters on the total cost. This allows to make statements about the effects of single parameters on the economics of the test strategies. This evaluation is the classical sensitivity analysis.

In a first step, the author has selected the parameters, which are subject to a sensitivity analysis. The parameters were selected by the following criteria:

1. Typical range of the parameter. If a parameter has a large range of possible values for different environment, it is important to know, how the variation of this parameter affects the total cost.
2. Inaccuracy or uncertainty of the parameter. If the value of a parameter is an

inaccurate prediction, or if the value is uncertain, it is important to know, how a variation of this parameter affects the total cost.

3. Character of a parameter prediction. There are situations, where a parameter value may not be known, when the analysis is performed. In that case, it is interesting to know, how different parameter values affect the total cost.

The author has selected the following parameters to perform a sensitivity analysis:

labour rate, manufacture cost of the board, additional manufacture cost for boundary scan, mean time between failure, repair time of boards, which breakdown in the field and the interest rate.

The parameters have been varied by ±50% of their nominal value. Figures 41 and 42 show the total cost as a function of the sensitivity parameters for the In-circuit- and the boundary scan test strategy, where the x-scale shows the variation of the parameters in percentage of their nominal value. Therefore, all the curves cross at a variation of 0%, which is the total cost for the nominal values. If the gradient for a parameter is large, the total cost is very sensitive for a variation of this parameter, and vice versa. Figures 41 and 42 show, that the manufacture cost is the parameter most sensitive parameter, and also the interest rate is very sensitive for both test strategies. It is interesting to see, that the mean time between failure is much more sensitive for In-circuit test than for boundary scan.

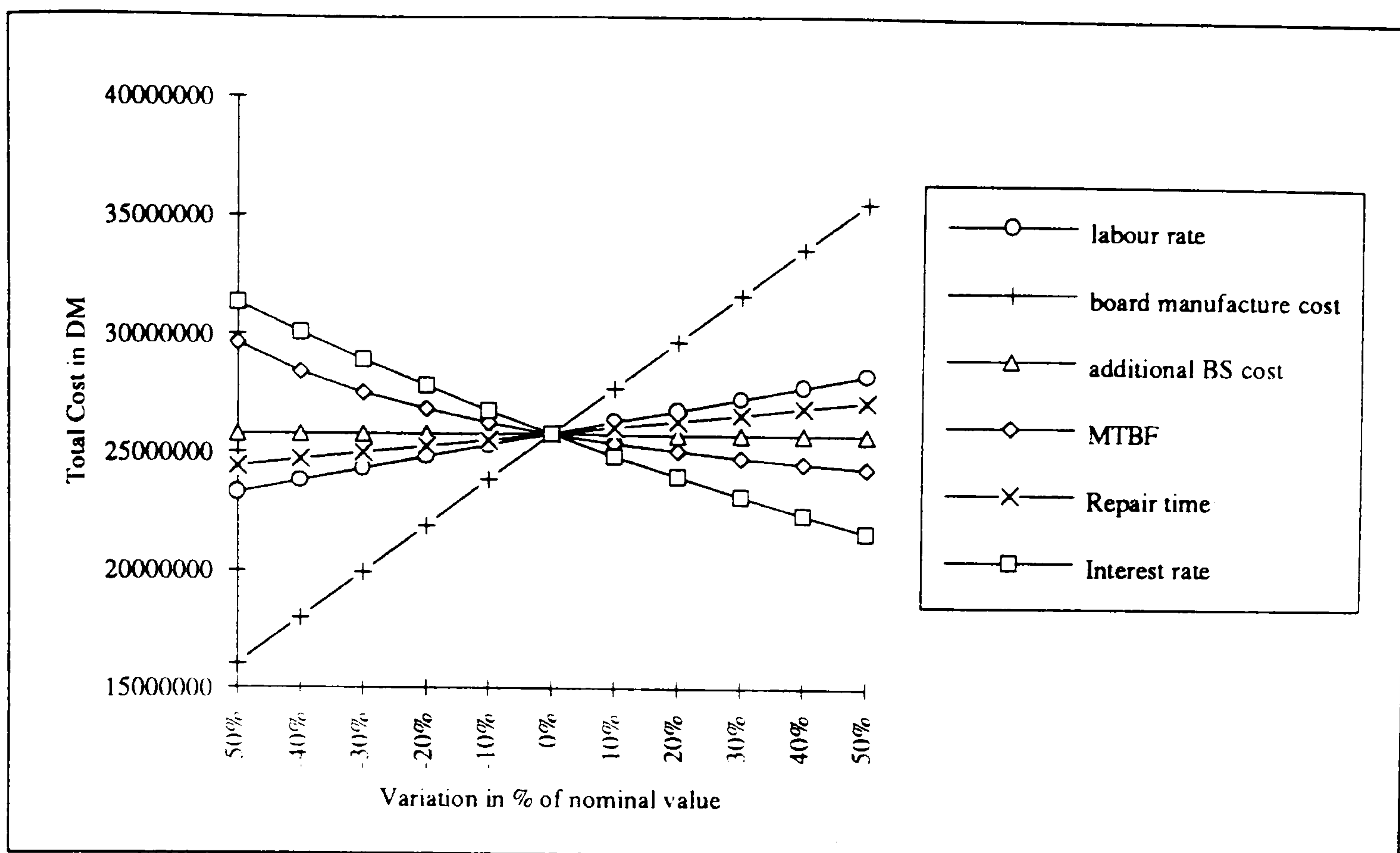


Figure 41: Sensitivity of the total cost in DM to the variation of parameter values in percent of the nominal value for In-circuit test

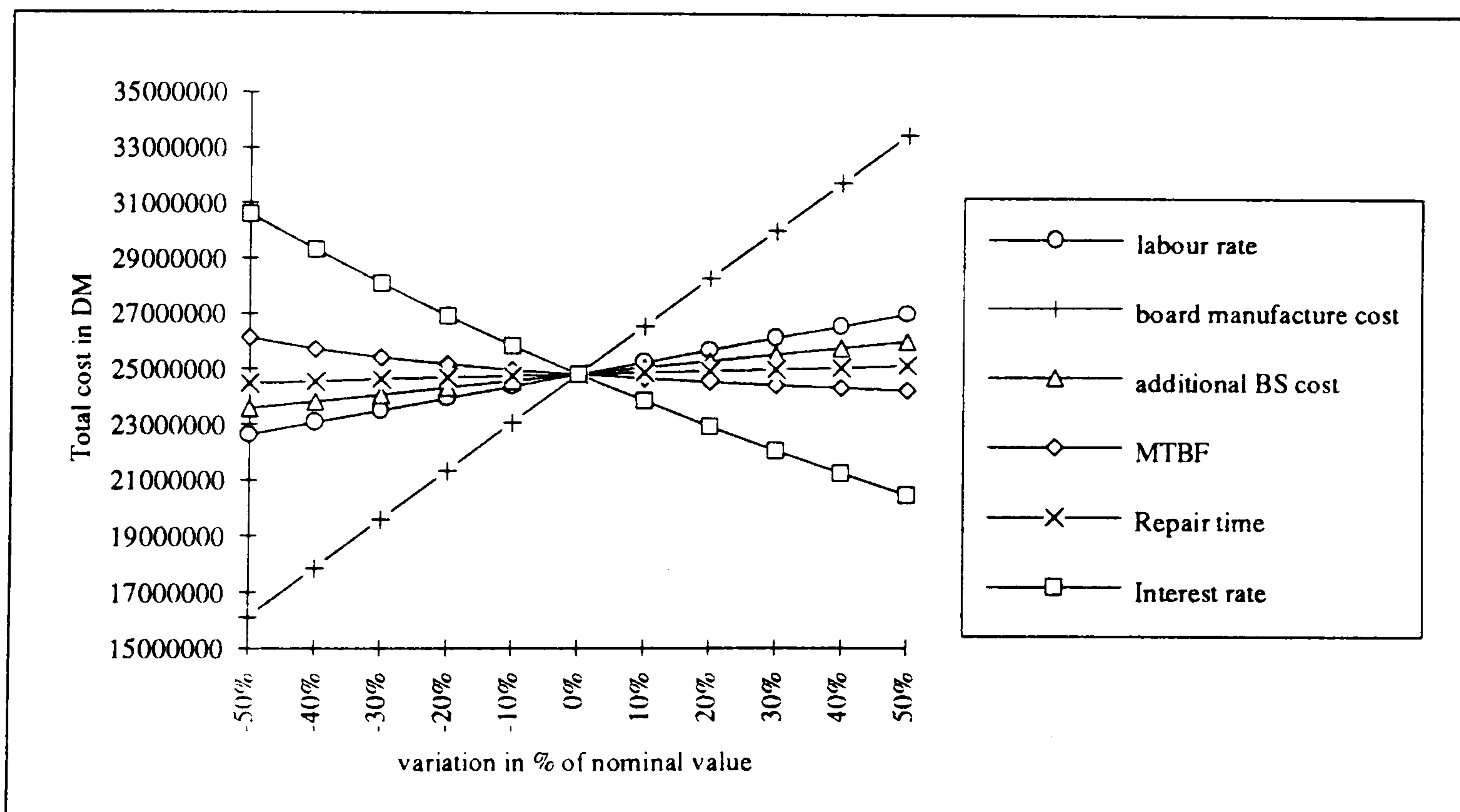


Figure 42: Sensitivity of the total cost in DM to the variation of parameter values in percent of the nominal value for boundary scan

The figures above can answer the question which parameter the total cost is most sensitive to, but they do not answer the question which parameters are critical to the test strategy decision to be made. To answer this question, it is important to analyse the sensitivity of the total cost difference between the two parameters. In figure 43 the

sensitivity of the cost difference, to parameter variations as described above is presented. The cost difference is defined to the total cost of the In-circuit test strategy minus the total cost of the boundary scan test strategy. If the cost difference is positive, the boundary scan test strategy is more economical than the In-circuit test strategy. When a curve crosses zero, the In-circuit test strategy becomes more economical. This figure says the following:

> In the range, in which none of the curves cross the zero line, the boundary scan test strategy is more economical than the In-circuit test strategy. This means, that the decision for boundary scan is very stable, if we can assume, that none of the parameters, which are analysed, deviates more than 35% of the nominal value.

It is interesting to see, that the interest rate, which was one of the most sensitive parameters for the total cost for both test strategies, is very insensitive to the cost difference. This means, that the interest rate is important for the total cost, but is unimportant for the test strategy decision to be made.

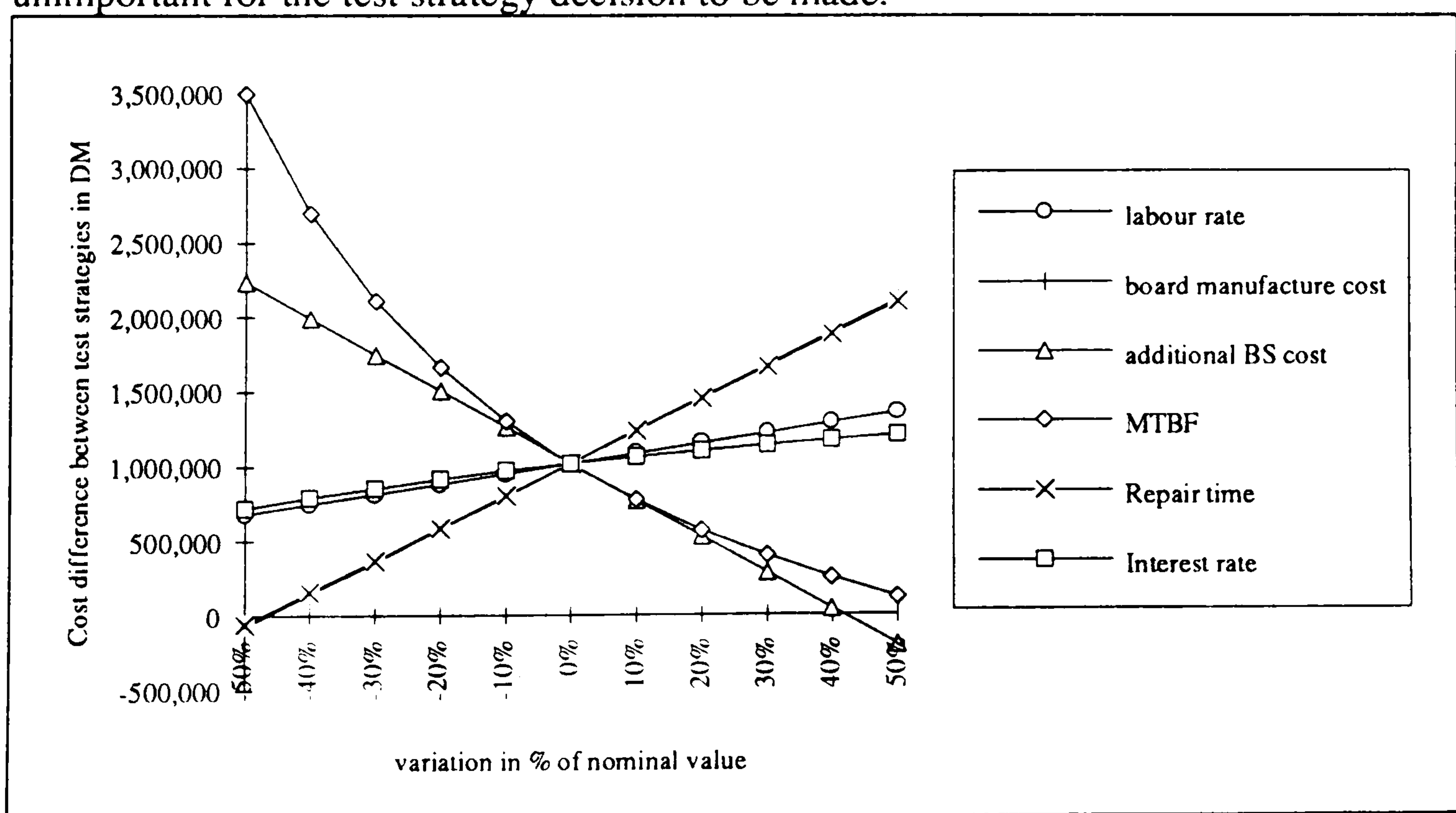


Figure 43: Sensitivity of the total cost difference in DM to the variation of parameter values in percent of the nominal value for In-circuit test minus boundary scan

8.3. Test Strategy Planning with ECOTEST

8.3.1. Test Strategy Planning for Selected Industrial Designs

The author has used ECOTEST for five industrial designs. By these applications of ECOTEST, he wanted to proof the applicability of the system in industrial environments. The five designs came from four different companies. One of the designs, the AM2909, is a standard IC from AMD, the ERCO design is a silicon compilation design (see [Bee90]), and the others are standard cell designs. The main economic parameters of the designs are listed in table 11.

Circuit Name	# equivalent gates	#blocks	production volume
SCR	42,772	10	10,000
ERCO	69,483	30	10,000
PRI	9,071	11	10,000
AMS	13,613	13	100,000
AM2909	254	8	1,000,000

Table 11: Main data of the industrial designs used for ECOTEST

The AM2909 is a micro sequencer from the 29-series of micro processors of AMD. It consists of one register, one RAM, three sequential blocks and three combinational blocks.

The ERCO is a macro design, where the macros are generated by a silicon compiler system. The design consists of two RAMs, five ROMs, two PLAs, five sequential random logic blocks, two combinational blocks, 13 registers and one ALU. The field of application of this design is consumer electronics.

The PRI is standard cell design with two RAMs, four sequential random logic blocks, three combinational blocks and two registers. The application field is telecommunication.

The SCR is a standard cell design with four RAMs, one ROM and five sequential random logic blocks. The application field of the chip is the cypherment of data in LANs.

The AMS is a standard cell design with 11 sequential random logic blocks and two RAMs. The field of application is numeric control.

For each of the designs we have performed automatic test strategy planning. Figures 44 through 48 present the total cost for each test strategy, in the same order as they are evaluated by ECOTEST.

The automatic test strategy planning algorithm, which was used here, is described in chapter 5. The first test strategies are related to making all testable units accessible. Therefore the total cost increases for these test strategies. The arrow points to the last test strategy, which is related to making the testable units (TUs) accessible. For this and all subsequent test strategies, the testable units are accessible.

For each TU, all appropriate internal test methods are applied, the total cost is evaluated, and the test method with the lowest cost is selected. For that reason, an increase in the total cost may occur for a subsequent test strategy during the evaluation of one TU. But at the end of the evaluation of a TU, the cost of the selected test strategy must be less or equal than the cost of all previous test strategies with full accessibility.

It is interesting to see, that for the AMS and the SCR the cost to make the TUs accessible is high. But for these circuits, the total costs are decreasing significantly with the selection of appropriate internal test methods for the first TUs which are evaluated. The reason for this effect is, that for these TUs self test methods are selected. By that the decrease of the cost is achieved by replacing the accessibility enhancing circuitry by self test circuitry, which reduces the test application cost and remains the gate count nearly unchanged. This type of test strategy decision, which is made by ECOTEST reflects the idea of using self test for TUs, for which the accessibility is pure.

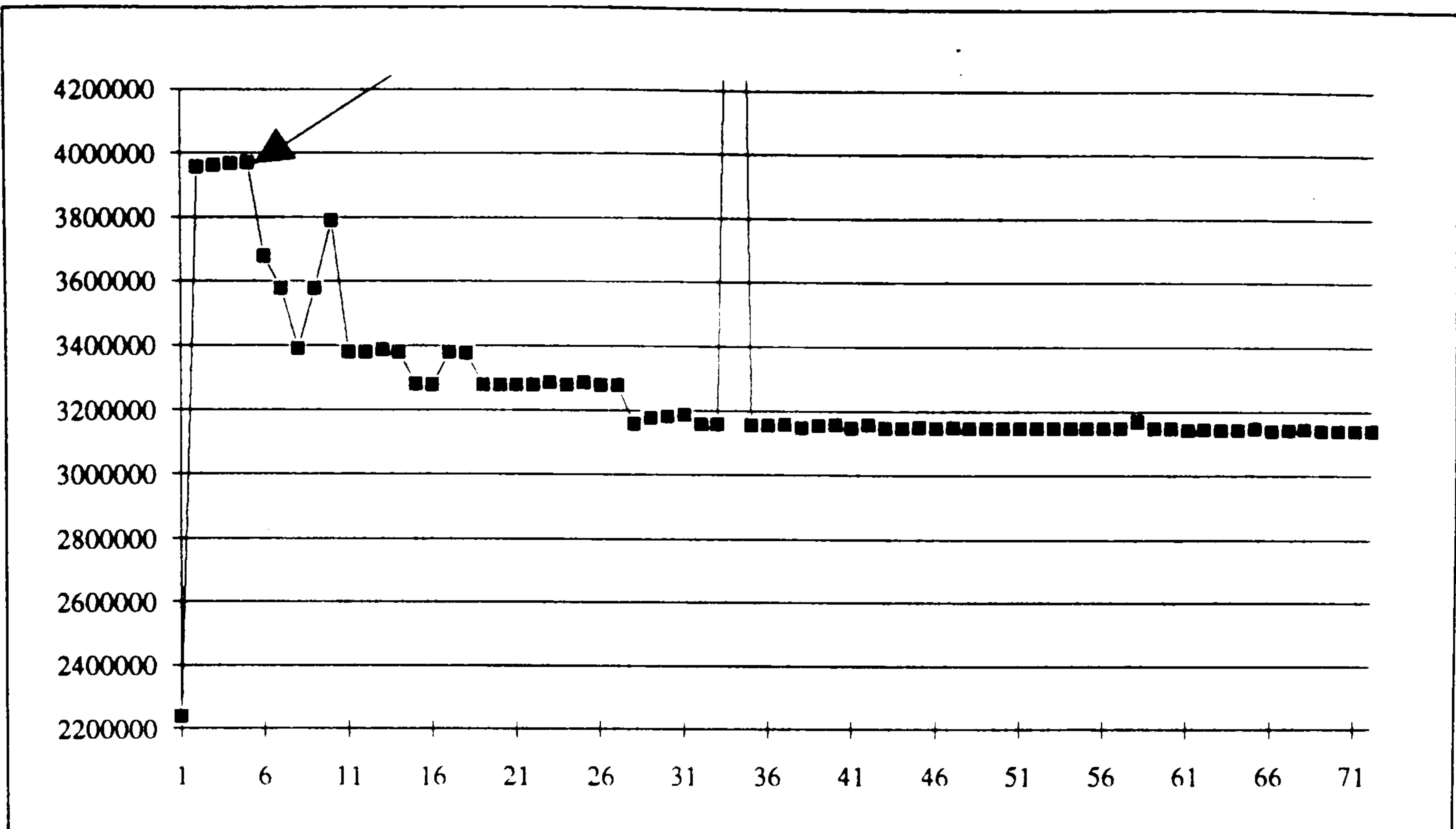


Figure 44: Cost of test strategies for automatic test strategy planning of ERCO

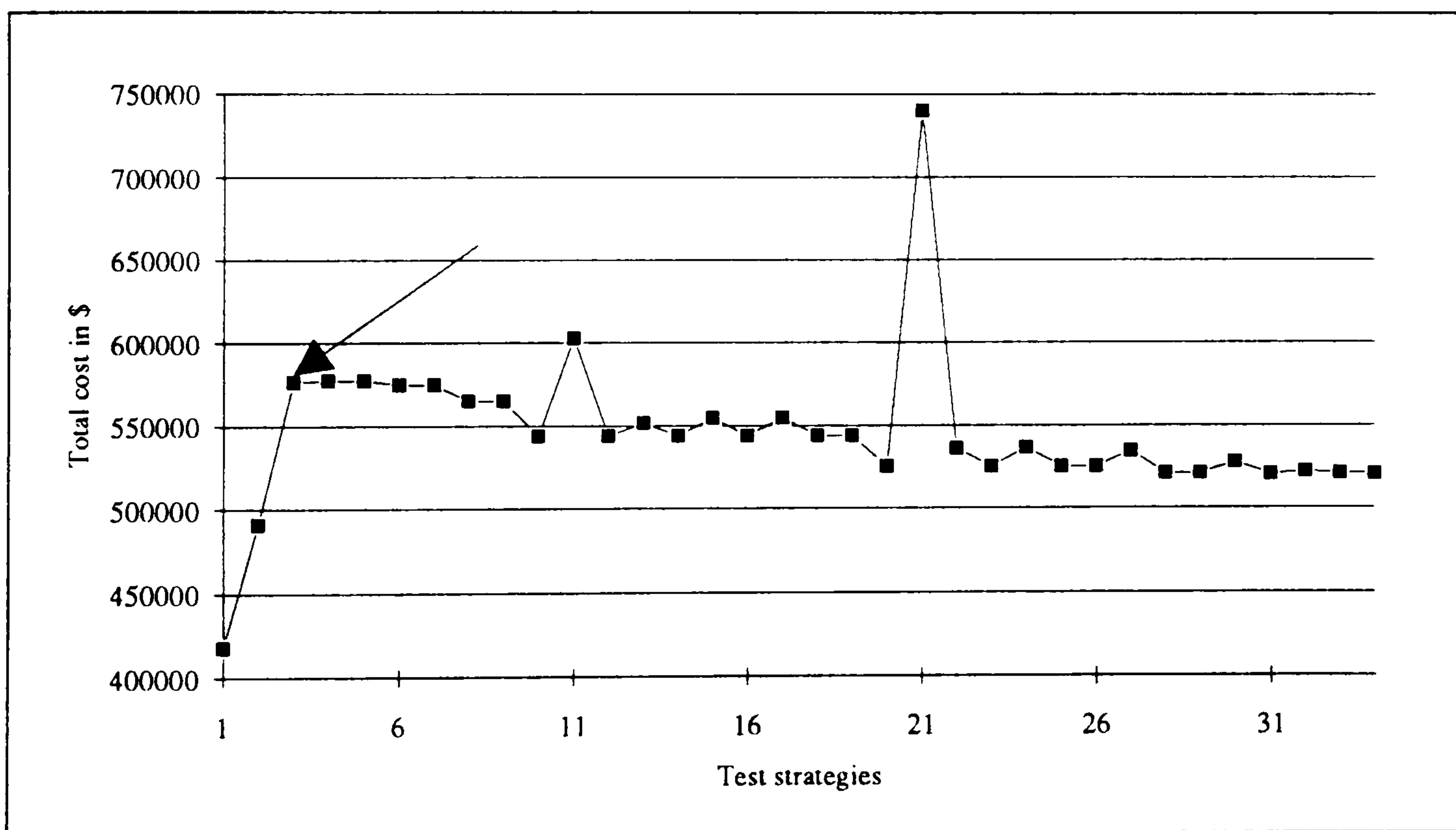


Figure 45: Cost of test strategies for automatic test strategy planning of PRI

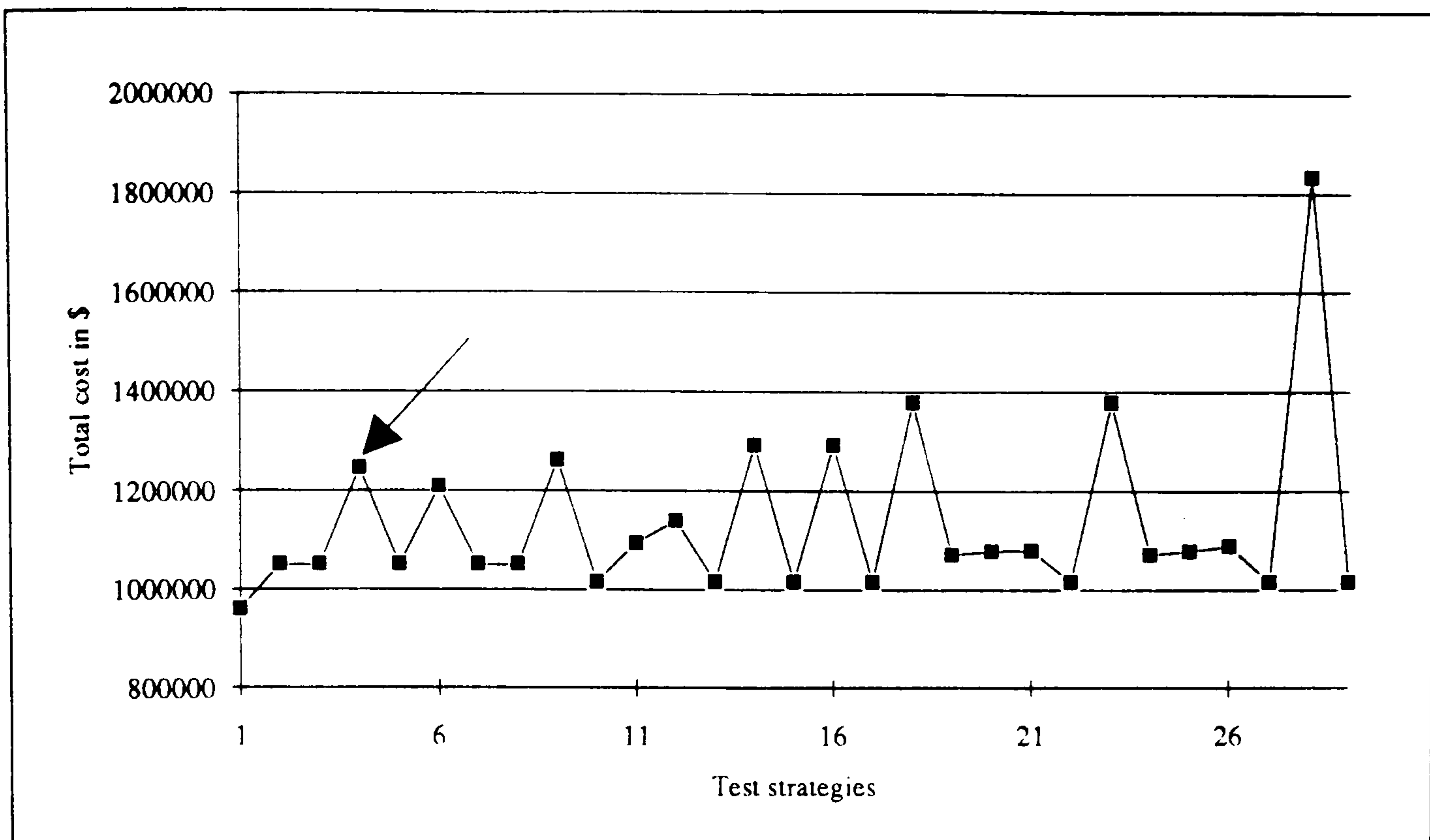


Figure 46: Cost of test strategies for automatic test strategy planning of AM2909

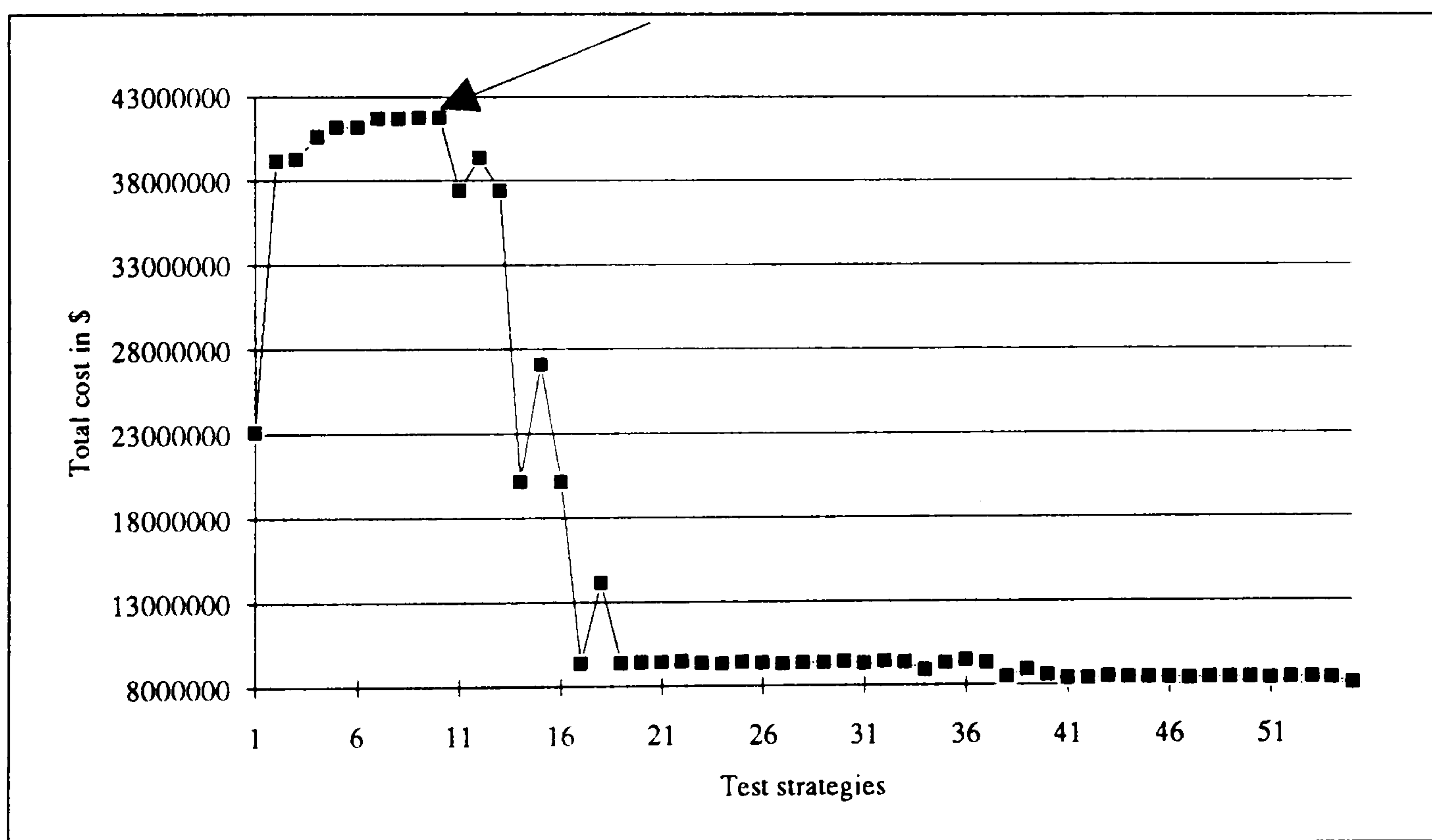


Figure 47: Cost of test strategies for automatic test strategy planning of AMS

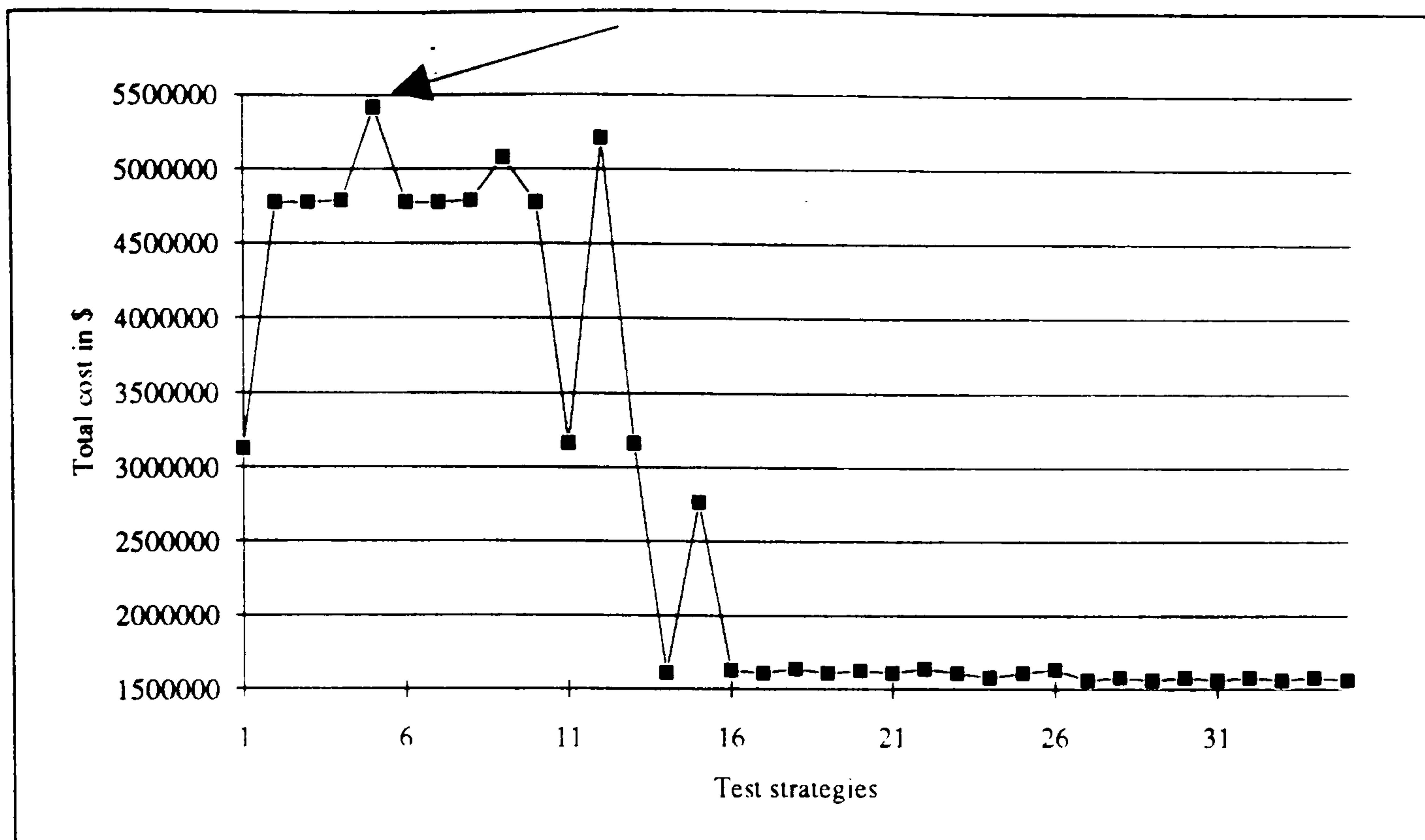


Figure 48: Cost of test strategies for automatic test strategy planning of SCR

Table 11 shows the run times for the automatic test strategy planning function and the savings in the total cost. The CPU times are measured on a HP-Apollo series 400 workstation. For the savings in the total cost, the author has set the final test strategy to the minimum cost and has compared it to the most expensive realistic test strategy, i.e. the most expensive test strategy, which may be selected by the designer without using ECOTEST. This selection does not include the peaks of the figures above. The numbers in the table show, that the savings can be significantly, in percentage of the total cost as well as in an absolute value.

Design	CPU time in sec	Minimum Cost in $	Maximum Cost in $	Savings in %
AM2909	1.8	1,015,200	1,377,725	36%
AMS	77	8,080,380	9,441,310	17%
ERCO	200	3,140,366	3,969,537	26%
PRI	12	520,166	602,841	16%
SCR	17	1,559,635	1,636,006	5%

Table 12: CPU times and cost savings by test strategy planning for industrial designs

The author has run the automatic test strategy planning procedure repeatedly, by using the *no reset* option for the second and subsequent runs. This means, that the starting test strategy is not a fixed initial one, but the optimised test strategy from the previous run. It was interesting to see, that for all but the SCR design this option lead to a different test

strategy. In some cases, a more economical test strategy was found (PRI, AMS), in some cases (ERCO, AMD) the test strategy resulting from not resetting the initial test strategy was more expensive. In all cases, this test strategy planning procedure ran into a loop, which means, that a resulting test strategy has been selected earlier. This matter of fact in shown in figure 49, where for all circuits the test strategies resulting from automatic test strategy planning are shown. The arrows point from the initial test strategy to the resulting test strategy. ITS stands for the initial test strategy, if the no reset option is not used. This initial test strategy applies to each TU the first test method in list of test methods, which is applicable, and which does not provide any accessibility. The list of test methods is ordered alphabetically.

The figure shows, that, for example, for the design AMS four different test strategies are generated, before the automatic test strategy planning process runs into a loop.

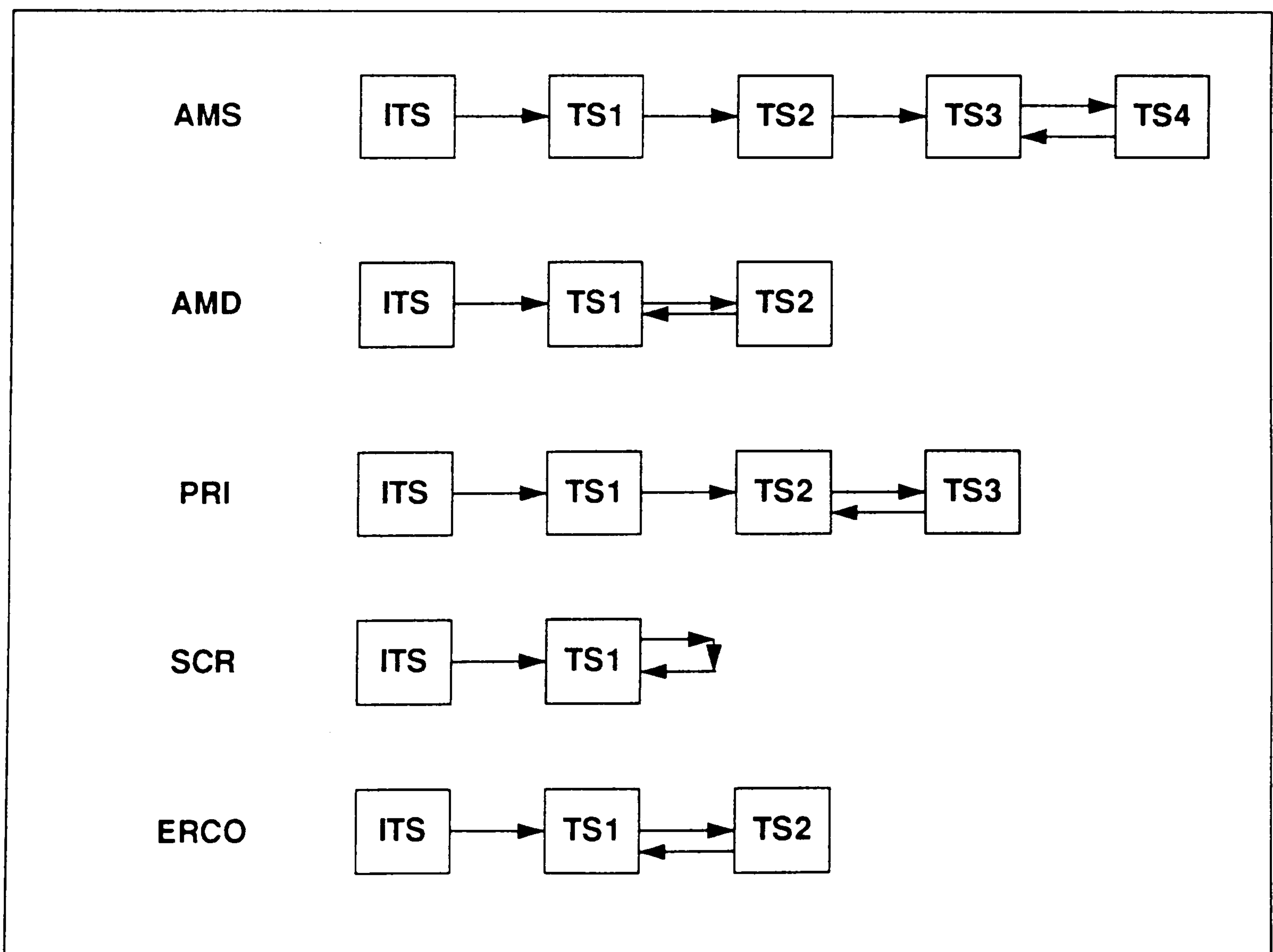


Figure 49: Automatic test strategy planning by using the previous automatic test strategy as the initial test strategy

Figures 50 and 51 show the costs related to the test strategies for the designs PRI and AMS. The difference between the most expensive and the most economical test strategy is 7% for the PRI and 12% for the AMS. This shows, that a further optimisation of the automatically generated test strategy by using previously generated automatic test strategies as initial test strategies, can produce further cost savings.

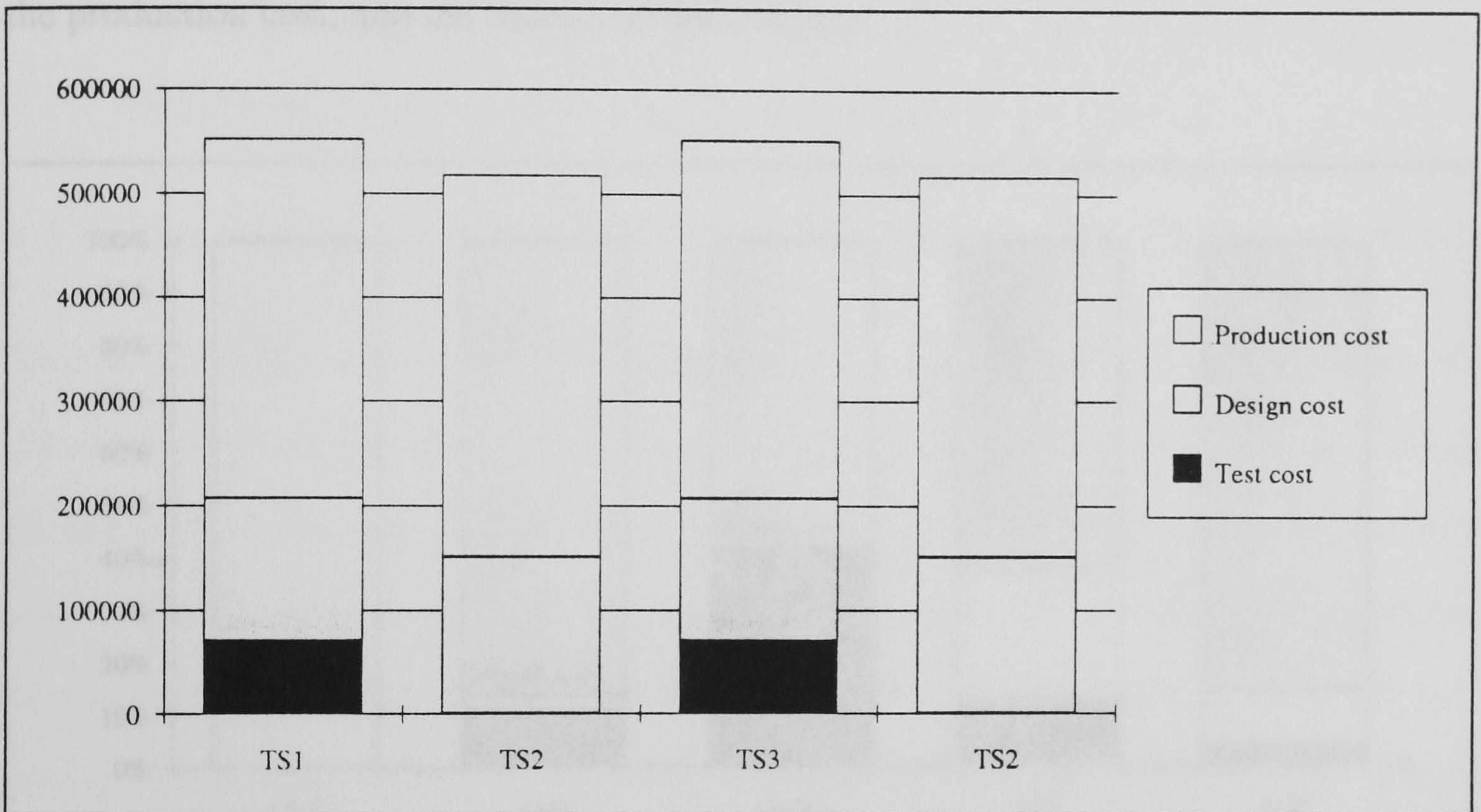


Figure 50: Cost of automatically generated test strategies with different initial test strategies for the design PRI

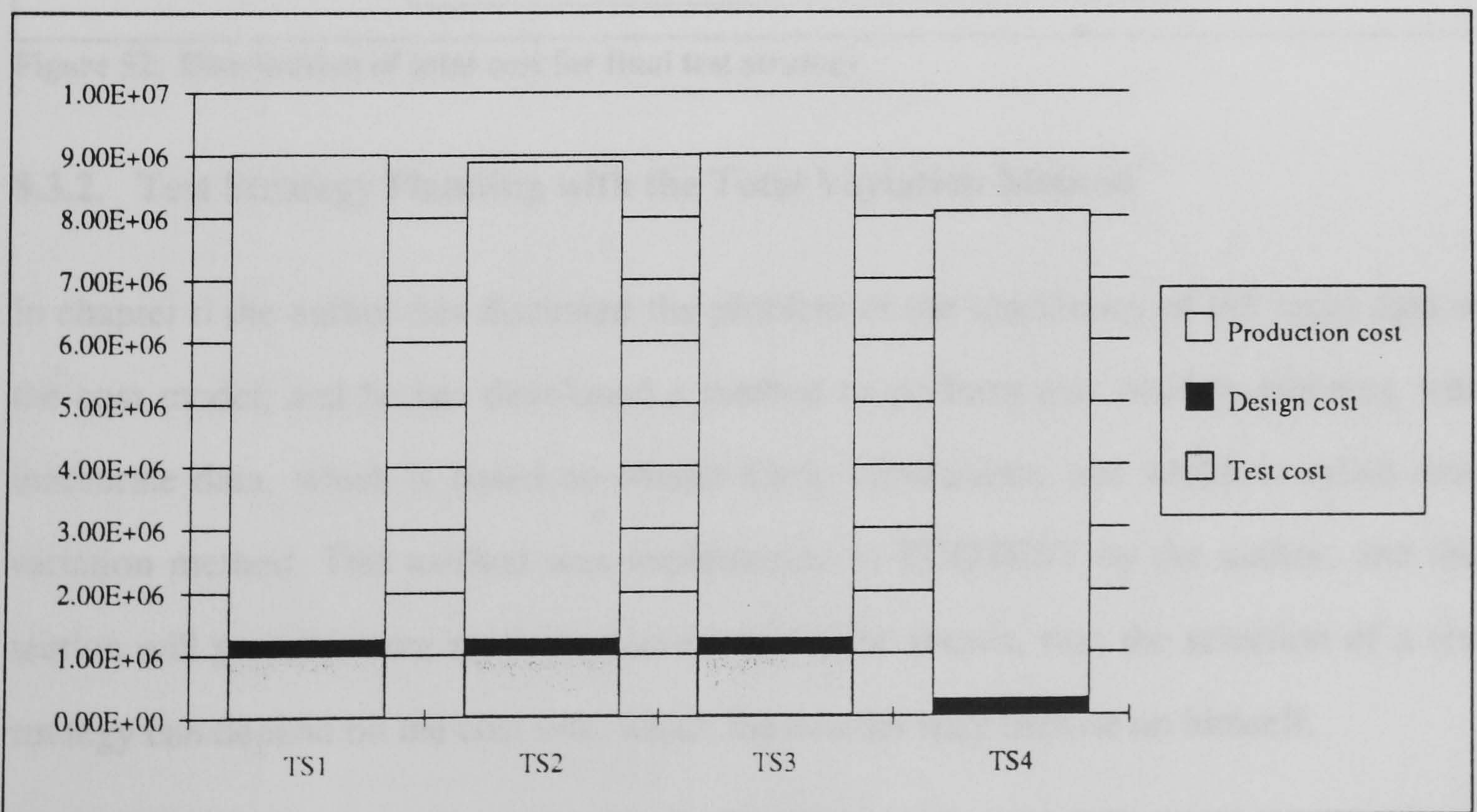


Figure 51: Cost of automatically generated test strategies with different initial test strategies for the design AMS

Figure 52 shows the distribution of the total cost between test, design and production. For all designs, the largest cost part is produced by the production. Test costs are important for the ERCO design, and the design costs are important for the PRI design. This shows, that the additional design effort needed for implementing DFT methods cannot be significant for these designs, because the basic design effort is makes only a small portion of the total cost. More important is the increase in the size, which impacts the production cost, and the impact on the test cost.

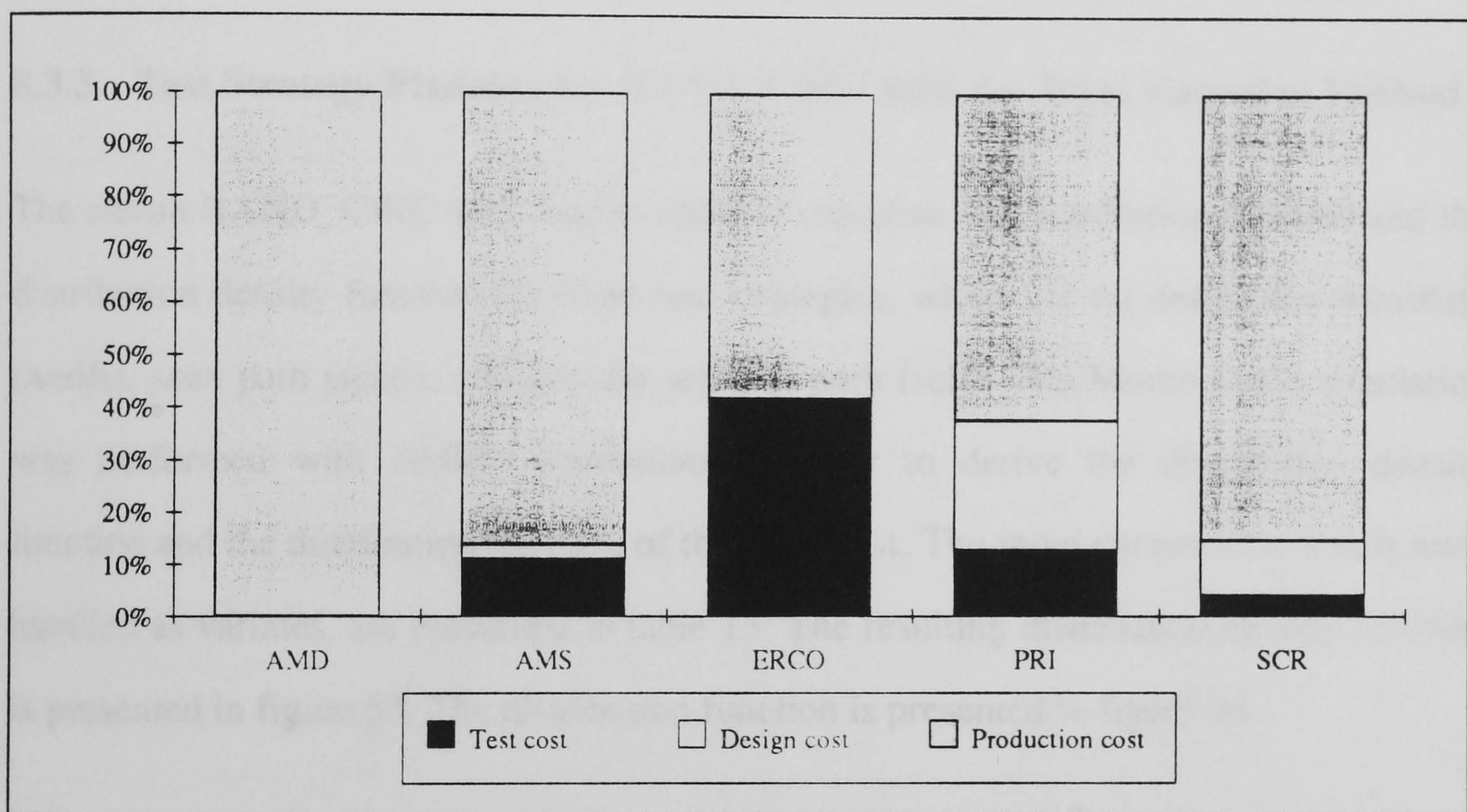


Figure 52: Distribution of total cost for final test strategy

8.3.2. Test Strategy Planning with the Total Variation Method

In chapter 6 the author has discussed the problem of the inaccuracy of the input data of the cost model, and he has developed a method to perform test strategy planning with inaccurate data, which is based on Monte Carlo simulations, and which is called total variation method. This method was implemented in ECOTEST by the author, and this section will present some results achieved. It will be shown, that the selection of a test strategy can depend on the cost risk, which the decider may impose on himself.

The experiments are performed with ECOTEST by using seven different designs:

- the first design, called RAND_CIRC, is a large homogeneous random logic design with a gate count of 50,000 gates.
- The second design, called DEMO_CIRC, is a heterogeneous design with four testable units, two random logic blocks with a complexity of 5,000 gates each, one 4K RAM, and one PLA with 50 product terms.
- The other designs are industrial circuits from several sources. The AM2909 is a standard IC, and the circuits ERCO, AMS, SCR and PRI are VLSI designs. They have been used and described already in the previous section.

8.3.3. Test Strategy Planning for RAND_CIRC with the Total Variation Method

The circuit RAND_CIRC was used in order to calculate the distribution function and the distribution density function for three test strategies, which are no design-for-testability (nodft), scan path (scan), and circular self test path (self). The Monte Carlo simulation was performed with 10,000 simulations in order to derive the distribution density function and the distribution function of the total cost. The input parameters, which were handled as variates, are presented in table 13. The resulting distribution density function is presented in figure 53. The distribution function is presented in figure 54.

Parameter Name	Abbreviation	Mean value	Standard deviation
Number of cells	cells	10,000	500
Number of gates	cgate	50,000	2,500
Performance complexity	cperf	1	0.05
CPU time	cputime	100	20
Designer's experience	exper	5	1
Average number of faults per gate	fpg	3	0.6
Manual test generation time per fault	mtgtime	0.5	0.1
Number of flip flops	dffs	2500	50
Productivity of the CAD system	pcad	1	0.1
Percentage of design time an external design centre is used	percuse	0.3	0.015
Sequential depth	seqdepth	8	4
Production volume	vol	10000	1000

Table 13: Description of normal distributed parameters

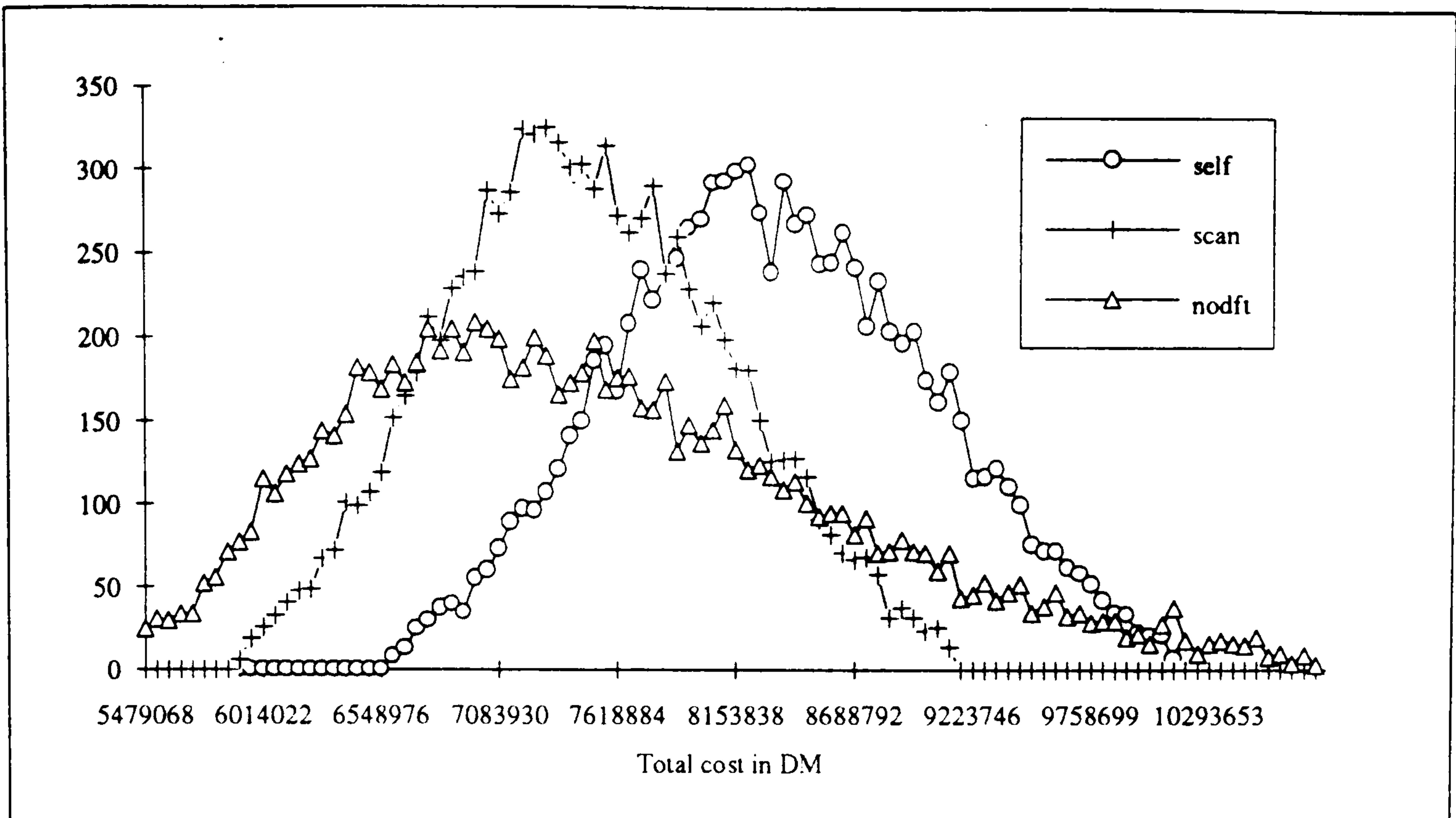


Figure 53: Distribution density function of total cost for RAND_CIRC

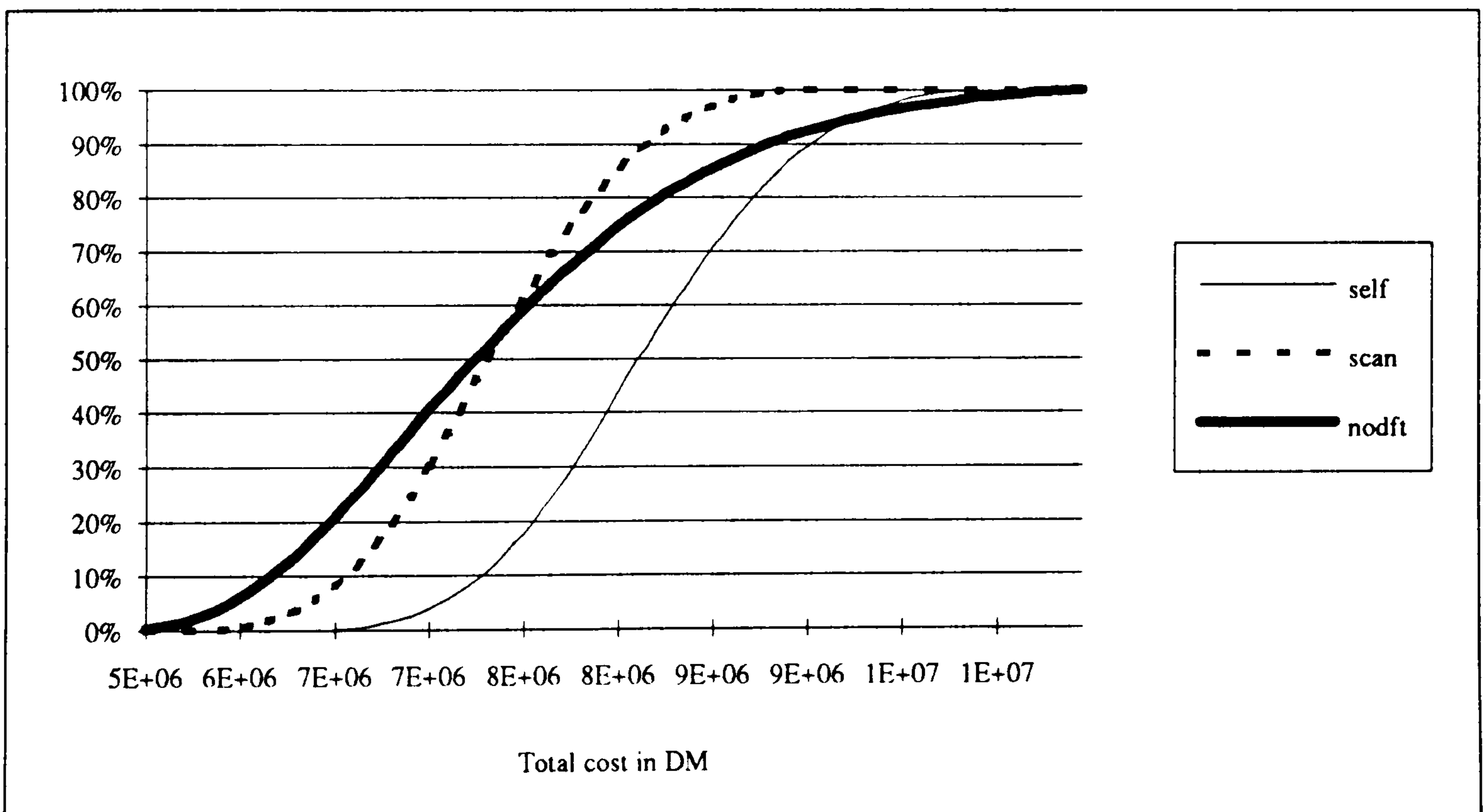


Figure 54: Distribution function of total cost for RAND_CIRC

The distribution density function shows, that the deviation of the total cost the nodft test strategy is larger than for the other test strategies. This is due to the large deviation of the sequential depth, which does only affect the nodft test strategy but not the scan- and the self test strategies.

By considering the inaccuracy for test strategy planning, the user has to define the cost limit probability, which is a measure of the risk of the decision he wants to take.

The cost limit probability *clp* is defined to *the probability, that the cost will be less than the related cost value.*

The related cost value is called *cost limit.* The cost limit is used as the optimisation criteria for automatic test strategy planning. The cost limit can be derived from the distribution function. For example, the cost limit for a clp of 30% is 6.7, 7.1 and 7.9 mio DM for the nodft-, scan- and self test strategy. This means, that the probability, that the total cost will be less than the cost limit, is 0.3. If this clp value is selected, the selected test strategy will be nodft. If the clp value is set to 70%, the cost limit is 8, 7.7 and 8.9 mio DM for the nodft-, scan- and self test strategy. In this case, the scan test strategy will be selected. A large clp may cause the selection of a test strategy, which is in average more expensive as another test strategy, but which has a low risk, that the total cost exceeds the cost limit.

We have used this sensitivity analysis method for various designs. The results in terms of different test strategies and run times will be presented and discussed in the following two sections.

8.3.4. Test Strategy Planning for DEMO_CIRC with Inaccurate Input Data

The design DEMO_CIRC was used to perform automatic test strategy planning with ECOTEST and with inaccurate input data. The input data, which are set to variates, are listed in table 14. They all have a normal distribution and a standard deviation of 10% of its mean value.

Parameter Name	Abbreviation	Deviation in % of mean value
Number of cells	cells	5%
Number of gates	cgate	5%
Performance complexity	cperf	5%
CPU time	cputime	20%
Designer's experience	exper	20%
Average number of faults per gate	fpg	20%
Manual test generation time per fault	mtgtime	20%
Number of flip flops	dffs	2%
Productivity of the CAD system	pcad	10%
Percentage of design time an external design centre is used	percuse	5%
Sequential depth	seqdepth	50%
Production volume	vol	10%
Fault coverage from verification	fcv	10%
Originality	or	10%
Number of test patterns from verification	tpver	10%

Table 14: Description of normal distributed parameters

The author has compared three different test strategies, which are listed in table 15.

Test Strategy Name	NODE1	NODE2	NODE3	NODE4
NODFT	treuer	no DFT	no DFT	illman
MIXDFT	treuer	no DFT	Scan Path	illman
SCANDFT	treuer	Scan Path	Scan Path	illman

Table 15: Test Strategies for DEMO_CIRC

The distribution function of the total cost is presented in figure 55.

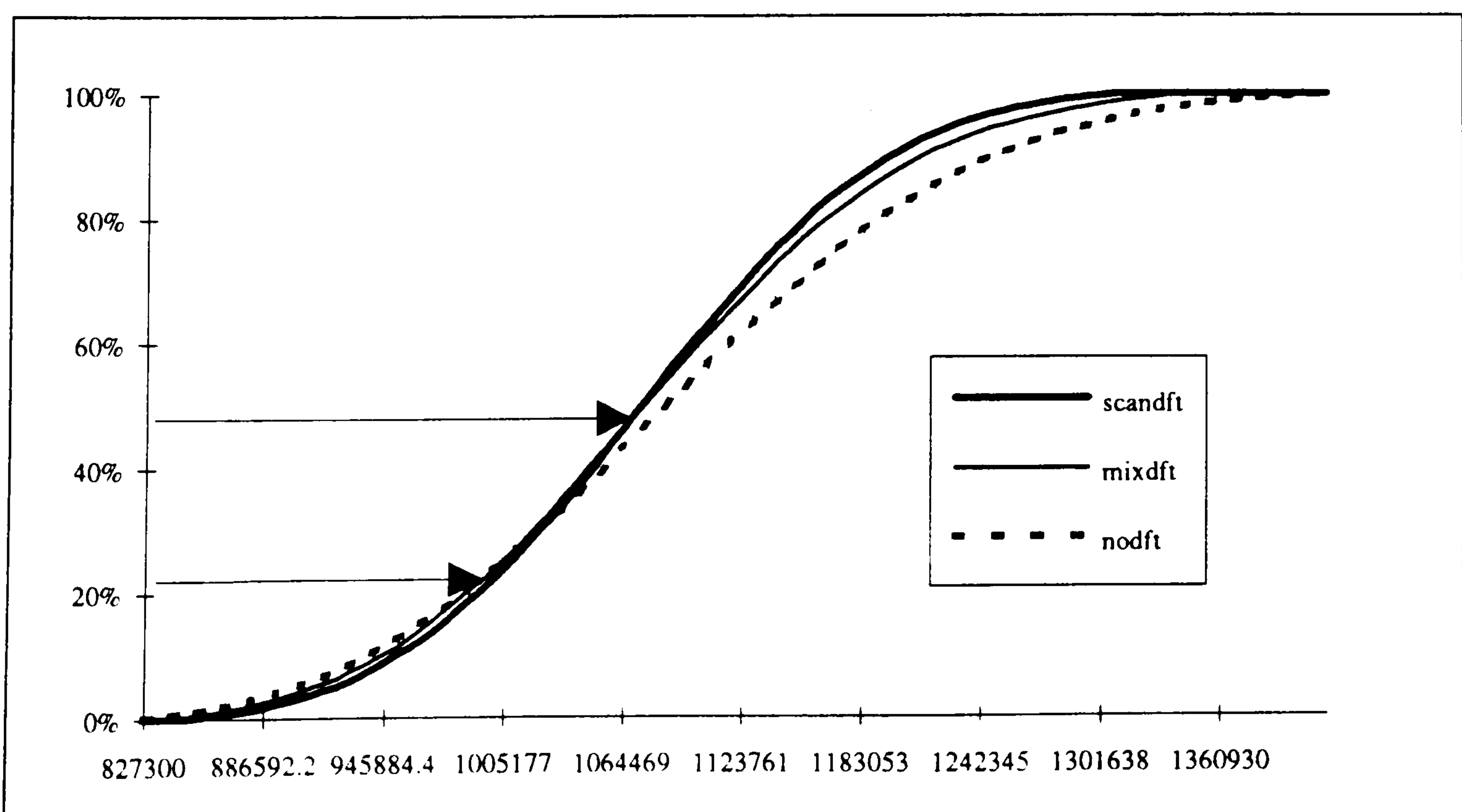


Figure 55: Distribution function of total cost for DEMO_CIRC

The two arrows point to the crossover points for the cost limit probability, where a different test strategy becomes the selected one. For clp values less than 23%, the nodft test strategy will be selected, for clp values between 23% and 48%, the mixdft test strategy will be selected, and for clp values of more than 48%, the scan test strategy will be selected.

8.3.5. Test Strategy Planning for Industrial Designs with the Total Variation Method

In order to proof the applicability of these methods, we have used the total variation method for industrial designs. This section will present the run times in comparison to the run times without Monte Carlo simulation. The deviation of the parameters was set to a percentage of the mean value as defined in table 14. The CPU times were measured on a HP-Apollo workstation series 400. Table 16 presents the CPU times and the relation of the CPU times without and with total variation. The number of Monte Carlo simulations is set to 200, which was found to be appropriate in chapter 6. The maximum possible relation between the CPU time with and without total variation is 200. The reason, why for all designs the actual value is smaller, is more or less significant run time for the accessibility calculation, which is performed only once. If the accessibility calculation run time is large compared to the cost model calculation run time, the relation factor is small, and vice versa. The accessibility calculation run time is large, if many transfer paths are defined, or if the testable units consist of many IOs. Altogether, the run times with total variation are less than 1.5 hours in all cases, which is acceptable for this type of non-interactive application.

Design	CPU time in sec with total variation	CPU time in sec without total variation	Relation of CPU times
ERCO	4062	200	20
PRI	1036	12	86
AMS	2758	77	36
AM2909	238	1.8	132
SCR	876	17	52

Table 16: Run times for industrial designs

Automatic test strategy planning was performed with the clp values of 50% and 95%. The resulting test strategy was different for the ERCO design in one testable unit, for the SCR in two testable units, for PRI in one testable unit and in two testable units for the AMS. For the AM2909, the resulting test strategies were the same.

8.4. Test Strategy Planning with ECOvbs

This section provides results on test strategies by using ECOvbs. The analysis was done on a large, double sided board containing surface mounted components, including a number of VLSI. The board is a typical complex computer board. The main data about the board are presented in table 17.

Parameter	Value
Number of VLSIs	40
Number of RAMs	28
Number of resistors	800
Number of capacitors	100
Number of analog components	100
Production volume	5,000
Production yield	50%
Number of solder joints	15,000
Total design effort	353 weeks

Table 17: Main design data of computer board

The data are typical industrial values. Costs are calculated in ECU. Most of the costing data, such as the component prices, are test strategy dependent, and therefore they are not listed here. Appendix D provides a full description of the board data.

An In-circuit test could not be applied to the board, because it is double sided, and the pin grid is too small. Therefore the alternative test strategies relate to functional test

versus boundary scan test, using a manufacturing defect analyser for a prescreen test, and performing an incoming inspection to the components. The four test strategies which are evaluated are listed in table 18.

TS1	pre screen test, functional test, system test
TS2	boundary scan test, system test
TS3	incoming test, prescreen test, functional test, system test
TS4	board cluster test, system test

Table 18: Test strategies for the computer board

Test strategy TS1 comprises of a pre screening test which is performed by a manufacturing defect analyser, a functional board test and a system test.

For test strategy TS2 the board and its digital components include boundary scan. The board is tested by a boundary scan test on board level and system test on system level, which benefits from the boundary scan for the fault diagnosis.

In the third test strategy, TS3, an incoming test is applied to the digital components, the active analog components, which are the deskew elements, and the bare board. On board level, a pre screen test and a functional test are applied, and on system level a conventional system test is applied.

The forth test strategy, TS4, uses test clusters in order to partition the board test into three parts:

- The RAMs are implemented with boundary scan and a self test. They comprise on test cluster, which is self testable.
- The VLSIs are implemented with boundary scan. They are tested as one test cluster by using the boundary scan.
- The rest of the board is tested by a classical functional board test.

In addition to the cluster tests, a system test is applied on system level.

Figure 56 shows the total cost per test strategy, and figure 57 shows the remaining faults

after the final test. The remaining faults range from 0.0024 to 0.0032 faults per board, which relates to a quality levels from 99.74% to 99.68%, which is acceptable. The most economical test strategy is the boundary scan test strategy (TS") with an a quality level of 99.75%. The functional test strategies (TS1, TS3) have about the same cost (99.304 versus 99.282 mio ECU), but the incoming test used in TS3 leads to a higher quality level (99.76% versus 99.75%). Test strategy TS4 is more expensive than TS2 (98.176 mio ECU) and leads to the lowest quality level (99.68%).

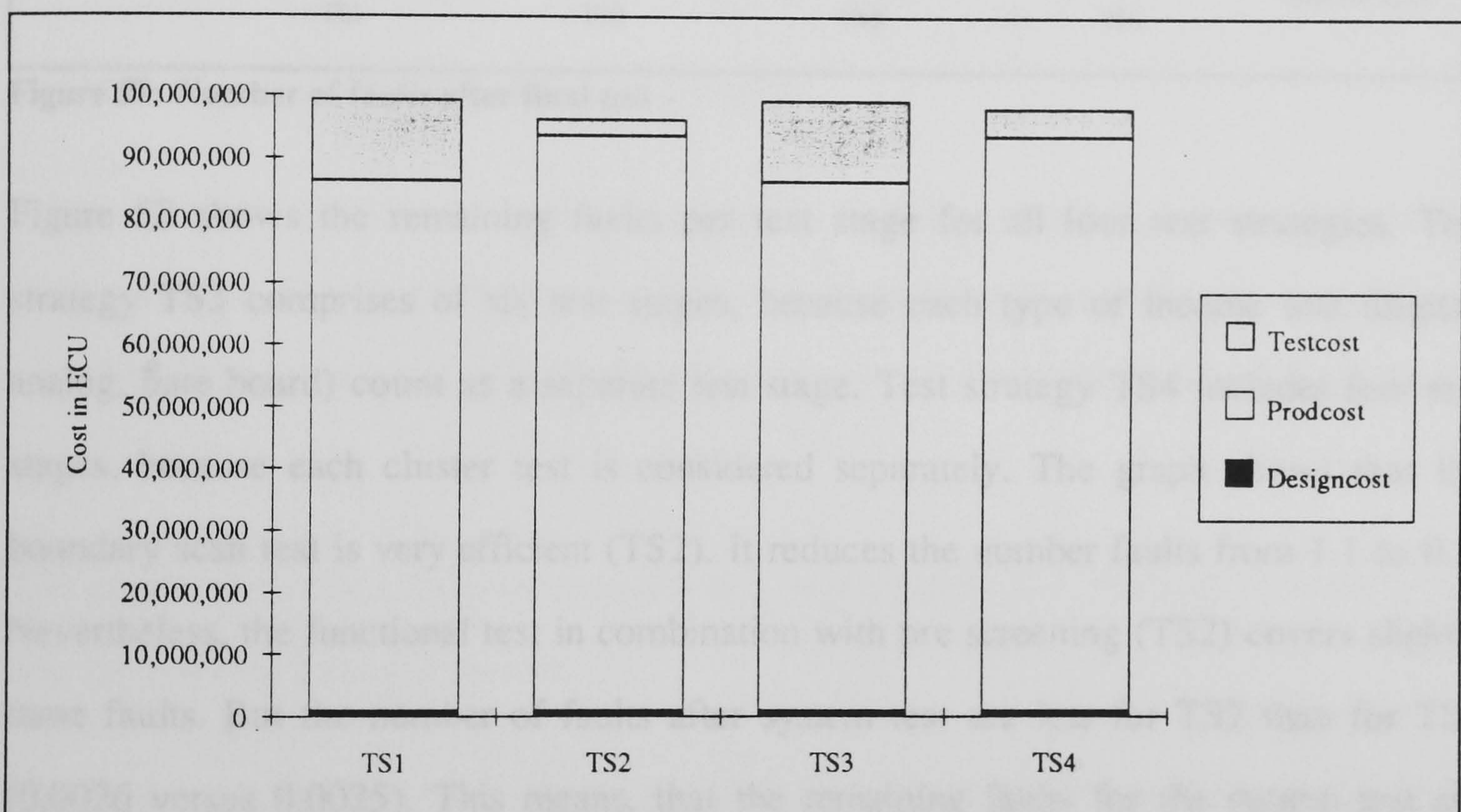


Figure 56: Cost of board test strategies

The remaining faults are mainly static faults for test strategies TS1 and TS2, which are based on functional testing. The dynamic faults make a significant portion for the boundary scan and test cluster strategies, because the fault coverage for dynamic faults is lower for boundary scan than for functional board test. For test strategy TS4, a significant portion of the remaining faults are resistor faults, because they are not detected during the cluster test. The fault coverage for capacitor faults is zero for all test but the system test, where the fault coverage is 5%. Therefore, the capacitor faults make a significant fraction of the remaining faults for all test strategies.

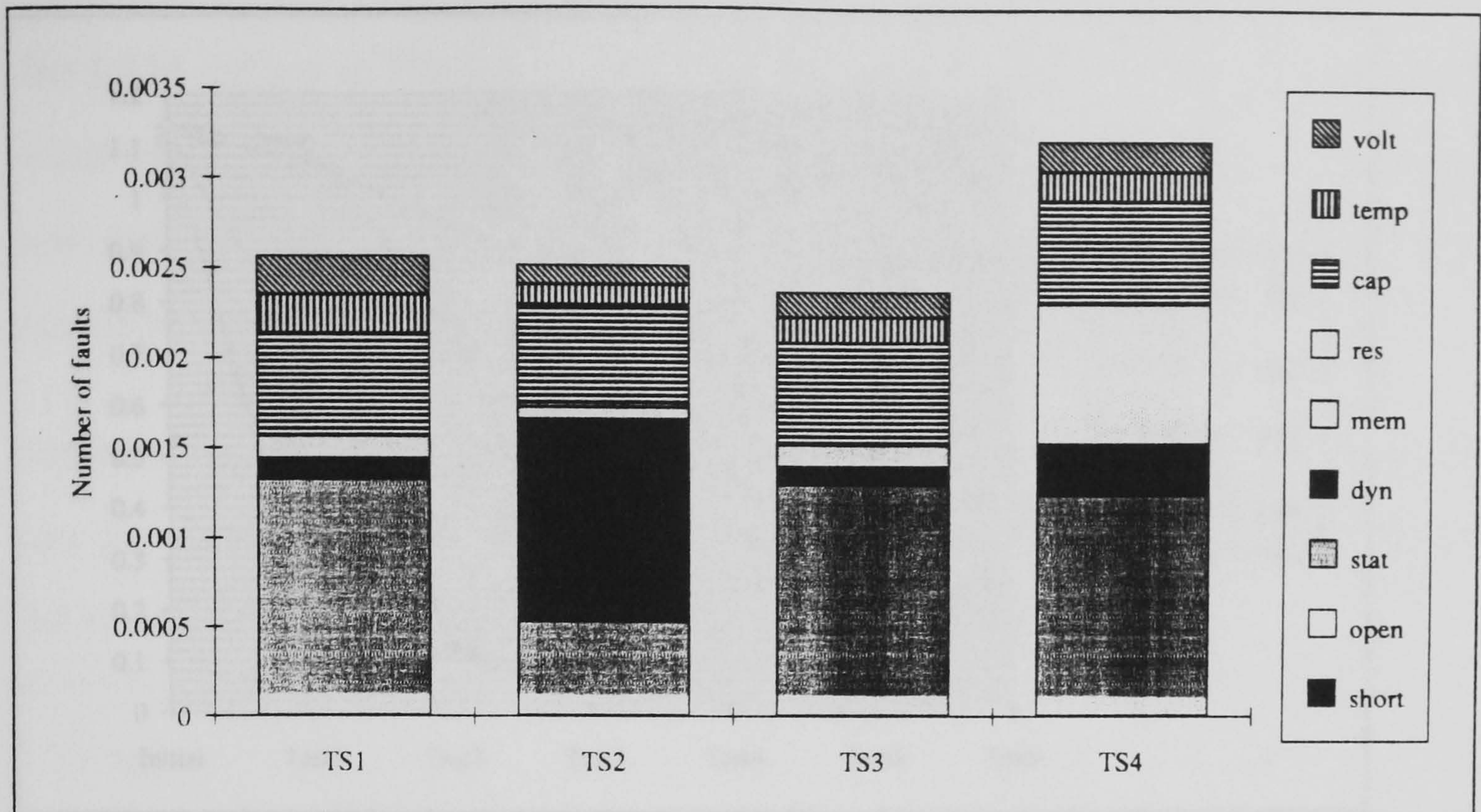


Figure 57: Number of faults after final test

Figure 58 shows the remaining faults per test stage for all four test strategies. Test strategy TS3 comprises of six test stages, because each type of income test (digital, analog, bare board) count as a separate test stage. Test strategy TS4 includes four test stages, because each cluster test is considered separately. The graph shows that the boundary scan test is very efficient (TS2). It reduces the number faults from 1.1 to 0.2. Nevertheless, the functional test in combination with pre screening (TS2) covers slightly more faults. But the number of faults after system test are less for TS2 than for TS1 (0.0026 versus 0.0025). This means, that the remaining faults for the system test are better covered for the fault spectrum of the boundary scan test.

The incoming test reduces the number of faults on board level by 0.25 faults to 0.87 faults. But this gain is reduced to 0.02 faults after the pre screen test and the functional test. But this leads to a reduction of the system test cost from 3.541 mio ECU to 3.250 mio ECU, due to less faults to be detected and located through an expensive fault diagnosis at system level test.

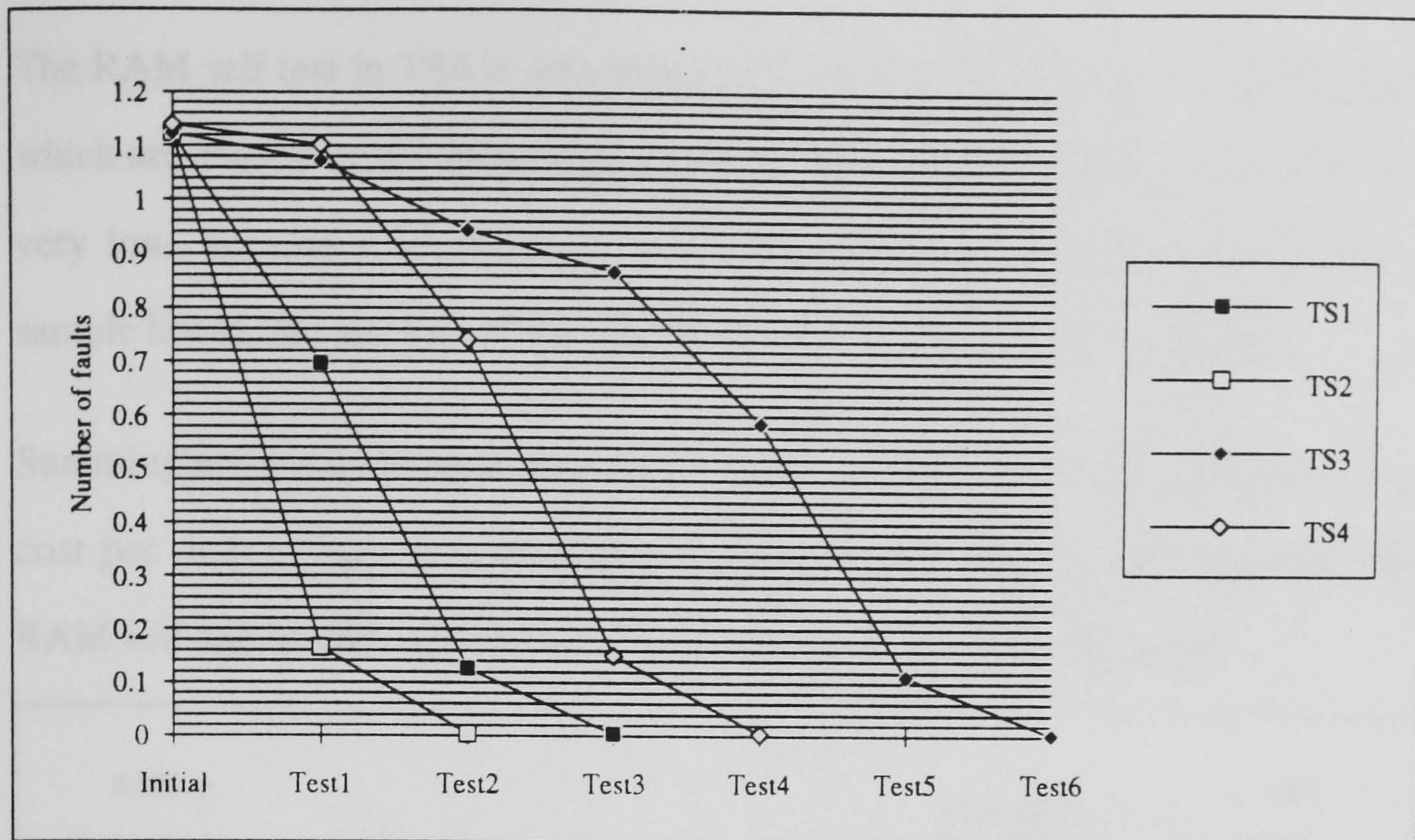


Figure 58: Number of remaining faults per test stage

Figures 59 through 62 show per test stage the related cost per detected fault and the fault coverage. The cost per detected fault include DFT cost as well as test application cost. The fault coverage is calculated by the relation of the total faults which are detected by the test.

The graphs show, that the prescreen test is a very efficient test, because it covers about one third of the faults at low cost.

The functional test becomes very economical, if it is not applied to the complex VLSI components, which is the case for the cluster tests. In all cases the fault coverage of the functional test is more than 80% at reasonable cost per detected fault. But the boundary scan test in TS2 achieves a higher fault coverage at about half the cost.

System test is in all test strategies very expensive. But it is needed, because it detects the "hard faults", i.e. the faults which are hard to detect. Nevertheless a test strategy should always avoid the slippage of faults to the system test, due to its high fault diagnosis cost. The system test cost per fault for TS2 is much lower than for all other test strategies. This is due to the boundary scan, which is used for fault diagnosis, and which reduces the diagnosis cost. Additional DFT cost is not included, because it is related to the board level test.

The RAM self test in TS4 is very expensive, due to the high additional production cost, which are caused by the expensive, self testable RAMs. In addition, the fault coverage is very low, because it is limited to the faults which are related to the RAMs. For this sample board, the self test of the RAMs is not an economical test strategy.

Summing up, it may be said, that a test stage, for which the fault coverage is low, but the cost per fault is high, is a candidate to be removed. This is definitely the case for the RAM self test in TS4, and eventually for the digital income test in TS3.

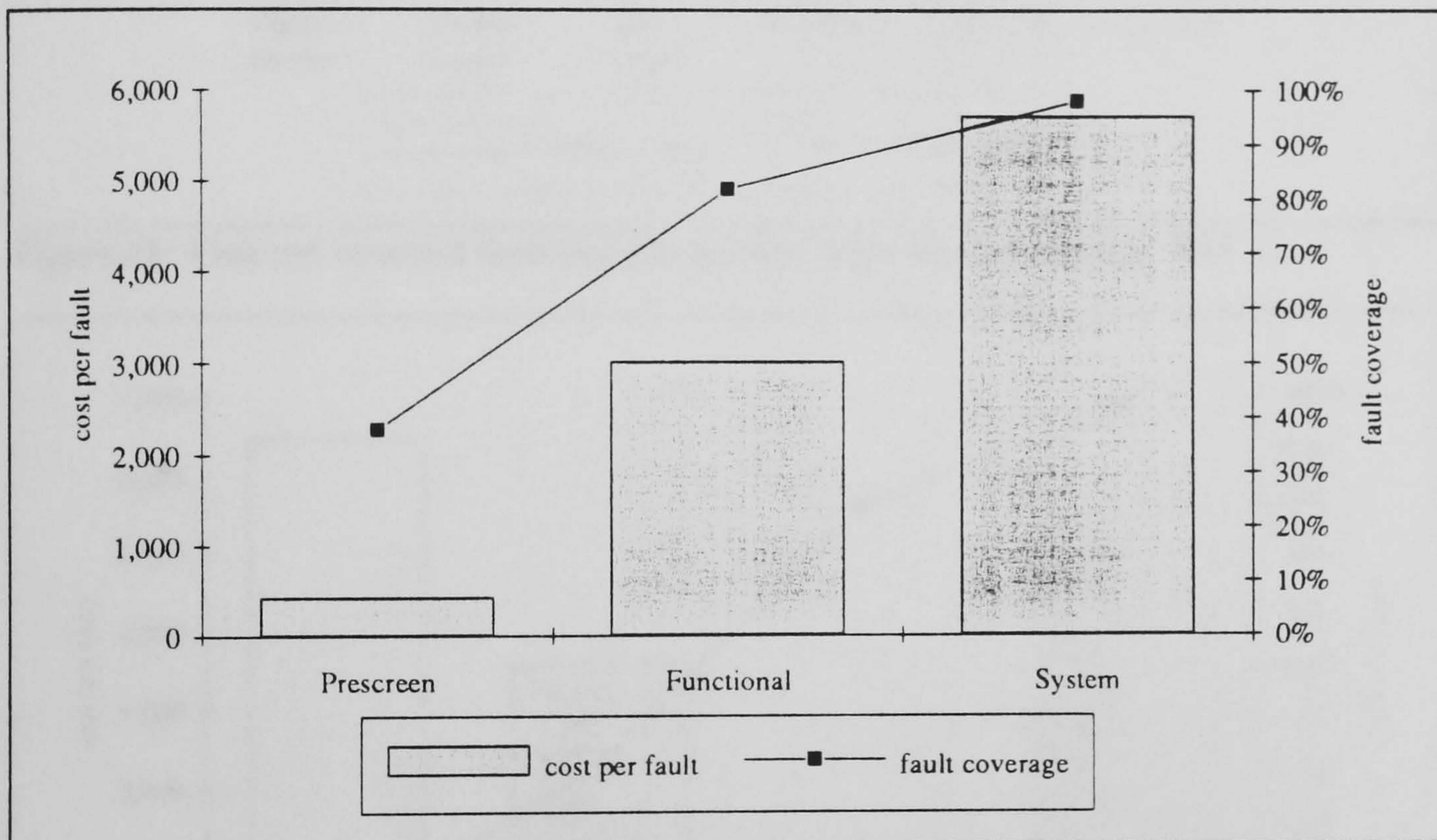


Figure 59: Cost per detected fault in ECU per test stage for test strategy TS1

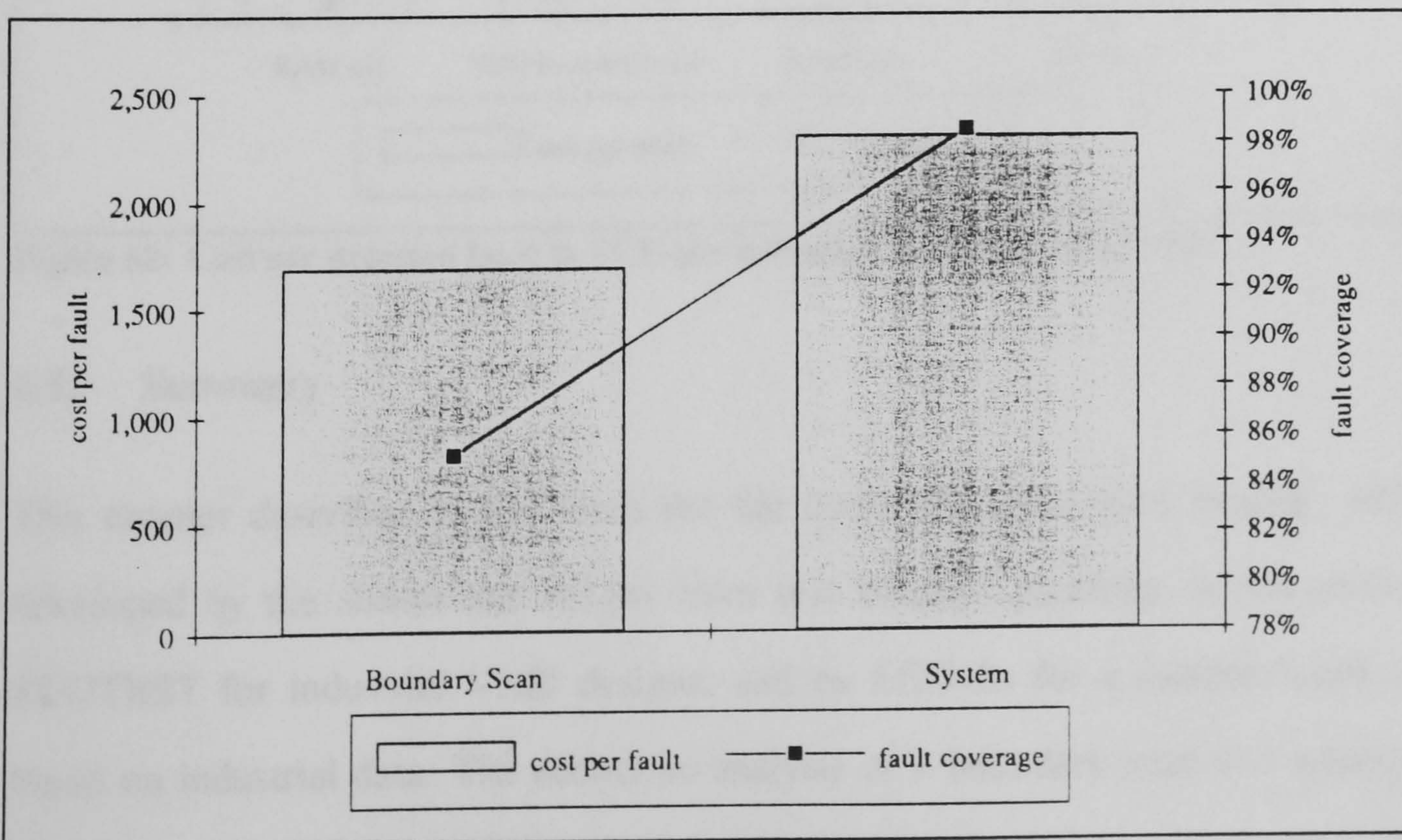


Figure 60: Cost per detected fault in ECU per test stage for test strategy TS2

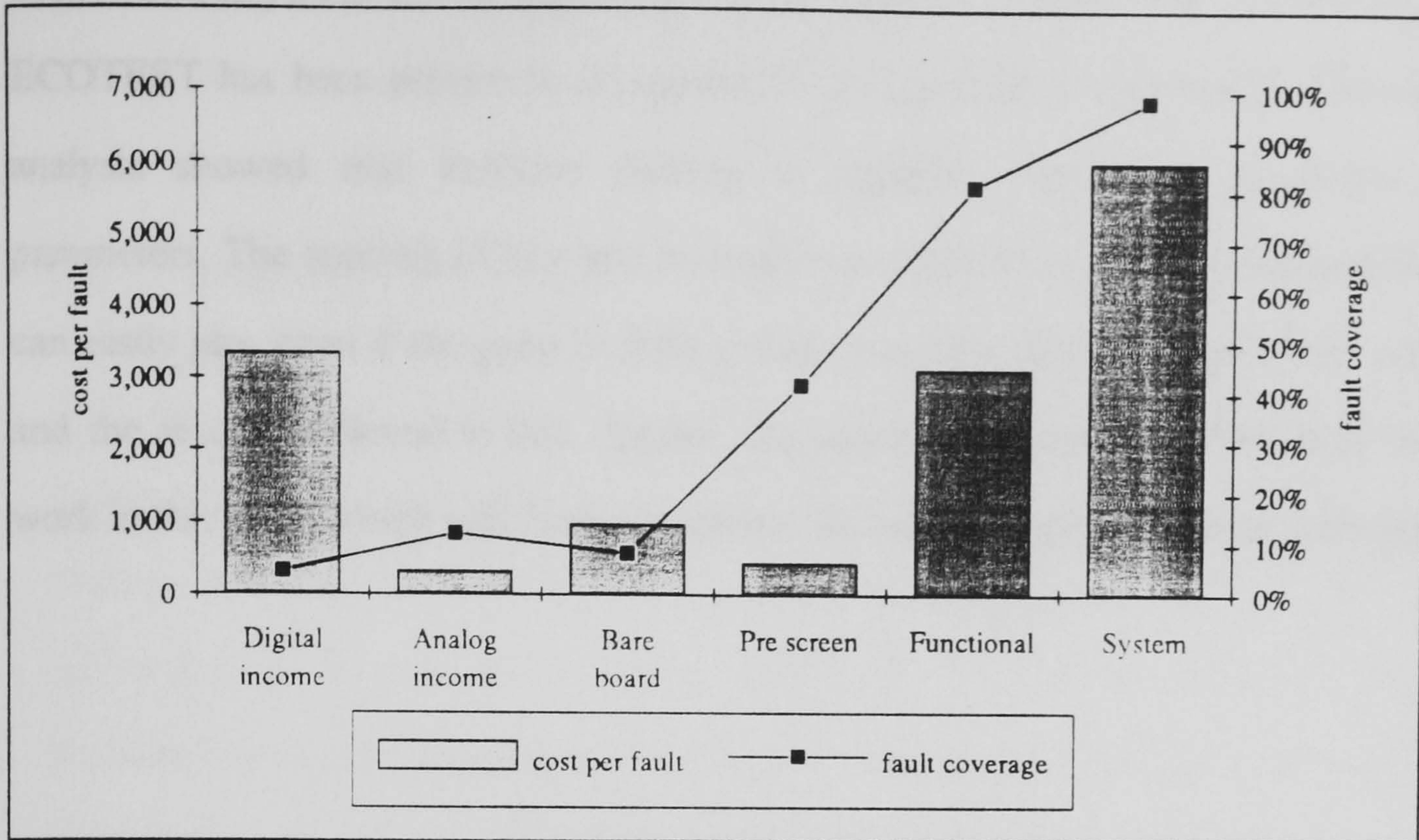


Figure 61: Cost per detected fault in ECU per test stage for test strategy TS3

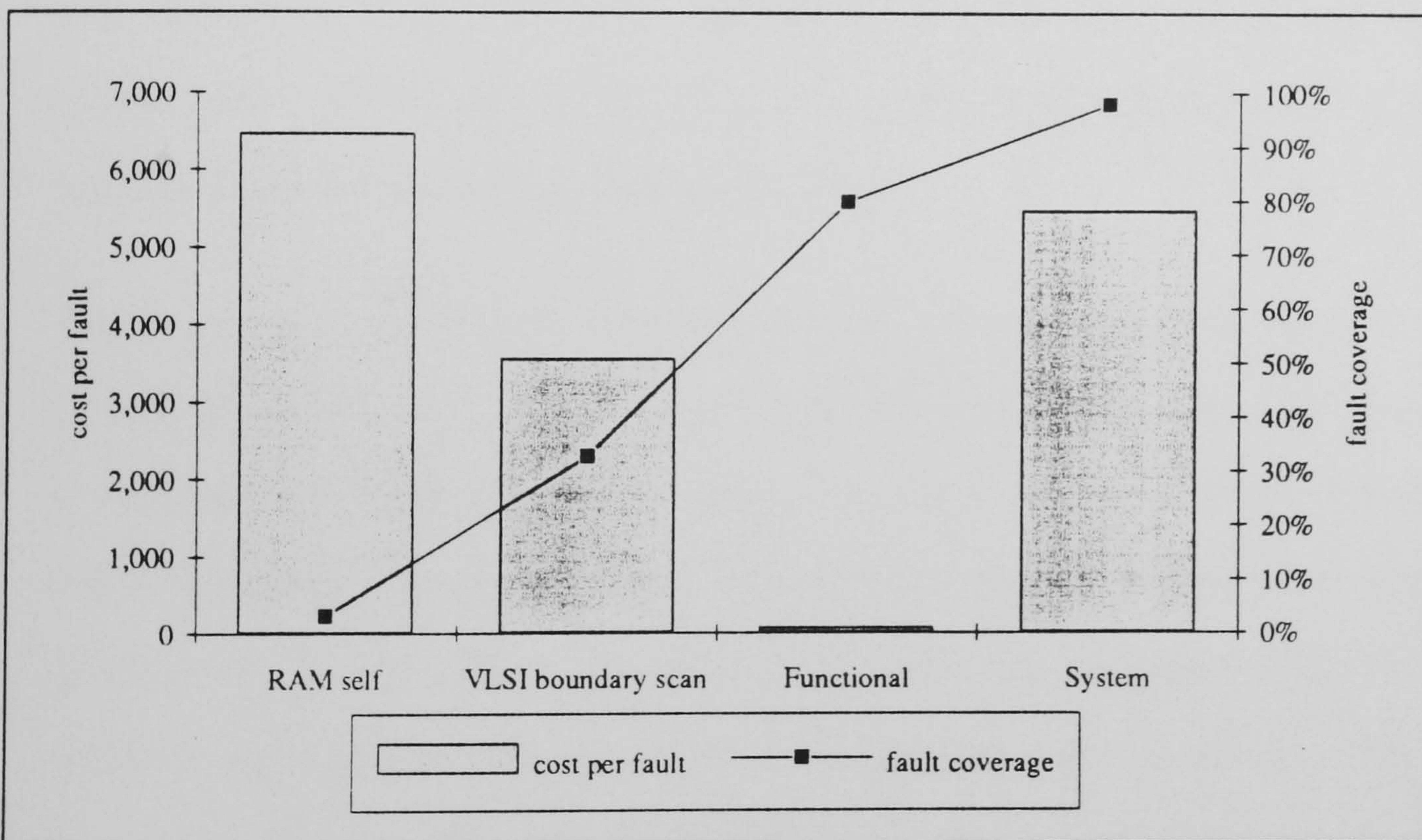


Figure 62: Cost per detected fault in ECU per test stage for test strategy TS4

8.5. Summary

This chapter described results from the life cycle test economics models, which were developed by the author and results from test strategy planning, which performed by ECOTEST for industrial VLSI designs, and by ECOvbs for a sample board, which is based on industrial data. The economic analysis of a boundary scan test strategy versus an In-circuit test strategy, showed, that the consideration of the whole life cycle for an

economic analysis of test strategies may be an important aspect. The test strategy planner ECOTEST has been proven to be applicable in industrial environments. The sensitivity analysis showed that decision making is possible even with inaccurate costing parameters. The analysis of four test strategies with ECOvbs showed, that high DFT cost can easily pay, even if the gains in field test are not considered. Based on the experience and the results achieved in this chapter, the author will present some ideas for future work in this field, which will further improve the tools for optimising test strategies.

Chapter 9

Conclusions

9.1. Summary of the Work

This thesis described a novel approach for creating test strategies for complex, VLSI based systems. It presented previous work in this field, which is mainly related to test strategy planning for ICs and not boards or entire systems. The author has discussed the advantages of his approach over previous systems. The test strategy planning system described is based on life cycle economic evaluations, which defines a test strategy for the life cycle of the entire system instead of limiting it to a certain integration level. The evaluation of the economics of test strategies is based on predictions, which are subject to inaccuracy. A method for evaluating and handling this inaccuracy was developed and integrated into the test strategy planning system.

The test strategy planning system comprises of two software tools, which can be linked to each other, and which can be used by different groups of persons. ECOTEST is a test strategy planner for heterogeneous ASICs, which enables to create test strategies for the cell based ASICs. The test strategies can be generated automatically, semi automatically, or interactively. The system can generate different test strategies which are based on different requirements, such as different achievable fault coverages. The economic parameters of these different test strategies can then be used in ECOvbs, the test strategy planner for VLSI based systems. The IC level test strategies are used as part of the life cycle test strategies, and the related economics during the life cycle of the entire system can be evaluated. In the same way, a board level test strategy is integrated into the life cycle test strategy.

Both test strategy planners are based on a test economics model and on stored knowledge about the impact of test methods on the economics parameters (the test method descriptions). The test economics model and the test method descriptions can be

altered by the user of the system. The system is implemented by an object oriented approach in C++.

9.2. Conclusions

The results, which are presented in chapter 8 and the usage of the system in industrial environments by users other than the developers of the system showed that the approaches and the implementation are satisfactory.

The sensitivity analysis methods removed a problem, which has led to concerns about the test economics approach by several test experts. By handling the costing parameters as variates, the inaccuracy of predicting values for the primary parameters as well as the inaccuracy of the test economics model itself is taken into account by determining its impact on the inaccuracy of the total cost. The decision criterion of minimising the total cost is then replaced by minimising the worst case of the total cost in a user defined percentage of cases. This method allows to select between a certain probability to get very low cost by a high risk to get very high cost and a high probability to get moderate cost with a low risk to get very high cost.

Another concern about the test economics modelling approach for test strategy planning was addressed to the cost of data gathering. This problem has been partly solved by refining the complex test economics model, which was developed at Brunel University ([Var84], [Dis92]), by removing some parameters not necessarily important in industrial uses. In addition, the sensitivity analysis method can also be used for reducing the cost for data gathering. The related approach, which was developed by the author, is called *iterative sensitivity analysis*. This method allows to start test strategy planning with a rough but cheap estimate of the primary parameters. The outcome of the sensitivity analysis is then used to analyse, which parameters are more and which are less sensitive. Based on this result, a more accurate parameter prediction can be performed, but only for the sensitive parameters.

The partitioning of the system into two separate tools allows the usage by two different group of persons. This matches to the organisation of most electronic companies. A quality assurance department is responsible for test strategy planning for systems, whereas the ASIC designers are responsible for the IC level test strategy. Nevertheless, the quality assurance people get more and more interested in design-for-testability strategies, but their knowledge about these methods and its impact on the economics is often poor. This partitioning of the test strategy planning allows it to be used by the designers for generating alternative IC test strategies. The economic results can then be used by the quality assurance people in order to evaluate the economics of incorporating the DFT strategies into a test strategy for the entire system.

The test strategy planning system was used in different industrial environments, and it was applied to different industrial designs. The feedback coming from the industrial users, which came from different companies in different countries with a different business (T.I.D. in Spain, Philips in the Netherlands, Bull in France, Siemens, Telefunken and SNI in Germany), showed, that the test strategy planning system is a useful tool for solving the problem of test strategy planning. The discussions with the users showed the big interest in the methods of economics based decision making and the way this method is implemented in the test strategy planning system. These discussions and the study of the future needs in the field of design and production of complex electronic systems have led to the following ideas, which could extend the approach of economics based test strategy planning, and which could further improve the system.

9.3. Future Research

The test strategy planner ECOTEST is based on a fixed hierarchy of the design. The testable units are defined by the hierarchy of the design. The partitioning of the design into testable units is not always straightforward, and different partitioning leads to different test strategies with different economics. To evaluate the different partitioning of the design with ECOTEST, the design must be repartitioned by the user. This can cause

a significant effort, especially if many different partitioning should be evaluated. Also, this method does not allow for automatic generation of the optimum partitioning. For these reasons, an extension of the system, which allows an online partitioning of the design, either by the user or automatically by the system, could lead to a further improvement of ECOTEST.

Test strategy planning in ECOvbs is not performed automatically. This is not a critical weakness, because test strategies can be set up by the user very quickly, and the system allows for parallel evaluation of alternative test strategies. But an automatic evaluation of test strategies by evaluating all possible combinations, or a user defined subset of it, can be an improvement of the system for cases where the number of test strategy alternatives is very large, or if there are economical alternatives, which are not taken into account by the user for certain reasons.

Other enhancements of the system are mainly related to integrating it into other environments. ECOTEST in its current version is a pure advisory tool. The implementation of the test strategy, especially the implementation of the DFT techniques is not linked to ECOTEST. Today's design methodology includes more and more logic synthesis which allows to describe the design on a higher level than gate level. A description of DFT techniques on this higher level could be formulated independent of the design. This DFT synthesis description can be part of the test method descriptions of ECOTEST, which allows to link ECOTEST to logic synthesis tools. The DFT part of the generated test strategy is then implemented automatically by a push-button link from ECOTEST to the synthesis tool.

The test strategy planner ECOvbs takes into account many production data, which may be available in a computer integrated manufacturing data base (CIM). By integrating ECOvbs into such a CIM environment, many data could be accessed directly from its source. This reduces the cost of data gathering and makes the provision of data more secure, especially concerning data updates and data entering errors.

The methods of economics driven test strategy planning, which have been developed by the author, are very general. They can easily be adopted for other applications of engineering decision making. In the introduction it was shown that test strategy decisions are tightly linked to design and manufacture decisions. This fact can be used in integrating other decision making processes, such as technology decisions, or manufacturing strategies, into the economics based test strategy planning system. This would extend the test strategy planning system in an engineering decision making system, which would provide a real concurrent engineering tool.

The complex problem of test strategy planning for VLSI based systems has mainly been solved by the approach which has been described in this thesis. Using this system will help to avoid the development of untestable and therefore non sellable electronic systems. The system helps in particular to minimise the expenditures which are needed to assure a high quality of the systems which are shipped to the customers. This work has attracted significant attention by industry. This attention has made the author optimistic, that the approach will be used extensively in the future, and that other applications for this approach, as discussed in the previous section, will arise soon.

References

[Aba89] Abadir, M.:
TIGER: Testability Insertion Guidance Expert System
Proc. IEEE International Conference on CAD, 1989

[Abr84] Abramovici, M., Menon, P.R., Miller, D.T.:
Critical Path Tracing : AN Alternative to Fault Simulation.
IEEE Design & Test of Computers, 1/1984

[Aga90] Agarwal, V.:
Comment on the presentation "Planning Testable VLSI Design under Economic Aspects"
Workshop "Testmethoden und Zuverlässigkeit von Schaltungen und Systemen", 1990

[Agr82] Agrawal, V.D., Mercer, M.R.:
Testability Measures: What Do They Tell Us?
Proc. ITC 1982

[And80] Ando, H.:
Testing VLSI with Random Access Scan.
Dig. COMPCON 1980

[Ant82] Antreich, K.J., Koblitz, R.K.:
Design Centering by Yield Prediction
IEEE Transactions on Circuits and Systems, vol. CAS-29, no. 2, February 1982, pp. 88-95

[Ant87] Antreich, K.J., Schulz, M.H.:
Accelerated Fault Simulation and Fault Grading in Combinational Circuits.
IEEE Transactions on CAD, vol CAD-6, Sept. 1987

[Ard87] Ardeman, A.:
ASICs: Costing and Planning
New Electronics, 20 January 1987

[Arm72] Armstrong, D.B.:
A Deductive Method for Simulating Faults in Logic Circuits.
IEEE Transactions on Computers, 21/1972

[Arm82] Armaos, J.
Zur Optimierung elektrischer Schaltungen unter Berücksichtigung der Parametertoleranzen.
PhD thesis, Institute for Computer Aided Circuit Design, Technical University of Munich, 1982

[Bar92] Bartesch, V.:
CAE - Verfahren zur Auswahl und Bewertung von Selbsttestmethoden
Diploma thesis at Fachhochschule Muenchen, Fachbereich Elektrotechnik, 1992

[Bee89] Beenker, F., Dekker, R.:
General Introduction to the Philips Macro-Test Concepts.
EVEREST report, Doc. No. PHI 0062 TN 01 IN WP5

[Bee90] Beenker, F., Dekker, R., Stans, R., Van der Star, M.:
Implementing Macro Test in Silicon Compiler Design
IEEE Design&Test of Computers, April 1990, pp. 41-51

[Bel61] Bellman, R.:
Adaptive control processes: A guided tour
Princeton University Press, 1961

[Ben80] Bennetts, R.G., Maunders, C.M., Robinson, G.D.:
CAMELOT: A Computer-Aided Measure for Logic Testability.
Proc. ICCD 1980

[Ben84] Bennetts, R.G.:
Design of Testable Logic Circuits.
Addison-Wesley Publishing Company, 1984

[Ben87] Bennetts, R.G.:

Evaluating the cost of using an ATE - a structured approach

Computer-Aided Engineering Journal, August 1987

[Ben89] Bennetts, R.G.:

Definition of basic terms used in testing, and a list of abbreviations used in WP5.

ESPRIT - EVEREST report no. BEN0026 TN 01 IN W5

[Bla78] Blanchard, B.S.:

Design and Manage to Life Cycle Cost

M/A Press, Portland, 1978

[Blo88] Blohm, H., Lueder, K.:

Investition

Verlag Vahlen, Muenchen 1988, 6th edition

[Brg84] Brglez, F.:

On Testability Analysis of Combinational Networks.

Proc. ISCAS 1984

[Brg89] Brglez, F.; Bryan, D.; Kozminski, K.:

Combinational Profiles of Sequential Benchmark Circuits.

Proceedings of Int. Symposium on Circuits nad Systems, 1989

[Che89] Cheng, K. T., Agrawal. V. D.:

Design of Sequential Machines for Efficient Test Generation

Proc. IEEE International Conference on Computer-Aided Design, 1989, pp 358-361

[Dae81] Daehn. W., Mucha, J.:

A Hardware Approach to Logic Testing of Large Programmable Logic Arrays.

IEEE Trans. on Computers, vol. C-30, Nov 1981, pp 829-833

[Das82] Das Gupta, S., Goel, P., Walther, R.G., Williams, T.W.:
A Variation of LSSD and Its Implication on Design and Test Pattern Generation in VLSI.
Proc. ITC 1982

[Dav82] Davis, B.:
The Economics of Automatic Testing
McGraw-Hill, 1982

[Dav92] Davis, B.:
The economics of design and test
in Proc. Economics of Design and Test for electronic circuits and systems, Ellis Horwood Limited, Chichester, 1992

[Dea89] Dear, I.D.D.:
Test Strategy Planning Methodology for the Functional Testing of Integrated Circuits that is Driven by Economic Parameters.
PhD thesis, Brunel University, 1989

[Dic87] Dick, J.:
Entwicklung und Implementierung eines Programmes zum Erkennen redundanter Fehler in kombinatorischen Schaltungen.
Master thesis at Technische Universität München, Lehrstuhl für Rechnergestützes Entwerfen, 1987

[DiG89] Di Giacomo, J.:
VLSI Handbook.
McGraw Hill publishing Company, 1989

[Din69] Dinkelbach, W.:
Sensitivitätsanalysen und parametrische Programmierung
Springer Verlag Berlin Heidelberg New York, 1969

[Din82] Dinkelbach, W:
Entscheidungsmodelle
Walter de Gruyter Verlag Berlin New York, 1982

[Dis89] Dislis, C., Dear, I.D., Miles, J.R., Ambler, A.P.:
Cost Analysis of Test Method Environments
Proc. IEEE Intenational Test Conference, 1989

[Dis91] Dislis, C., Dear, I.D., Ambler, A.P.:
An Economics Based Test Strategy Planner for VLSI Design
Proc. European Test Conference, 1991

[Dis92] Dislis, C.:
A Financially Based Automated Advisor for Design for Test Strategy Generation.
PhD thesis, Department of Electrical Engineering and Electronics, Brunel University, 1992.

[Eic77] Eichelberger, E.B., Williams, T.W.:
A Logic Design Structure for LSI Testability.
Proc. Design Automation Conference 1977

[Eic80] Eichelberger, E.B. and Lindbloom, E.:
A Heuristic Test Pattern Generator for Programmable Logic Arrays.
IBM J.Research and Development, vol.24, Jan. 1980, pp 15-22

[Fuj83] Fujiwara, H., Shimono, T.:
On the Acceleration of Test Generation Algorithms.
IEEE Transactions on Computers, 32/1983

[Gho89] Ghosh, A., Devadas, S., Newton, A.R.:
Test Generation for Highly Sequential Circuits.
Proc. ICCAD 1989

[Goe80] Goel, P.:
Test Generation Costs Analysis and Projections.
Proc. Design Automation Conference, 1980

[Goe81] Goel, P.:

An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits.

IEEE Transactions on Computers,30/1981

[Gol80] Goldstein, L.H., Thipgen, E.L.:

SCOAP: Sandia Controllability/Observability Analysis Program.

Proc. DAC 1980

[Gra79] Grason, J.:

TMEAS - A Testability Measurement Program.

Proc. Design Automation Conference 1979

[Gun90] Gundlach, H.H.S., Mueller-Glaser, K.D.:

On Automatic Test Point Insertion in Sequential Circuits

Proc. IEEE International Test Conference, 1990, pp. 1072-1079

[Ham65] Hammersley, J.M., Handscomb, D.C.:

Monte Carlo Methods

Methuen & Co. Ltd., London, 1965

[Hey89] Heymbeeck, L.:

A Literature Survey of Functional/Behavioral Testing.

EVEREST report DOC. No. PHI 0066 SS TN

[Hil85] Hills, T.G., Davis, M.J., Rogers, W.E.:

Cost Visibility Through Life Cycle Cost Boards.

Logistics Spectrum, vol. 19, Los Angeles 1985, pp. 23-25

[Hwa86] Hwang, K.S., Mercer, M.R.:

Derivation and Refinement of Fanout Constraints to Generate Tests in Combinational Logic Circuits.

IEEE Transactions on Computer Aided Design, vol. CAD-5, no. 4, Oct. 1986

[Ill86] Illman, R.J.:
Design of a self testing RAM.
Proc. Silicon Design Conference, 1986, pp 439-446

[Ill89] Illman, R.:
Reviewers Report
Review of the ESPRIT project EVEREST, 1989, Munich

[Keu93] Keutner, K.:
SPLASH: A Highly Efficient Path Sensitization Method for Algorithmic/Structural Hardware Descriptions
submitted to IEEE Design&Test of Computers

[Kir79] Kirk, S.J.:
Life Cycle Costing: Problem Solver for Engineers
Specifying Engineer, Washington D.C. 1979, pp. 123-129

[Kje81] Kjellstrom, G., Taxen, L.:
Stochastic Optimization in System Design.
IEEE Transactions on Circuits and Systems, vol. cas-28, No. 7, July 1991

[Koe79] Koenemann, B., Mucha, J., Zwiehoff, G.:
Built-In Logic Block Observation Techniques
Proc. IEEE Test Conference, 1979

[Kov81] Kovojanic, P.G.:
Single Testability Figure of Merit.
Proc. ITC 1981

[Kre75] Kreyszig, E:
Statistische Methoden und ihre Anwendungen.
Vandenhoeck & Ruprecht Verlag Göttingen, 5th edition, 1975

[Kub84] Kuban, J.R., Bruce, W.C.:
Self Testing the Motorola MC 6804P2.
IEEE Design & Test, vol.1, no.2, May 1984

[Laf91] Laffitte, M.:

Interactive Test Strategy Planning: A Model and a Prototype.

Proc. European Test Conference, 1991

[Lar89] Larrabee, T.:

Efficient Generation of Test Patterns Using Boolean Difference.

Proc. ITC 1989

[Lue73] Luenberger, D.G.:

Introduction to Linear and Nonlinear Programming.

Addison Wesley Publishing Company, Reading Massachusetts, 1973

[Mad84] Madauss, B.J.:

Projektmanagement

2nd edition, Stuttgart 1984

[Mar86] Marlett, R.A.:

An Effective Test Generation System for Sequential Circuits.

Proc. DAC 1986

[McC86] McCluskey, E.J.:

Logic Design Principles

Prentice-Hall International Editions, 1986

[Mil91] Miles, J., De Bondt, R., Daemen, L.:

A Test Economics Model and Cost Benefit Analysis of Boundary Scan

Proc. European Test Conference, 1991

[Mye83] Myers, M.A.:

An Analysis of the Cost and Quality Impact of LSI/VLSI Technology on PCB Test Strategies

Proc. IEEE International Test Conference, 1987, pp. 382-395

[Ohl87] Ohletz, M.J., Williams, T.W., Mucha, J.P.:

Overhead in Scan and Self-Testing Designs.

Proc. ITC 1987

[Pab87] Pabst, J.S.:
Elements of VLSI Production Test Economics.
Proc. IEEE International Test Conference, 1987, pp. 982-986

[Poa63] Poage, J.F.:
Derivation of Optimum Tests to Detect Faults in Combinational Circuits.
Proc. Symposium on Mathematical Theory of Automata, Polytechnic Press, New York 1963

[Pyn86] Pynn, C.:
Strategies for Electronics Test.
McGraw-Hill, 1986

[Rat82] Ratiu, I.M.:
VICTOR: A Fast VLSI Testability Analysis Program.
Proc. ITC 1982

[Rei83] Reinertsen, D.G.:
Whodunit? The search for the new-product killers.
Electronic Business, vol. 11, July 1983, pp. 106 - 109

[Rog85] Rogers, W.A., Abraham, J.A.:
CHIEFS: A Concurrent, Hierarchical and Extensible Fault Simulator.
Proc. ITC 1985

[Rot66] Roth, J.P.:
Diagnosis of Automata Failures: A Calculus and A Method.
IBM Journal of Research & Development, vol. 10, July 1966

[Rot80] Roth, J.P.:
Computer Logic, Testing, and Verification.
Computer Science Press 1980

[Rot89] Roth, W., Johansson, M., Glunz, W.:
The BED Concept: A Method and a Language for Modular Test Generation
Proc. International Conference on VLSI, 1989, pp. 143-152

[Rub86] Rubinstein, R.Y.:
Monte Carlo Optimization, Simulation and Sensitivity of Queueing Networks
John Wiley & Sons, New York 1986

[Saa61] Saaty, T.L., Webb, K.W.:
Sensitivity and renewals in scheduling aircraft overhaul.
Proceedings of the Second International Conference on Operational Research, pp.708-716, English University Press, 1961

[Sal85] Saluja, K.K., Kinoshita, K., Boswell, C.:
A Design of Parallel Testable Programmable Logic Arrays.
Tech. report EE8501 Dept of Electrical and Computer Eng., Univ. of Newcastle, NSW, 2308, Australia, 1985

[Sch76] Schmitz, N. and Lehmann, F.:
Monte-Carlo Methoden 1
Verlag Anton Hain, Meisenheim am Glan, 1976

[Sch84] Schuster, M.D., Bryant, R.E.:
Concurrent Fault Simulation of MOS Digital Circuits.
Proc. Conf. Advanced Research in VLSI 1984

[Sch88] Schulz, M.H., Trischler, E., Sarfert, T.M.:
SOCRATES: A Highly Efficient Automatic Test Pattern Generation System.
IEEE Transactions on Computer-Aided Design, Jan. 1988

[Sed92] Sedmak, R.:
Economics and Other Management Issues
Design for Testability Training Course, held at Texas Instruments, Munich, 1992

[Sel68] Sellers, E.F., Hsiao, M.Y., Bearnson, L.W.:
Analysing Errors with the Boolean Difference.
IEEE Transactions on Computers 1968, pp. 676-683

[Ste77] StewartH.J.:

Future Testing of Large LSI Circuit Cards.

Dig. IEEE Semiconductor Test Conference 1977

[Su84] Su, S., Lin, T.:

Functional Testing Techniques for Digital LSI/VLSI Systems.

Proc. DAC 1984

[Szy92] Szygenda, S.A.:

Profit, liability, and education: Influencing factors on the economics of non-testing

in Proc. Economics of Design and Test for electronic circuits and systems, Ellis Horwood Limited, Chichester, 1992

[TEN92] Siemens-Nixdorf:

TENaphro user manual, 1992

[Tre85] Treuer, R., Fujiwara, H., and Agrawal, V.K.:

Implementing a self test PLA design.

IEEE Design and Test of Computers, vol. 2, Apr. 1985 pp 37-48.

[Tri82] Trischler, E.:

Scan Structures Study.

Siemens Report No. RTL-82-TR-003, 1982

[Tri83] Trischler, E.:

Testability Analysis and Incomplete Scan Path

Proc. IEEE International Conference on Computer Aided Design, 1983, pp.38-39

[Tri84] Trischler, E.:

ATWIG: An Automatic Test Pattern Generator with Inherent Guidance.

Proc. ITC 1984

[Tsu87] Tsui, F.F.:

LSI/VLSI Testability Design.

McGraw-Hill Book Company, 1987

[Tur90] Turino, J.:

Design to Test.

2nd edition, Van Nostrand Reinhold, New York, 1990

[Var84] Varma, P., Ambler, A.P., Baker, K.:

An Analysis of the Economics of Self-Test.

Proc. IEEE International Test Conference, 1984

[Wai85] Waicukauski, J.A., Eichelberger, E.B., Forlenza, D.B., Lindbloom, E., McCarthy, T.:

A Statistical Calculation of Fault Detection Probabilities by Fast Fault Simulation

Proc. ITC 1985

[Wil83] Williams, T.W., Parker, K.P:

Design For Testability - A Survey

Proc. IEEE, vol. 71, pp 98-112, Jan 1983

[Wue84] Wuebbenhorst, K.:

Konzept der Lebenszykluskosten

Verlag fuer Fachliteratur, Darmstadt, 1984

[Zhu86] Zhu, X.:

A Knowledge Based System for Testable Design Methodology Selection PhD

[Zhu88] Zhu, X., Breuer, M.: Analysis od Testable PLA Designs IEEE Design & Test of Computers, vol.5, no. 4, Aug 1988, pp.14-28

Proc. DAC 1982

Appendix A

Determination of Correlation Coefficients

In chapter 6 the author has described and performed several sensitivity analysis techniques for cost models which are all based on variates as input parameters. Most of the input parameters are independent from each other, but some of them are correlated. Due to being pairwise correlations, the related variates can be generated by using the correlation coefficients.

In this appendix, the author will describe the method to derive the correlation coefficients, and the correlation coefficients will be derived from statistical data for the correlations between the gate count and the number of cells, the gate count and the number flip flops and between the number of flip flops and the sequential depth.

1. Algorithm to Determine Correlation Coefficients

Based upon a table of values for two variates X and Y with expectations of μ_X and μ_Y and variances of σ_X^2 and σ_Y^2, which are correlated, the correlation coefficient can be determined as follows:

- determine the covariance of X and Y:

$$\sigma_{XY} = E([X - \mu_X] \cdot [Y - \mu_Y])$$

  where E( ) stands for the expectation or the mean value of the term in brackets.

- calculate the correlation coefficient by

$$\rho = \frac{\sigma_{XY}}{\sigma_X \cdot \sigma_Y}$$

2. Calculation of Correlation Coefficients

We have received data about the gate count and the related cell count for 21 standard cell designs, which were designed at Siemens. The data are presented in table 1.

Design	Gates	Cells
SIE1	2085	485
SIE2	2000	300
SIE3	1000	200
SIE4	3922	2500
SIE5	7717	1844
SIE6	1955	1022
SIE7	5493	1025
SIE8	14806	4076
SIE9	4207	12
SIE10	15418	3005
SIE11	13849	2723
SIE12	45028	4274
SIE13	27990	4095
SIE14	33042	1807
SIE15	264	27
SIE16	326	98
SIE17	3881	787
SIE18	4872	1520
SIE19	1685	675
SIE20	3096	1062
SIE21	217	46
μ	9183.476	1503.952381
σ	12181.59	1422.973172
σ_{XY}		13479221.22
ρ		**0.777614131**

Table 1: Gate count and related number of cells for Siemens standard cell designs

We have selected 27 ISCAS'89 benchmark circuits ([Brg89]) and 11 standard cell designs to derive the correlation factor for the gate count and the number of flip flops. The data are presented in table 2.

Name	gates	dff
s1196	747	18
s1238	739	18
s13207	9958	669
s1423	1192	74
s1488	879	6
s1498	877	6
s15850	14376	597
s208	161	8
s298	267	14
s344	273	15
s349	274	15
s35932	21249	1728
s38584	23609	1452
s386	271	6
s420	326	16
s444	380	21
s510	287	6
s526	451	21
s5378	3316	179
s641	565	19
s713	585	19
s820	536	5
s832	542	5
s838	653	32
s9234	7540	228
s953	635	29
UB1	3801	74
SIE22	7327	420
SIE23	9513	788
SIE24	9780	400
SIE25	11000	900
NS1	20000	700
SIE26	22428	140
SIE27	32185	2500
SIE12	45000	1500
SIE28	55198	1600
SIE29	70000	1500
μ	10187	425
σ	16529	643
σ_{XY}		8656983.442
ρ		**0.814090569**

Table 2: Gate count and related number of flip flops

The correlation factor for the correlation between the number of flip flops and the

sequential depth for the circuits in table 3 was derived by calculating the average number of test patterns per target fault which were generated by SNI's sequential ATPG system TENace and by setting the sequential depth to that value.

Circuit	#dff	seq. depth
s208	8	2.29
s298	14	5.86
s344	15	2.95
s349	15	2.84
s386	6	1.69
s444	21	9.85
s641	19	1.33
s713	19	1.37
s1196	18	1.25
s1238	18	1.16
μ	15.3	3.06
σ	4.90	2.77
σ_{XY}		3.087311
ρ		0.227071

Table 3: Number of Flip flops and related sequential depth

On the basis of the calculated correlation factors, the following values were used for the sensitivity analysis in chapter 6:

ρ_{cell}: 0.78

$\rho_{flipflop}$: 0.81

$\rho_{seqdepth}$: 0.23

Appendix B

Listing of the Cost Models

Appendix B: Cost Model Parameters and Syntax of Equations

In the following the syntax for describing equations is listed. The meta language used here in the one of the UNIX tool YACC.

```
equation :  "(" equation ")"     |
            equation "+" equation  |
            equation "-" equation |
            equation "*" equation  |
            equation "-" equation  |
            equation "%" equation  |
            equation "^" equation  |
            equation "==" equation  |
            equation "<" equation |
            equation ">" equation   |
            "-" equation  |
            "+" equation  |
            equation "!"  |
            equation "?" equation ":" equation  |
            function  |
            double   |
            parameter  |
            in_var
double :double constant
function :  "@LOG10"  "(" equation ")"  |
            "@PUC"  "(" equation ")"|
            fname  "(" par_list ")"  |
            fname  "("  ")"
par_list : |
            equation |
            equation "," par_list
parameter : NAME
in_var:     one of the design identifiers listed in chapter 8.5
```

Definition of the parameters

In the following list the cost model parameters are defined by its internal name and the data type of its value. Two signs have special meanings:

- The "~" means, that this parameter is expanded for an application to the number of TUs in the design by copying the internal name and indicating it by consecutive numbers.

The meaning of the parameters will be described later.

1) Design Independant Primary Parameters

Identifier	Data Type
kdes	float
kp	float
cexp	float
hpw	float
exper	int
pcad	float
descentrate	float
equrate	float
costrate	float
fpg	float
fcv	float

2) Cost-Model Design-Dependant Primary Parameters

Identifier	Data Type
nre	float
vol	int
percuse	float
cputime	float
fcreq~	float
tpf~	float
tpver	float
pms	int
pps	float

3) Cost-Model Test-Dependant Primary Parameters

Identifier	Data Type
numtp_normal	int
numtp_scan	int
numtp_self	int
cells~	int
in	int
out	int
bi	int
cperf~	float

or~	float
cgate~	int
avs~	float
costtgs~	float
fcach~	float

4) Secondary parameters

Identifier	data type	equation
puc	float	@PUC
pc	int	nre+vol*puc
plib	float	1/cells
pdes	float	1-1/(kdes+exper)
prod	float	kp*pcad*pdes*plib
cpin	int	2*in+out+3*bi
lcompl~	float	cperf~*(1-or~)*cgate~
ocompl	float	cpin*lcompl^cexp
mp	float	compl/prod
engcost	float	mp*costrate
descentcost	float	percuse*destime*descentrate
mainframecost	float	cputime*equrate
descost	float	engcost+descentcost+mainframecost
faults~	int	cgate~*fpg
faults	int	faults&
remfaults~	float	((fcreq~-fcach~)>0)*(fcreq~-fcach~)*faults~
mtgtime~	float	remfaults~*tpf~/hpw
mtc	float	mtgtime*costrate
faultsa~	int	((fcach~-fcv)>0)*(fcach~-fcv)*faults~
numtpa~	int	faultsa~*tppf*(avs~+1)
numtpm~	int	remfaults~*(avs~+1)
tpgen~	int	numtpa~+numtpm~
tsl	int	tpver + numtp_normal + numtp_scan + numtp_self
tac	float	(tsl-tsl%pms)/pms*pps*vol
tcost	float	tac+mtc+costtgs
ovcost	float	pc+descost+tcost+ttmcost

Description of the Parameters:

1) Design Independant Primary Parameters

kdes:

Normalising factor, which normalises the designers experience. The factor normalises the productivity of a designer with no experience related to the productivity of most experienced designer. See equation for pdes

kp:

Linear factor to normalise the design time.

cexp:

Exponent to derive the design complexity from the gatecount

hpw:

Working hours per week

exper:

Designer"s experience in number of designs he has already performed

pcad:

Productivity of the CAD system as a linear paramater of the design time

descentrate:

Cost rate of hiring a design center per week

equrate:

Accounted cost per CPU-hour for the usage of mainframe computers

costrate:

Weekly cost rate of designer

fpg:

Average number of stuck-at faults per gate

fcv:

Typical fault coverage achieved by verification patterns

2) Design-Dependant Primary Parameters

nre:

Non-recurring engineering cost, accounted by the silicon producer

vol:

Production volume

percuse:

Percentage of design time, a accounted design center is used

cputime:

CPU time in hours, an accounted equipment is used

fcreq˜:

Required fault coverage per TU

tpf˜:

Time needed to get a test pattern set for testing a single fault by hand

tpver:

Number of test patterns for verification

pms:

Number of test patterns, for which the price for test application increases steplike

pps:

Price per additional test pattern set per device

costtgs:

Cost for ATPG

3) Test-Dependant Primary Parameters

avs˜:

Average sequential depth per TU; here the typical number of clock cycles (or patterns) is meant for controlling and observing internal nodes.

numtp_normal:

number of test patterns per chip to be applied by a test equipment

numtp_scan:

number of scannable test patterns per chip to be applied by a test equipment

numtp_self:

number of self test patterns per chip to be applied

puc:

Production unit cost

cells:

Number of cells

in:

Number of input pins

out:

Number of output pins

bi:

Number of bidirectional pins

cperf:

Performance complexity; the performance implication is a linear cost factor whose value range is between 1 and ∞ . A value of 1 indicates the uncritical case whereas ∞ means, that it is impossible to perform the design. For example, a factor of '2' doubles the design time.

or:

Originality; this linear parameter describes, whether some of the architecture, functions or algorithms have been developed before, and in what way this knowlegde accelerates the design process. The value must be in between 0 (design is fully original) and 1 (the TU has already been designed). cgate~:

Equivalent gate count per TU, derived from the equivalent gatecount of the cells used; this parameter is a complexity measure as well for estimating the design time as for estimating the test pattern generation cost. In addition, the production unit cost are derived from this parameter.

cgate:

Equivalent gate count of chip; this value is simply the sum of the TU's gatecounts.

4) Secondary Parameters

pc:

Production cost

plib:

Productivity of the cell library used; by this parameter it is calculated, how well the cell library fits to the design.

pdes:

The experience of the designer; this parameter is based upon the number of designs, the designer has already performed

prod:

Overall productivity, composed of productivity of the cell library, productivity of the CAD system and the experience of the designer.

cpin:

Pin complexity

compl:

Overall complexity, composed of performance complexity, pin complexity, originality and the equivalent gatecount.

mp:

Design effort in weeks

numdes:

Number of designer needed

engcost:

Engineering cost of the design

descentcost:

Cost for hiring a design center

mainframecost:

Cost for using computers accounted for CPU-time

descost:

Overall design cost

faults:

Number of possible stuck-at faults on chip

fcach:

Achievable fault coverage by ATPG

remfaults:

Number of aborted faults from ATPG

mtgtime:

Time needed for manual TPG

mtc:

Overall cost for manual TPG

faultsa:

Number of faults, for which test patterns were generated by the ATPG system

numtpa:

Number of test patterns per TU generated by the ATPG system

numtpm:

Number of test patterns generated by hand

tpgen:

Number of generated test patterns

tsl:

Number of overall test patterns (test set length)

tac:

Test application cost

tcost:

Overall test cost

ovcost:

Overall or final cost

EVEREST Activity Report
Task ID: 095
Activity: 3.2.4.b

Test Economics Model Development

CEC Deliverable: No

Distribution: Free

Authors: Jochen Dick, SNI AG

Approved: Francis Gourdy, Bull S.A.

Table of Contents

1. Introduction 3
2. The phase model of a VBS life cycle 4
3. Model overview 6
4. A model of the cost of PCBs 8
4.1 A model of the development costs 8
4.2 A model of the production costs 11
4.3 A model of the test costs 12
4.4 Cost summaries 17
5. A model of the cost of VBSs 19
5.1 A model of the development costs 19
5.2 A model of the assembly costs 21
5.3 A model of the test costs 21
5.4 Cost summaries 23
6. Conclusions 24
7. References 25

1. Introduction

The purpose of this activity was to develop a model which enables to predict all test-sensitive costs of a VLSI based system (VBS). The application of the model is the planning of test strategies for VBSs, which may have an impact on the overall cost the VBS product. This prediction should be based on known data. The planning of test strategies should be done in the specification phase of the VBS product. So, the data needed to use the test economics model should be known in the specification phase.

The model is implemented as a set of equations which reflect the design and production of the VBS. The input parameters of the test economics model are data which are typically known at the end of the specification. The output data of the model are the costs for the development and production of the analysed VBS.

This report describes the underlying scenario for the development and production of VBSs and the parameters of the test ecomics model. The scenario is described as a hierarchical phase model. The levels of integration (e.g system - board - component) are modeled by the hierarchy of the phase model. The development and production tasks are reflected by the phases of the model. The parameters of the test economics model are partitioned in order to fit into the hierarchical phase model. Each parameter is described by its meaning and its dependency on other parameters.

2. The phase model of a VBS life cycle

In order to give the test economics model a clear structure we developed a model of cost areas which reflect the in-house life cycle of VBSs. By in-house life cycle we mean all phases of a VBS product covered by its development and production. The in-house life cycle is part of the overall life cycle of the product. Typically a VBS is build upon different levels of assembly. From a testing point of view, the most important levels of assembly are the component level, the board level and the system level. Different test methods are existing for the different levels, and the occuring defect types to detect are also different. The objective of testing at component level on one side and board or system level on the other side is different:

- At component level a non-working component normally is simply sorted out.
- At board and system level, for a non-working board or system the defect needs to be located by a fault diagnosis. Then the defect is repaired.

Other levels of assembly, like multi-chip-modules can be handled as one of the three levels mentioned here.

For each hierarchy level, the phases are modeled seperately. Some of the phases can be neglected for an economic analysis of test strategies, because their costs are not relevant for test strategy planning. These phases are not included in this test economics model. The test economics model for components was developed in the first part of this subtask (/4/). It will not be presented again in this report. Figure 1 presents the phase model, which is used for the test economics model.

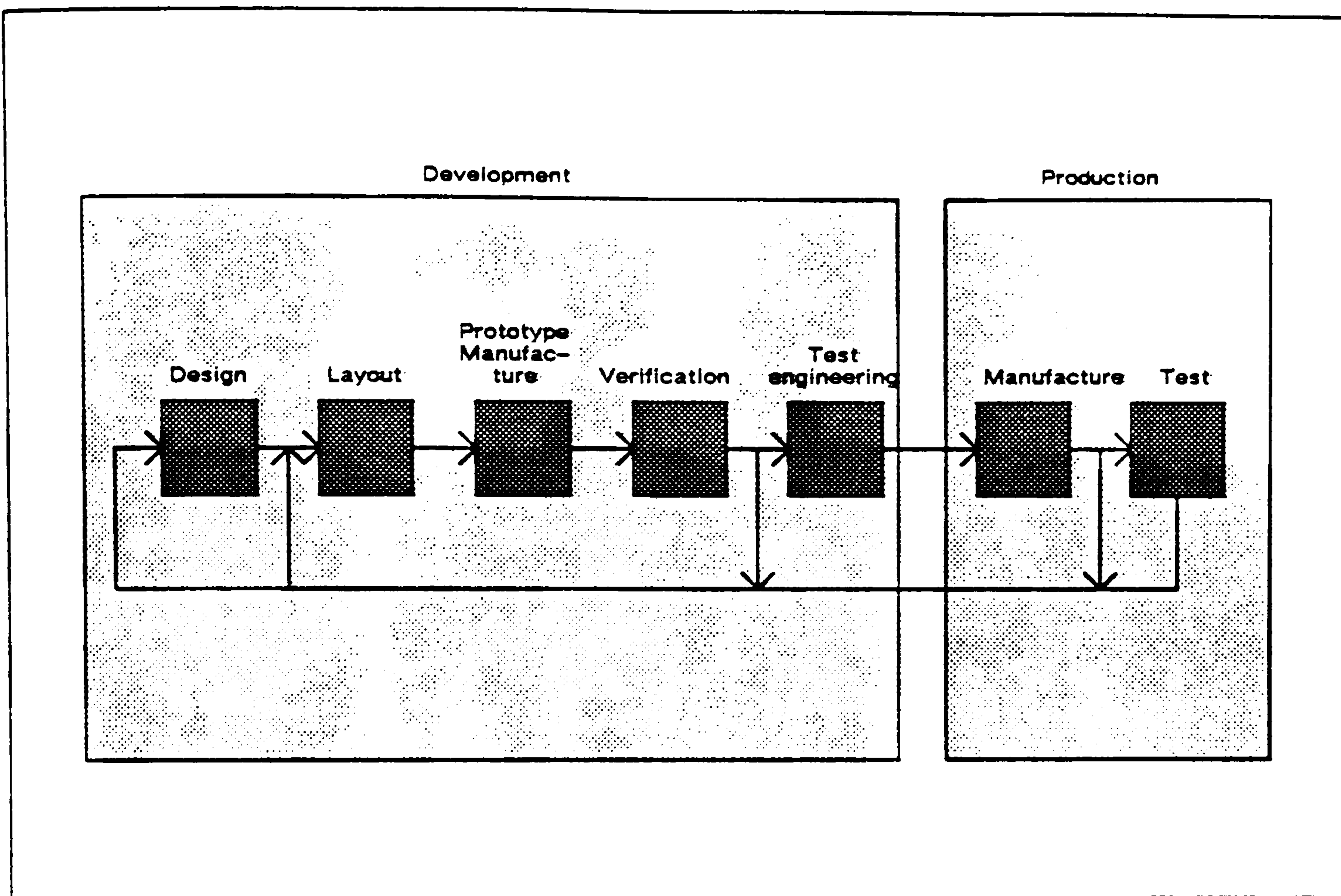


Figure 1: Phase model of development and production of VBSs and PCBs

In the following the phases are described:

- The design phase includes the initial design, design entry and simulation.
- The layout phase includes the placement and floor planning of the PCBs.
- The prototype manufacture covers the construction and manufacture of prototypes.
- The verification phase covers the verification of the board or system by evaluating the prototypes.
- Test engineering covers the generation of test patterns and the generation of test programs.
- The manufacture includes the production preparation, fabrication and assembly.
- The test phase includes test tool manufacture such as the manufacture of adaption, and test application.

3. Model overview

The model is partioned into three levels, where the costs of each level are included in the model on top of it. For each level, cost areas are defined relating to the tasks of the development and production. All costs are classified into the following three classes:

- **Volume related cost (VRC)** are all expenditures, which occur per device to produce.
- **Non recurring costs (NRC)** are those expenditures, which occur once per product type.
- **Investments (INV)** are all expenditures, which can be shared with other products.

The data needed in the model from a lower level model does not include all details, i.e. all parameter values of the lower level. This would make a cost analysis much too complex. But the overall costs of the lower level need to be classified and seperated into the cost classes specified above. See figure 2 for the hierarchy model.

The model is implemented as a parameterised cost model. This means that each cost is derived from technical and cost parameters by a set of equations. This way of modeling makes sense, if a variation of the parameters leads to a variation of the accompanied costs. It makes no sense, if costs occur in terms of prices to be paid no matter what the parameter values are.

PAGINATION AS IN ORIGINAL

4. A model of the cost of PCBs

In this chapter all parameters of the PCB test economics model are described. Every parameter is identified by its match name, which stands at the beginning of the parameter description. The match name is used for describing the equations of other parameters, which are dependent on this parameter. A brief description of the parameter is following. If the parameter can be classified as mentioned in chapter 3, the class is specified in brackets. If the parameter is a secondary parameter (i.e. the parameter value depends on the values of other parameters), the equation is described centered in bold letters.

Global Parameters:

HPW: Hours per working week

HPM: Hours per working month

AWH: Annual working hours

Mi: Market interest rate / year

Npba: Maximum number of boards expected to be produced per year

TNPBA: Total Number of Boards Expected to be produced in the systems life cycle

4.1 A model of the development costs

Ipre: iterations before production

Ipost: Iterations after production

SPECIFICATION

Tspec: Initial specification time

Kspec: Labour rate per hour

Cspec: Specification labour cost (NRC)

KSPEC*TSPEC

DESIGN (ENTRY + SIMULATION)

Tid: Average time taken for initial design

d: Design iteration factor; this means the relation of design effort of a redesign to the original design effort.

Kde: Labour rate per design hour

KCAE: Hourly rate for cae software& equipment / h

Tdes: Time taken to complete design

(1-D^(IPRE+IPOST+1))/(1-D)*TID

Cdes: Total labour cost for design (NRC)

KDE*TDES

Cdeq: Total cost of cae software & equipment (NRC)

TDES*KCAE

LAYOUT

Tilay: Average time taken for initial layout

l: Layout iteration faktor

Kl: Labour rate per layout hour

KLAY: Cost rate for layout software & equipment / h

Tlay: Time taken to complete layout

(1-L^(IPRE+IPOST+1))/(1-L)*TILAY

Clay: Total labour cost for layout (NRC)

KL*TLAY

Cleq: Total cost of layout software & equipment / h (NRC)

KLAY*TLAY

PROTOTYPE FABRICATION

Tip: In-house prototype construction time (h)

p: Prototype iteration factor

Kp: Labour rate per hour of prototype phase

Cmat: Material cost for prototype per iteration

Nprot: Number of prototypes per iteration

Tp: Total prototype time

(1-P^(IPRE+IPOST+1))/(1-P)*TIP

Cp: Prototype cost(including material) (NRC)

(1+IPRE+IPOST)*CMAT*NPROT+KP*TP

VERIFICATION (PROTOTYPE TEST)

Tfv: Functional verification time(h)

Tsysv: Verification time in the system (h)

v: Verification iteration factor

Kv: Labour rate for verification per hour

KVER: Cost rate for verification equipment / hour

Tv: Total verification time

(1-V^(IPRE+IPOST+1))/(1-V)*(TFV+TSYSV)

Cv: Labour verification cost (NRC)

KV*TV

Ceqv: Total cost of verification equipment (NRC)

TV*KVER

TEST ENGINEERING

te: Production test iteration factor

Prg: Length of test program (1000s lines)

Fga: Fraction of test program genererated automatically

Tprg: Length of tools for automatic test program generation (1000s lines) to be developed

Kte: Labour rate of test engineers

Ktee: Cost rate for test engineering equipment / h

Tte: Total test engineering time (h)

((2,4*(PRG*(1-FGA))^1,1))*HPM

Tswt: Total software engineering time for tools(h)

(2,2*TPRG^1,05)*HPM

Cte: Test engineering labour cost

KTE*TTE

Csw: Software engineering cost

KTE*Tswt

Cteq: Total cost for test engineering equipment (NRC)

(TTE+Tswt)*KTEE

4.2 A model of the production costs

MANUFACTURE PHASE

Cbb: Bare board cost

Cpp: Production prepare cost

Cc: Component cost / board

Number of Components

nax: Axial

nrad: Radial

nic: ICs

nsmd: SMDs

nman: Manual

noth: Others

Cost of Assembly per Component

Caax: Axial

Carad: Radial

Caic: ICs

Casmd: SMDs

Caman: Manual

Caoth: Others

Cass: Overall assembly cost/ board (VRC)

NAX*CAAX+NRAD*CARAD+NIC*CAIC+NSMD*CASMD+NMAN*CAMAN+NOTH*CAOTH

PUC: Overall production cost / board (VRC)

CASS+CBB+CC+CPP/SPV

4.3 A model of the test costs

TEST PHASE

MTBF: Mean time between failure per board (h)

Fsh: Shorts per 100 boards

Fop: Opens per 100 boards

Fpla: Wrong/missing compon. per 100 boards

Fan: Faulty analog & digital ICs without boundary scan

Fdig: Faulty digital ICs per 100 boards

Foth: Other faults per 100 boards

FPB: Total faults 100 boards

SUM(FSh..Foth)

PV: Maximum production volume / y

NPBA

Yp: Production yield

EXP(-FPB/100)

QUALITY MODEL

nit: Number of iterations in test

FCsh: Shorts fault coverage

FCop: Opens fault coverage

FCpla: Wrong/missing component fault coverage

FCan: Fault coverage of analog & digital ICs without boundary scan

FCdig: Digital fault coverage

FCoth: Fault coverage of other faults

Eg: Good board coverage

USh: Undetected shorts remaining

FSH

Uop: Undetected opens remaining

FOP

Uwr: Undetected wrong/missing components remaining

FPLA

Uan: Undetected faulty analog & digital without boundary scan remaining

FAN

UnDig: Undetected faulty digital remaining

FDIG

UnOth: Undetected other faults remaining

FOTH

Tfun: Total undetected faults remaining

SUM(USh..UnOth)

DetSh: Detected shorts

FCSH*USH

DetOp: Detected opens

FCOP*UOP

DetPla: Detected wrong/missing components

FCPLA*UWR

DetAn: Detected faulty analog & digital without boundary scan

FCAN*UAN

DetDig: Detected faulty digital

FCDIG*UNDIG

DetOth: Detected others

FCOTH*UNOTH

ds: Total detected faults per 100 boards

SUM(DetSh..DetOth)

Yat: Yield after test

EXP(-(TFUN-DS)/100)

Ngf: Number of good boards failed

(1-EG)*NTEST*YP

Tit: Test iteration factor

YAT*(1-YP)

Ntest: Number of boards going to test

(1-TIT^(NIT+1))/(1-TIT)*$PV

Ndr: Number of boards going to diagnosis/repair

(1-TIT^NIT)/(1-TIT)*$PV*TIT

Nnr: Number of non-repairable boards

PV*TIT^(NIT+1)

TEST TIME MODEL

Lot: Lot size

A: Acceleration factor (accelerating life test)

Tsu: Set up time/lot (minutes)

Tatt: Attended test time for good b.(min)

Tun: Unattended test time (minutes)

Tdiag: Diagnosis time per fault (minutes)

Trep: Repair time per fault (minutes)

Tdgf: Diagnosis time for good boards failed (min)

Tteff: Effective test time for good board (minutes)

TATT+TUN+TSU/LOT

Td: Diagnosis & repair time/board

TDIAG+TREP

TTY: Total test time / y (hours)

TTEFF*NTEST/60

TER: Testers required

INTEGER (TTY/(AWH))+1

Tdryr: Total diagnosis/repair time per year (h)

TD*NDR+NGF*TDGF)/60

TRER: Diagnosis/repair stations required

INTEGER (TDRYR/(AWH))+1

TEST COST MODEL

Kt: Test cost per minute

Kd: Diagnosis cost per minute

Cd: Cost of diagnosis (VRC)

KD*TD

Ct: Cost of testing (VRC)

KT*(TATT+TSU/LOT)

Cptr: Total cost/board (VRC)

CD+CT

Cann: Annual production test cost

NTEST*CT

PATE: Purchase cost for ATE (INV)

UpATE: Utilization period of ATE/ y (as systems production time)

OvATE: Variable operating costs / min (VRC)

DcATE: Depreciation cost / y for ATE (INV)

PATE/UPATE

IcATE: Interest cost / y for ATE (INV)

(PATE/2)*MI

OfATE: Other fixed operating costs / y (NRC)

TATEY: Total fixed costs for 1 ATE (during systems production time)

(DCATE+ICATE+OFATE)*UPATE

PDRE: Purchase cost for diagnosis/repair equipment (INV)

UpDRE: Utilization period of PDRE

UPATE

OvDRE: Variable operating costs / min (NRC)

DcDRE: Depreciation cost / y for DRE (INV)

PDRE/UPDRE

IcDRE: Interest cost / y for DRE (INV)

(PDRE/2)*MI

OfDRE: Other fixed operating costs / y (NRC)

TDREY: Total fixed costs for 1 diagnosis/repair equipment / y

DCDRE+ICDRE+OFDRE

Cpeq: Equipment cost per test stage (VRC)

TATEY*TER+TDREY*TRER

Cpte: Total test & diagnosis/repair equipment cost / y

SUM(Cpeq of all testers)

CTdr: Total diagnosis & repair cost per boards

SUM(Cd of all d/r stations)

CTt: Total test cost per boards

SUM(Cd of all testers)

CTptr: Total production test & repair per boards

SUM(CTt, CTdr)

PRC: Annual production test & repair cost

NTEST*CT+NDR*CD

APTR: Overall production test & repair cost

SUM(PRC over all test&repair stations)

Clb: Cost of lost boards (VRC)

PUC*SUM(Nnr over all tests)

4.4 Cost Summaries

DEVELOPEMENT COSTS

DLAB: Labour

CSPEC+CDES+CLAY+(CP-((1+IPRE+IPOST)*NPROT*CMAT))+CV+CTE

DMAT: Material

NPROT*CMAT*(IPRE+IPOST+1)

DEQU: Equipment

CDEQ+CLEQ+CEQV+CTEQ

SWD: Sofware development cost (for test engineering tools)

Csw

NRE: Total developement costs

SUM(DLAB, DMAT, DEQU, SWD)

MANUFACTURING COSTS

PUC: board production cost

tpc: Total production cost

PUC*TNPBA

TEST COST

TLAB: Test-related labour cost per board

APTR/PV

TEQU: Total annual test,d.r. eq. cost per board

CPTE/$PV

TLB: Cost of lost boards / board

CLB/$PV

ttc: Total test cost

(SUM(TLAB, TEQU, TLB))*TNPBA

ov: Overall board cost through the life-cycle

TPC+TTC+NRE

5. A model of the cost of VBSs

The test economics model for the VBS is mainly composed of the board costs derived from the test test economics model for boards.

5.1 A model of the development costs

In the area of development costs, only the test engineering costs are relevant. The development costs of the boards are summarised in here.

Ipre: Iterations before production

Ipost: Iterations after production

<u>BOARD DEVELOPMENT COST</u>

BDEV: Development costs of all boards

SUM (all board's NRE)

<u>PROTOTYPE FABRICATION</u>

Tip: In-house prototype construction time (h)

p: Prototype iteration factor

Kp: Labour rate per hour of prototype phase

Cmat: Material cost for prototype per iteration

Nprot: Number of prototypes per iteration

Tp: Total prototype time

(1-P^(IPRE+IPOST+1))/(1-P)*TIP

Cp: Prototype cost(including material) (NRC)

(1+IPRE+IPOST)*CMAT*NPROT+KP*TP

VERIFICATION (PROTOTYPE TEST)

Tfv: Functional verification time(h)

Tsysv: Verification time in the system (h)

v: Verification iteration factor

Kv: Labour rate verification per hour

KVER: Cost rate for verification equipment / h

Tv: Total verification time

(1-V^(IPRE+IPOST+1))/(1-V)*(TFV+TSYSV)

Cv: Labour verification cost (NRC)

KV*TV

Ceqv: Total cost of verification equipment (NRC)

TV*KVER

TEST ENGINEERING

te: VBS production test iteration factor

Prg: Length of test program (1000s lines)

Fga: Fraction of test program genererated automatically

Tprg: Length of tool for automatic test generation (1000s lines)

Kte: Labour rate of test engineers

Ktee: Cost rate for test engineering equipment / h

Tte: Total test engineering time (h)

(2,4*(PRG*(1-FGA))^1,1)*HPM

Tswt: Total software engineering time for tools(h)

(2,2*TPRG^1,05)*HPM

Cte: Test engineering labour cost

KTE*TTE

Csw: Software engineering cost

KTE*Tswt

Cteq: Total cost for test engineering equipment (NRC)

(TTE+Tswt)*KTEE

5.2 A model of the assembly costs

The assembly cost itself are assumed to be not relevant for the cost evaluation of test strategies. For that reason the VBS assembly cost are composed of the production cost of the boards.

BOARD PRODUCTION COST

BPRC: Production costs of all boards

SUM (all board's tpc)

5.3 A model of the test costs

BOARD TEST COST

BTC: Test costs of all boards

SUM (all board's ttc)

SYSTEM TEST

The system test cost are modeled in the same way as the board test cost. Use the board test cost model here.

STC: Total system test cost

INSTALLATION

Tass: System assembly time (h)

Ttinst: Installation test time per VBS (h)

Kinst: Incremental installation cost per hour

Finst: Probability of early life failure per VBS

1-EXP((-TTINST*NPBS-TOP)/MTBF))-SUM(all test times)/100

Tinst: Time taken for installation per VBS (h)

TASS+(1+FINST+FPROD)*TTINST+(FINST+FPROD)*TDINST

Cir: Cost of installation and repair per PBA

TINST*KINST+(FINST+FPROD)*CR

Cira: Annual cost of installation and repair

CIR*NPBA

FIELD

Tw: Ave operation time under warranty/PBA

Kfld: Incremental field repair cost per hour

Fw: Probabilty of early life failure in warranty

1-EXP(-TW/$MTBF)+FPROD

Cfr: Cost of field repair per PBA

FW*(TDINST*KFLD+CR)

Cfra: Annual cost of warranty repair

CFR*NPBA

RETROFIT COSTS

Nmod: Average number of VBS modified/retrofit

Cret: Average retrofit cost per VBS

TCret: Total cost

IPOST*NMOD*CRET

5.4 Cost summaries

DEVELOPEMENT COSTS

Tdc: Overall development cost

BDEV+
(CP-((1+IPRE+IPOST)*NPROT*CMAT))+CV+CTE+
NPROT*CMAT*(IPRE+IPOST+1)+
CDEQ+CLEQ+CEQV+CTEQ+
Csw

PRODUCTION COSTS

Tpc: Total production cost

BPRC*TNPBA

TEST COSTS

TtC: Total test cost

BTC+STC+Cira+Cfra+TCret

TOTAL COST

Ov: Overall system cost

Tdc+Tpc+Ttc

6. Conclusions

A test economics model was presented which allows to evaluate the economic impact of test method applications. This model will be evaluated in activity 3.2.4.d (data gathering) and 3.2.4.f (evaluation) by using the model for an evaluation with real world data. The impact of test methods to the input parameters of the model is studied in activity 3.2.4.c (test methods studies). A report on this activity will be delivered soon. In the next step the existing test strategy planner (developed in activity 3.2.4.e) will be refined. We will adopt the test economics model for VBS, and the test strategies to be planned will be extended by the test methods for boards and systems.

7. References

Books:

/1/ Pynn, Craig T.:
The Low Cost Board Test Handbook
Zehntel Inc. internal print

Papers and Brochures:

/2/ Miles, John; De Bondt, R.; Daemen, L.:
A Test Economics Model & Cost-Benefit Analysis of Boundary Scan
Proceedings 2nd European Test Conference 1991, Munich 1991,
pp. 375-384

/3/ Hewlett Packard:
A Test Workcell Analysis Tool
Brochure of Hewlett Packard Company
Loveland, Colorado 1986

/4/ Dick, Jochen:
A Test Economics Model for ASICs
Everest Activity Report

Appendix C

Costing Data for a Boundary Scan and In-Circuit Test Strategy

Appendix C: Costing Data for a Boundary Scan and In-Circuit Test Strategy

		Development Cost	
		STRATEGY 1 (WITHO. B/S)	STRATEGY 2 (WITH B/S)
Number of boards to be produced	tnpba1	5201	5207
2. DEVELOPEMENT PHASE			
Iterations before production	Ipre	1	1
Iterations after production	Ipost	0,5	0,5
2.1 SPECIFICATION			
Initial spec. Time (h)	Tspec	400	400
Hourly Labour Rate	Kspec	120 DM	120 DM
Specification Labour Cost	Cspec	48.000 DM	48.000 DM
2.2 DESIGN (ENTRY + SIMULATION)			
Aver.Time taken for Initial Design	Tid	600	620
Design iteration factor	d	20%	20%
Hourly Design Labour Rate	Kde	120 DM	120 DM
Hourly CAE Softw.& Eqipment Rate	KCAE	0 DM	0 DM
Time Taken To Complete Design (h)	Tdes	737	761
Total Labour Cost for Design	Cdes	88.390 DM	91.336 DM
Total Cost of CAE Softw.& Eq.	Cdeq	0 DM	0 DM
Total Design Cost		88.390 DM	91.336 DM
2.3 LAYOUT			
Aver.Time taken for Initial Layout	Tilay	150	180
Layout Iteration Faktor	l	10%	10%
Hourly Layout Labour Rate	Kl	100 DM	100 DM
Hourly Layout Soft.& Eqipment Rate	KLAY	0 DM	0 DM
Time Taken To Complete Layout (h)	Tlay	166,13962039	199,367544468
Total Labour Cost for Layout	Clay	16.614 DM	19.937 DM
Total Cost of Layout S.& Eq./h	Cleq	0 DM	0 DM
Total Layout Cost		16.614 DM	19.937 DM
2.4 PROTOTYPE FABRICATION			
In-house Prototype Constr.time (h)	Tip	120	120
Prototype iteration Factor	p	90%	90%
Hourly Prototype Labour Rate	Kp	100 DM	100 DM
Mat.Cost for Prototoper Iteration	Cmat	5.000 DM	5.700 DM
Number of Prototypes per Iteration	Nprot	1	1
Total Prototype Time (h)	Tp	278	278
Prototype Labour Cost	Cpl	27.788 DM	27.788 DM
Prototype Material Cost	Cp	12.500 DM	14.250 DM
Total Prototype Cost		40.288 DM	42.038 DM
2.5 VERIFICATION (PROTOTYPE TEST)			
Functional Ver. Time (h)	Tfv	500	500
Verif. time in the System (h)	Tsysv	500	500
Verification Iteration Factor	v	80%	80%
Hourly Verification Labour Rate	Kv	120 DM	120 DM
Hourly Verification Equipment Rate	KVER	0 DM	0 DM
Total verification time (h)	Tv	2138	2138
Labour Verification Cost	Cv	256.540 DM	256.540 DM
Total Cost of Verification Eq.	Ceqv	0 DM	0 DM
Total Verification Cost		256.540 DM	256.540 DM

1. BASIC ASSUMPTIONS

Application Period		T=0
Hours per working week	HPW	37
Hours per working month	HPM	155
Annual working hours	AWH	1860
Market interestrate/ Y	Mi	12%
Depreciation Period of ATE in Years (As Systems Production Time)	Depy	6

2.6 TEST ENGINEERING

		STRATEGY 1 (WITHO. B/S)			STRATEGY 2 (WITH B/S)		
Test Stage Name		VI	ICT	FUNCTIONAL	VI	B/SCAN TEST	FUNCT +B/S
Test Plan	Tpl	0	1	0	0	1	0
Length of Test Prog.(1000s lines)	Prg	0	500	0	0	550	18
Fraction of Test Prog.gener autom.	Fga	0%	80%	10%	0%	0%	10%
Length of Tool for autom.Test.gen. (1000s lines)	Torg	0	0	0	0	0	0
Iteration factor	ft	0,00%	20,00%	0,00%	0,00%	20,00%	0,00%
Hourly Test Eng. Labour Rate	Kte	0 DM	100 DM	120 DM	0 DM	90 DM	120 DM
Hourly Test Eng. Equipment Rate	Ktee	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Preparation Cost /ATE (fixture)	PrepC	0 DM	30.000 DM	0 DM	0 DM	5.000 DM	0 DM
Software Eng. Time for Tools (h)	Tto	0	0	0	0	0	0
Test Eng. Time for Test Prog. (h)	Tte	0	259	0	0	298	0

		Strategy 1	Strategy 2
Labour Cost for Test Engineering	Cte	25.900 DM	26.640 DM
Total Cost for Test Eng.Equip.	Cteq	0 DM	0 DM
Total Test Engineering Cost		25.900 DM	26.640 DM

3 PRODUCTION PHASE

		STRATEGY 1 (WITHO. B/S)	STRATEGY 2 (WITH B/S)
3.1 MANUFACTURE PHASE			
Production Prepare Cost (Fixed)	Cpp	4.000 DM	4.000 DM
Bare Board Cost	Cbb	100 DM	100 DM
Component Cost / Board	Cc	5.000 DM	5.700 DM
Number of Components:			
Axial	nax	20	20
Radial	nrad	5	5
ICs	nic	20	20
SMDs	nsmd	35	35
Manual	nman	20	20
Others	noth	0	0
Assembly Cost:			
Axial	Caax	0,10 DM	0,10 DM
Radial	Carad	0,10 DM	0,10 DM
ICs	Caic	0,15 DM	0,15 DM
SMDs	Casmd	0,10 DM	0,10 DM
Manual	Caman	0,40 DM	0,40 DM
Others	Caoth	0,10 DM	0,10 DM
Overall Assembly Cost / Board	Cass	17 DM	17 DM
Production Variable Cost/per Board	Bvar	5.117 DM	5.817 DM
Tot. Production Var. Cost	Pvar	26.614.800 DM	30.286.488 DM
Tot. Production Prep.(Fixed) Cost	Pfix	4.000 DM	4.000 DM
Total Production Cost (For TNPA)	TCNPB	26.618.800 DM	30.290.488 DM
Production Unit Cost for Boards going to Systems Assembly	PUSA	5.324 DM	6.058 DM
3.2 TEST PHASE			
Number of Shifts	NS	1	
Max. Prod. Vol. / Year	PV	1500	
Failure rates:			
Shorts per 100 PBA	Fsh	7	7,1
Opens per 100 PBA	Fop	20	20,3
Wrong/Missing Compon. per 100 PBA	Fpla	15	15,2
Faulty An.& Dig.ICs	Fan	25	25,4
Dynamic faults	Fdig	1	1
Other Faults per 100 PBA	Foth	6,5	6,6
Total Faults/100 Boards	FPB	75	76
Production Yield	Yp	47%	47%
Mean time between failure (h)	MTBF	100000	100000

Test Stage Name		STRATEGY 1 (WITHO. B/S) VI	ICT	FUNCTIONAL	STRATEGY 2 (WITH B/S) VI	B/SCAN TEST	FUNCT.+B/S
3.2.1 QUALITY MODEL							
Number of Iterations in Test	nit	3	3	3	3	3	3
Shorts fault coverage	FCsh	5%	90%	100%	5%	90%	100%
Opens fault coverage	FCop	5%	90%	75%	5%	90%	75%
Wrong/Missing Comp. fault coverage	FCpla	80%	60%	50%	80%	60%	50%
Fault Cov. of An.& Dig.	FCan	0%	60%	65%	0%	60%	65%
Digital fault coverage	FCdig	0%	0%	0%	0%	0%	0%
Other	FCoth	10%	97%	100%	10%	97%	100%
Good Board Coverage	Eg	100%	100%	100%	100%	100%	100%
Shorts going to Test	USh	7,0	7,0	0,7	7,1	7,1	0.7
Opens going to Test	Uop	20,0	20,0	2.0	20.3	20.3	2.0
Wrong/miss. Comp.going to T.	Uwr	15,0	15,0	6,0	15.2	15.2	6.1
Faulty an.& dig. w/o BS gtT.	Uan	25.0	25,0	10.0	25,4	25,4	10.2
Faulty digit. going to Test	UnDig	1,0	1,0	1,0	1.0	1.0	1.0
Other faults going to Test	UnOth	6.5	6,5	0.2	6.6	6.6	0.2
Early life failures	elf	0,0	0,0	0.0	0.0	0,0	0.0
Total Faults going to test/100 PBA	Tfun	74.5	74.5	19.9	75.6	75.6	20.2
Detected Shorts	DetSh	0,00	6,30	0,00	0.00	6.39	0.00
Detected Opens	DetOp	0,00	18,00	0,00	0,00	18.27	0.00
Detected wrong/missing components	DetPla	0,00	9,00	0,00	0,00	9.12	0.00
Detected defective an.& dig. w/o BS	DetAn	0,00	15,00	0.00	0.00	15.24	0.00
Detected defective BS components	DetDig	0.00	0.00	0.00	0,00	0.00	0.00
Detected other	DetOth	0.00	6,31	0,00	0,00	6.40	0.00
Detected early life failures	DetElf	0,00	0,01	0,00	0,00	0.01	0.00
Total Detected Faults per 100 PBAs	ds	0,0	54,8	0.0	0,0	55,4	0.0
Yield after test	Yat	47%	82%	82%	47%	82%	82%
Number of good boards failed	Ngf	0	0	0	0	0	0
Test Iteration Factor	Tit	25%	43%	15%	25%	43%	15%
Max. Number of b. going to test / Y	Ntest	1991	2543	1759	1990	2554	1762
Max.Num of b going to diag./rep. / Y	Ndr	491	1043	259	490	1054	262
Number of non-repairable boards / Y	Nnr	6	52		6	53	
Total Number of B. going to Test	TNtest	5902	3819	5101	5907	3867	5113
Total Num.of B going to Diag./Rep.	TNdr	1701	3618	900	1700	3660	911
Total Number of non-rep. Boards	TNnr	20	179	2	20	184	[illegible]
Total Number of Good Boards failed	TNgf	0	0	0	0	0	0

Copyright Siemens-Nixdorf-Informationssysteme

3.2.2 TEST TIME MODEL							
Lot Size	Lot	1	25	1	1	25	1
Acceleration Fact.(accel.life test)	A	0	1	0	1	1	1
Set up time/lot (minutes)	Tsu	0	20	0	0	20	20
Attended Test Time for Good B.(min)	Tatt	0	5	0	0	5	0
Unattended test time (minutes)	Tun	0	1	0	0	1	0
Diagnosis Time per Fault (minutes)	Tdiag	0	2	0	0	2	0
Repair Time per Fault (minutes)	Trep	0	20	0	0	20	0
Accumulated test time (incl. Accel.)	Tacc	0	6	6	0	6	6
Diag. Time for Good B. failed (min)	Tdgf	0	0	0	0	0	0
Effective Test Time for Good Board (Minutes)	Teff	0,00	6,80	0,00	0,00	6,90	20,00
Operational test time (min)	Top	0	10	0	0	10	0
Diagnosis & Repair time / Board	Td	0,00	22,00	0,00	0,00	22,00	0,00
Total Test Time/Yr. (hours)	TTY	0	288	0	0	290	0
Testers Required	TER	0,00	1,00	0,00	0,00	1,00	0,00
Total Diag/Rep Time per year (h)	Tdryr	0	383	0	0	387	0
Diag/Rep Stations required	TRER	0,00	1,00	0,00	0,00	1,00	0,00

3.2.3 TEST COST MODEL							
Hourly Test Labour Rate	Kt	0 DM	75 DM	0 DM	0 DM	50 DM	0 DM
Hourly Diagnosis Labour Rate	Kd	0 DM	75 DM	0 DM	0 DM	50 DM	0 DM
Hourly Var. Operating Cost per ATE	KATE	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Hourly Var. Operating Cost per DRE	KDRE	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Labour Cost of Testing / Board	Ct	0 DM	7 DM	0 DM	0 DM	5 DM	0 DM
Labour Cost of Diagnosis / Board	Cd	0 DM	28 DM	0 DM	0 DM	18 DM	0 DM
Labour Cost of Diag./Good B.failed	Cdgf	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Test,Diagnosis,R. Labour Costs	PRC	0 DM	163.438 DM	0 DM	0 DM	109.955 DM	0 DM
Tot. Preparation Cost per T.Stage	SpreC	0 DM	30.000 DM	0 DM	0 DM	5.000 DM	0 DM
Purchase Cost for ATE	PATE	0 DM	350.000 DM	0 DM	0 DM	20.000 DM	0 DM
Fixed Operating Costs /Y	OfATE	0 DM	10.000 DM	0 DM	0 DM	1.500 DM	0 DM
Depreciation Cost /Y for ATE	DcATE	0 DM	58.333 DM	0 DM	0 DM	3.333 DM	0 DM
Interest Cost /Y for ATE	IcATE	0 DM	21.000 DM	0 DM	0 DM	1.200 DM	0 DM
Total LC Fixed Costs for 1 ATE (During Systems Production Time)	TATE	0 DM	536.000 DM	0 DM	0 DM	36.200 DM	0 DM
Purchase Cost for Diag.Rep.Eq.(DRE)	PDRE	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Other Fixed Operating Costs /Y	OfDRE	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Depreciation Cost /Y for DRE	DcDRE	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Interest Cost /Y for DRE	IcDRE	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Total LC Fixed Costs for 1 D.R.Eq.	TDRE	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Equipment Fixed C. per Test Stage	Cpeq	0 DM	536.000 DM	0 DM	0 DM	36.200 DM	0 DM
Var. Oper. Costs for ATE's /Board	OvATE	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Var. Oper. Costs for DRE's /Board	OvDRE	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
Var. Oper.Costs for DRE's /G.B.f.	OvDgbf	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM
ATE +DRE Var.Oper. Costs	OOVC	0 DM	0 DM	0 DM	0 DM	0 DM	0 DM

		STRATEGY 1 (WITHO. B/S)	STRATEGY 2 (WITH B/S)
Overall Labour T & R Cost	APTR	163.438 DM	109.955 DM
Total LC Var. Operating Costs	TOv	0 DM	0 DM
Total Fixed Test,D.,Rep. Eq. Cost	Cote	536.000 DM	36.200 DM
Total Test Preparation Cost	TpreC	30.000 DM	5.000 DM
Test-related Labour Cost per Board		32,69 DM	21,99 DM
Total Variable Operating Costs per Board		0,00 DM	0,00 DM
Total Fixed Test,D.R. Eq. Cost per Board		107,20 DM	7,24 DM
Cost of Preparation per Board		6,00 DM	1,00 DM
Operational test time (h)	optime	10,17	10,22
Remaining faults/100 PBAs:			
Shorts	remsh	0,70	0,71
Opens	remop	2,00	2,03
Wrong/missing components	rempla	6,00	6,08
Defective an.,dig. comp. w/o BS	reman	10,00	10,16
Defective BS components	remdig	1,00	1,00
Others	remoth	0,19	0,20
Total remaining faults / 100 PBA:	remsum	19,90	20,18
Remaining Faults per Board	frem	1,99E-01	2,02E-01

	Summary		
4. COST SUMMARIES FOR:	PB1		
4.1 DEVELOPEMENT COSTS			
Labour	dlab	463.232 DM	470.241 DM
Material	dmat	12.500 DM	14.250 DM
Equipment	dequ	0 DM	0 DM
Total Developement Costs	NRE	475.732 DM	484.491 DM
Developement beginning Year	DBY	0	0
Developement Period in Years	DP	1	1
Total Developement Costs at T=0	NRE0	424.761 DM	432.581 DM
4.2 PRODUCTION COSTS			
4.2.1 MANUFACTURING COSTS			
Tot. Production Prep.(Fixed) Cost	Pfi	4.000 DM	4.000 DM
Tot. Production Var. Cost	Pva	26.614.800 DM	30.286.488 DM
Manufacturing beginning Year	MBY	1	1
Manufacturing Period in Years	MP	5	5
Tot. Production Fixed Cost at T=0	Pfix0	3.571 DM	3.571 DM
Tot. Prod. Var. Cost at T=0	Pvar0	17.132.214 DM	19.495.716 DM
Tot. Prod. Fixed Cost per Board	Pfix	0,71 DM	0,71 DM
Tot. Prod. Var. Cost per Board	Pvar	3.426 DM	3.899 DM
Total Production Cost at T=0	tpc	17.135.785 DM	19.499.288 DM
4.2.2 TEST COST			
Test-related Labour Cost	tla	163.438 DM	109.955 DM
Variable Operating Costs	tva	0 DM	0 DM
Fixed Test,D/R. Eq. Cost	tfi	536.000 DM	36.200 DM
Test Equ. Preparation Cost	tpre	30.000 DM	5.000 DM
Total Test Cost	tbtc	729.438 DM	151.155 DM
Test beginning Year	TBY	1	1
Test Period in Years	TPY	5	5
Test-related Labour Cost at T=0	tlab0	105.207 DM	70.779 DM
Variable Operating Costs at T=0	tvar0	0 DM	0 DM
Fixed Test,D/R. Eq. Cost at T=0	tfix0	478.571 DM	32.321 DM
Test Equ. Preparation Cost at T=0	tprep0	19.311 DM	3.219 DM
Test-related Labour Cost /Board	tlab	21,04 DM	14,16 DM
Variable Operating Costs /Board	tvar	0,00 DM	0,00 DM
Fixed Test,D/R. Eq. Cost /Board	tfix	95,71 DM	6,46 DM
Test Equ. Preparation Cost /Board	tprep	3,86 DM	0,64 DM
Total Test Cost at T=0	ttc	603.090 DM	106.319 DM
Total Number of Boards going to System Assembly	TNSA	5000	5000
LIFE CYCLE BOARD COST	cv	18.163.635 DM	20.038.188 DM
LIFE CYCLE COST PER BOARD	LccpB	3.633 DM	4.008 DM

Produced per Year	nsysy:a	1500
Number of Systems in Life Cycle	nsys:a	5000
Production period	pper	5
Number of board types:	nbt:a	1

BOARD DATA

Board Names:		OTHERS	PB1 STRATEGIE 1 W/O B.SCAN	STRATEGIE 2 WITH B.SCAN
Strategie Plan	strpl:a	1	1	0
Number of Boards needed/System	npbs:a	0	1	
Total Number of Boards needed	tnpbs:a	0	5000	
Max. Number of boards per year	mnpb:a	0	1500	
Developement Cost	dc:a	0 DM	424.761 DM	432.581 DM
Manufacture Fixed Cost	mfc:a	0,00 DM	0,71 DM	0.71 DM
Manufacture Variable Cost	mvc:a	0,00 DM	3.426,44 DM	3.899,14 DM
Total Manufacture Cost	tmc:a	0 DM	17.135.785 DM	19.499.288 DM
Labour Test & Repair Cost	lab:a	0,00 DM	21,04 DM	14,16 DM
Total Test Preparation Cost	tpre:a	0,00 DM	3,86 DM	0,64 DM
Total Fixed Test, D.R. Eq. Cost	tfix:a	0,00 DM	95,71 DM	6,46 DM
Total Variable Operating Cost	tvar:a	0,00 DM	0,00 DM	0,00 DM
Total Test Cost	totc:a	0 DM	603.090 DM	106.319 DM
Remaining faults/PBA:				
Shorts	rsh:a	0,00E+00	7,00E−03	7,10E−03
Opens	roo:a	0,00E+00	2,00E−02	2,03E−02
Wrong/missing components	rwm:a	0,00E+00	6,00E−02	6,08E−02
Defective an./dig. comp. w/o BS	ran:a	0,00E+00	1,00E−01	1,02E−01
Dynamic faults	rdig:a	0,00E+00	1,00E−02	1,00E−02
Others	roth:a	0,00E+00	1,95E−03	1,98E−03
Test time (operation) of PBA (h)	ptim:a		10,17	10,22
OVERALL BOARD−TYPE COST	bov:a	0 DM	18.163.635 DM	20.038.198 DM

1. SYSTEMS TEST ENGINEERING PHASE		Development Cost		
		W/O B.SCAN	WITH B.SCAN	MIXED STRAT.
Length of Test Prog.(1000s lines)	spr:a	0	0	0
Fraction of Test Prog.gener.autom.	sfga:a	0%	0%	0%
Length of Tool for autom.Test.gen.(K lines)	stpr:a	0	0	0
Hourly Test Eng. Labour Rate	ttel:a	120 DM	120 DM	115 DM
Hourly Test Eng. Equipment Rate	ttee:a	0 DM	0 DM	0 DM
Syst.Test Preparation Cost	tpre:a	0 DM	0 DM	0 DM
Time for tools (h)	tsw:a	0	0	0
Time for Test Prog.(h)	ttp:a	1550	1300	0
Labour Cost for Test Engineering	ctel:a	186.000 DM	156.000 DM	0 DM
Equipment Cost for Test Eng.	ctee:a	0 DM	0 DM	0 DM
2. SYSTEM ASSEMBLY PHASE		Production Cost		
2.1 SYSTEM CONSTRUCTION				
Board Developement Cost	rreo:a	424.761 DM	432.581 DM	424.761 DM
Board Manufacture Fixed Cost	pfib:a	1 DM	1 DM	3.571 DM
Board Manufacture Var. Cost	pvab:a	17.132.214 DM	19.495.716 DM	17.132.214 DM
Total Board Manufacture Cost	ptob:a	17.132.214 DM	19.495.717 DM	17.135.785 DM
Board Labour Test & Rep. Cost	ctlabb:a	105.207 DM	70.779 DM	105.207 DM
Board Total Test Prep. Cost	SPREP::	19.311 DM	3.219 DM	19.311 DM
B. Total Fixed Test, D.R. Eq.Cost	STF:a	478.571 DM	32.321 DM	478.571 DM
Total Board Variable Oper. Cost	STV:a	0 DM	0 DM	0 DM
Total Board Test Cost	TSTC:a	603.090 DM	106.319 DM	603.090 DM
TOTAL BOARD COST AT T=0	TBC:a	18.160.065 DM	20.034.618 DM	18.163.635 DM
Build in Test Equip.Cost / System	BTeq:a	0 DM	0 DM	0 DM
Total Built in Equip. Cost	Tbteq:a	0 DM	0 DM	0 DM
2.2 SYSTEM TEST PHASE				
2.2.1 QUALITY MODEL				
Shorts	sh:a	7,00E-03	7,10E-03	7,00E-03
Opens	op:a	2,00E-02	2,03E-02	2,00E-02
Wrong/missing components	pla:a	6,00E-02	6,08E-02	6,00E-02
Defective an./dig. comp. w/o BS	an:a	1,00E-01	1,02E-01	1,00E-01
Dynamic faults	dig:a	1,00E-02	1,00E-02	1,00E-02
Others	oth:a	1,95E-03	1,98E-03	1,95E-03
Mean Time Between Failure (h)	MTBF:a	5000	5000	10000000
Total number of faults	tnf:a	1,99E-01	2,02E-01	1,99E-01
FC Shorts	fcsh:a	100%	100%	100%
FC Opens	fcop:a	100%	100%	100%
FC Wrong/missing components	fcpla:a	100%	100%	100%
FC Defective an./dig. comp. w/o BS	fcan:a	100%	100%	100%
FC Dynamic faults	fcdig:a	100%	100%	100%
FC Others	fcoth:a	100%	100%	100%
Detected faults	det:a	1,99E-01	2,02E-01	1,99E-01
Number of Systems going to Test	NSt:a	6005	6019	5995
Number of Systems going to D./Rep.	NSdr:a	1005	1019	995
Max. Num.Sys.going to Test / Year	NStY:a	1802	1806	1798
Max. Num.Sys.going to D.R. / Year	NSdrY:a	302	306	298
2.2.2 TEST TIME MODEL				
Number of Shifts	Nsn:a	1	1	[illegible]
Time at board test (h)	tb:a	10,17	10,22	10.17
Attended Test Time (min)	Tatts:a	7	7	7
Unattended Test Time (min)	Tuns:a	2	2	2
Diagnosis Time per Fault (min)	Tdiags:a	100	60	60
Repair Time per Fault (min)	Treps:a	0	0	0
Test Time per System (h)	TT:a	0,15	0,15	0.15
Diag.& Rep. Time per fault (h)	DT:a	1,67	1,00	1.00
Total Test Time (h)	TTT:a	901	903	899
Total Diag. Rep. Time (h)	TDT:a	1675	1019	995
Av. test time per system	att:a	0,18	0,18	0.18
Av. diag./repair time per system	adt:a	0,34	0,20	0.20
Max. Test Time per Year (h)	TTTY:a	270	271	270
Max. Diag. & Rep.Time per Year (h)	TDTY:a	503	306	298
System Testers Required	STRE:a	1	1	[illegible]
Diagnose & Rep. Stations Reqired	STDRE::	1	1	[illegible]

2.2.3 TEST COST MODEL

Hourly Test Labour Rate	SKt:a	70 DM	70 DM	67 DM
Hourly Diagnosis & Rep.Labour Rate	SKd:a	70 DM	70 DM	87 DM
Hourly Var. Operating Rate per Ate	SKATE:a	0 DM	0 DM	0 DM
Hourly Var. Operating Rate per DRE	SKDRE::	0 DM	0 DM	0 DM
Labour Cost of Testing / System	SCt:a	8 DM	8 DM	8 DM
Labour Cost of Diagnosis / System	SCd:a	117 DM	70 DM	67 DM
Test, Diagnosis,R. Labour Costs	TDRC:a	166.306 DM	120.510 DM	113.507 DM
Purchase Cost for Systems ATE	PSATE:a	0 DM	0 DM	0 DM
Other Fixed Operating Costs/Y	OfATE:a	0 DM	0 DM	0 DM
Depreciation Cost/Y for ATE	DcATE:a	0 DM	0 DM	0 DM
Interest Cost/Y for ATE	IcATE:a	0 DM	0 DM	0 DM
Total LC Fixed Cost for 1 ATE	TATE:a	0 DM	0 DM	0 DM
Purchase Cost for Systems DRE	PSDRE::	0 DM	0 DM	0 DM
Other Fixed Operating Costs/Y	OfDRE:a	0 DM	0 DM	0 DM
Depreciation Cost/Y for DRE	DcDRE:a	0 DM	0 DM	0 DM
Interest Cost/Y for DRE	IcDRE:a	0 DM	0 DM	0 DM
Total LC Fixed Cost for 1 DRE	TDRE:a	0 DM	0 DM	0 DM
Equipment Fixed Cost per Strategie	EfCS:a	0 DM	0 DM	0 DM
Systems Test Preparation Cost	STprC:a	0 DM	0 DM	0 DM
Var Oper.Costs for ATE's /System	CVATE:a	0 DM	0 DM	0 DM
Var.Oper.Costs for DRE's /System	CvDRE::	0 DM	0 DM	0 DM
Equipment Variable Cost /Strategie	OOVC:a	0 DM	0 DM	0 DM

Field Operation Cost

3. FIELD OPERATION PHASE

Mean Time Between Failure (h)	SMTBF::	5000	5000	10000000
Undet.Faults from System Test	Ufst:a	0,00E+00	0,00E+00	0,00E+00
Undet. to MTBF factor	u2mfact	0,1	0,1	0,1
MTBF incl. undetected faults	FMTBF:a	5000	5000	10000000

3.1 INSTALLATION

Operation Time at Inst.(h)	Topl:a	5	5	5
Diagnosis Time at Installation (h)	STdins:a	0	0	0
Hourly Inst. Labour Cost Rate	Lcr:a	150 DM	150 DM	200 DM
Field Diagnosis Cost /Fault	FDcr:a	0 DM	0 DM	0 DM
Operation Start Time at Inst.(h)	Tbegl:a	10,69	10,60	10,55
Operation End Time at Inst. (h)	Tendl:a	15,69	15,60	15,55
Prob. of Early Life Failure at Ins.	Pefl:a	9,97E-04	9,97E-04	5,00E-07
Installation Cost per System	SSAT:a	0 DM	0 DM	0 DM
Tot. Test Time at inst. per System	TTtins:a	5,005	5,005	5,000
Total Diagnosis Time at Instalation	TDtins:a	0,000	0,000	0,000
Probability of Field Repair	Probl:a	0%	0%	0%
Num.of Comp.,B. going to Serv.Ctr.	Nscl:a	5	5	0
Cost of Installation per System	CloS:a	751 DM	751 DM	1.000 DM
TOTAL INSTALLATION COST	TIC:a	3.753.740 DM	3.753.740 DM	5.000.002 DM

3.2 FIELD TEST

System Time Operation at Field (h)	TopF:a	3720	3720	3720
Operation Start Time at Field (h)	TbegF:a	15,69	15,60	15,55
Operation End Time at Field (h)	TendF:a	3735,69	3735,60	3735,55
Prob. of Early Life Failure at F. (Undetected + New Failures, + Probability of a Fail.occurrence)	PefF:a	5.23E-01	5.23E-01	3.72E-04

3.2.1 FIELD REPAIR

Probability of Field Repair	ProbF:a	0%	80%	0%
Downtime with Repl. U. Avail.(h)	DTwuf:a	4	4	2
Downtime without Repl. U. Avail.(h)	DTwouf:	8	8	4
Systems Downtime Cost per Hour	DTCh:a	0 DM	0 DM	0 DM
Hourly Diag.Rep. Labour Rate	Kfl:a	150 DM	150 DM	150 DM
Hourly Field Equipment Rate	Kfdre:a	0 DM	0 DM	0 DM
Average Material cost per repair	rmatc:a	100 DM	100 DM	100 DM
Total number of breakdowns	nbr:a	2616	2616	2
Number of repairable Faults in F.	Nrff:a	0	2093	0
Num. of Faults going to Service C.	Nscf:a	2616	523	2
TOTAL FIELD COST	FTRC:a	1.569.440 DM	1.778.729 DM	558 DM

3.2.2 SERVICE CENTER DIAGNOSIS AND REPAIR

Smallest Replaceble Field Unit (1=Component,2=Board)	SRFU:a	2	2	2
Number of Fail. going to S.Center	Nfsc:a	2621	528	2
Reparable Fail. at S.Center in %	Probsc:a	0%	0%	0%
Number of repairable B. at Serv.C.	Nbrsc:a	0	0	0
Ave.Shipping Time from/to Field (h)	Tship1:a	2	2	3
Hourly Shipping Cost Rate	Kshl:a	100 DM	100 DM	100 DM
Field/Serv.center shipping cost	sh-c:a	1.048.288 DM	211.257 DM	1.117 DM
Ave.Time taken for Diag.R.per B.(h)	Tsd:a	0	0	0
Hourly Diag.Rep. Labour Rate	Ksl:a	0 DM	0 DM	0 DM
Hourly S.Center Equipment Rate	Kseq:a	0 DM	0 DM	0 DM
Service center rep. cost	scrc:a	0 DM	0 DM	0 DM
Average Time for Repair(h)	avtr:a	309	65	304
Stock safety Factor	Stsf:a	2	2	2
Failures during a repair cycle:				
Failed systems in installation	Fsl:a	5	5	0
Non-working units	FsO:a	309	65	0
Number of B./C. needed (+safety F.)	NBn:a	627	141	0
Aver. Price for Board	PBoard:a	5.324 DM	6.058 DM	0 DM
Aver. Price for Component	PComp:a	100 DM	100 DM	100 DM
Stock and Interest Rate for Trb (Y)	Slr:a	30%	30%	30%
Storage Cost in the Life C.	TSC:a	5.008.057 DM	1.278.861 DM	0 DM
TOTAL SERVICE CENTER COST	SCC:a	6.056.345 DM	1.490.118 DM	1.117 DM

3.2.3 DEPOT REPAIR

Num. of Boards going to Depot	Nbd:a	2621	528	2
Rep. Failures at Depot in %	Probd:a	100%	100%	40%
Number of Repairable Boards	Nrd:a	2621	528	1
Ave.Time taken for Diag.R.per B.(h)	Tdd:a	1	0,67	0,8
Ave.Shipping Time from/to S.Center	Tship2:a	150	150	148
Hourly Shipping Cost Rate	Kdhl:a	0 DM	0 DM	0 DM
Hourly Diag.Rep. Labour Rate	Kdl:a	75 DM	50 DM	67 DM
Hourly Depot Equipment Rate	Kdeq:a	50 DM	0 DM	0 DM
Number of non reparable Boards	Nnonr:a	0	0	1
Ave. Cost of Lost Boards	Clb:a	0 DM	0 DM	0 DM
TOTAL DEPOT COST	TDC:a	589.562 DM	70.507 DM	114 DM

		Summary		
4. COST SUMMARIES AT T=0				
4.1 SYSTEMS TEST ENGINEERING PHASE				
Labour Cost for Test Engineering	SCta:a	166.000 DM	156.000 DM	0 DM
Cost for Test Eng.Equip.	SCteq:a	0 DM	0 DM	0 DM
Test Eng. beginning Year	TEBY:a	1	1	1
Test Eng. Period in Years	TEP:a	1	1	1
Labour Cost for Test Eng. at T=0	Lcte:a	148.278 DM	124.362 DM	0 DM
Cost for Test Eng.Equip. at T=0	Ecte:a	0 DM	0 DM	0 DM
TOTAL TEST ENGINEERING COST AT T=0	TTEC:a	148.278 DM	124.362 DM	0 DM
4.2 SYSTEM ASSEMBLY PHASE				
4.2.1 SYSTEM CONSTRUCTION				
Total Built in Equip. Cost	Tbteq:a	0 DM	0 DM	0 DM
Manufacturing beginning Year	MBY:a	1	1	1
Manufacturing Period in Years	MP:a	5	5	5
T.Built in Equipment Cost at T=0	Tbtec:a	0 DM	0 DM	0 DM
Total Board Life-Cycle Cost at T=0	TBC:a	18.160.065 DM	20.034.818 DM	18.163.635 DM
TOTAL CONSTRUCTION COST AT T=0	TCC:a	18.160.065 DM	20.034.818 DM	18.163.635 DM
4.2.2 SYSTEM TEST				
Systems Test Preparation Cost	STprC:a	0 DM	0 DM	0 DM
Test, Diagnosis,R. Labour Costs	TDRC:a	166.306 DM	120.510 DM	113.507 DM
Equipment Fixed Cost per Strategie	EfCS:a	0 DM	0 DM	0 DM
Equipment Variable Cost /Strategie	OOVC:a	0 DM	0 DM	0 DM
System Test beginning Year	STby:a	1	1	1
System Test period in Years	STP:a	5	5	5
S. Test Preparation Cost at T=0	STPR0:a	0 DM	0 DM	0 DM
Test, Diagnosis,R. Lab. C. at T=0	TDRC0:a	107.053 DM	77.573 DM	73.068 DM
Eq. Fixed C. per Strategie at T=0	EfCS0:a	0 DM	0 DM	0 DM
Eq. Var. Cost /Strategie at T=0	OOVC0:	0 DM	0 DM	0 DM
TOTAL TEST COST at T=0	TTC:a	107.053 DM	77.573 DM	73.068 DM
4.3 FIELD OPERATION PHASE				
Total Installation Cost	TIC:a	3.753.740 DM	3.753.740 DM	5.000.002 DM
Total Field cost	FTRC:a	1.569.440 DM	1.778.729 DM	558 DM
Total Service Center Cost	SCC:a	6.056.345 DM	1.490.118 DM	1.117 DM
Total Depot Cost	TDC:a	589.662 DM	70.507 DM	114 DM
Total Field cost	tfc:a	11.969.187 DM	7.093.095 DM	5.001.792 DM
Field operation beginning Year	Foby:a	1	1	1
Field operation Period	FOP:a	7	7	7
Total Installation Cost at T=0	TIC0:a	2.185.096 DM	2.185.096 DM	2.910.560 DM
Total Field cost at T=0	FTRC0:a	913.589 DM	1.035.419 DM	325 DM
Total Service Center Cost at T=0	SCC0:a	3.525.470 DM	867.415 DM	650 DM
Total Depot Cost at T=0	TDC0:a	343.249 DM	41.043 DM	67 DM
TOTAL FIELD OPERATION COST	TFOC:a	6.967.405 DM	4.128.974 DM	2.911.602 DM
SYSTEM LIFE CYCLE COST	SLCC:a	25.382.800 DM	24.365.527 DM	21.148.303 DM
LCC per System	LCCS:a	5.077 DM	4.873 DM	4.230 DM

Appendix D

Description of Computer Board Used for ECOvbs

******* DATA OF BOARD >>>vbs_board<<< ******

DFT types: ict, nodft,

** DESIGN EFFORTS (in weeks)**

DESIGN	VERIFICATION	LAYOUT	PROTOYPE	COMPONENTS
115.0	54.0	60.0	40.0	52.0

** ITERATION FACTORS**

DESIGN	VERIFICATION	LAYOUT	PROTOYPE
0.7	0.5	0.8	0.9

Number of pre/post iterations: 1.5 / 1.2

Number of prototypes / material cost per prototype: 3 / 80000.00

** PRODUCTION DATA **

Expected production volume: 5000

Production prepare cost: 5300.00

Solder join repair cost: 2.00

Component replacement cost: 5.00

Number of solder joins: 15000

Defect rate of solder joins: 25.00

Defect rate of pick&place: 300.00

** DEFECT SPECTRUM **

digital : 0 dpm

analog : 150000 dpm

passive : 0 dpm

board : 30000 dpm

edge_con : 105000 dpm

pla : 0 dpm

ram : 14000 dpm

rom : 0 dpm

micro : 0 dpm

asic : 80000 dpm

res : 40000 dpm

cap : 10000 dpm

solder : 381100 dpm

pick_and_place : 321600 dpm

*************** TEST CLUSTER LIST (3) *******************

Cluster1: ram256k, ram64k, ram1m,

Cluster2: vlsi,

Cluster3: resi1, capa1, deskew, edge, pcb,

*********** TEST COMPLEXITY ***************

Combinational design: 20.00%

Pipeline structure: 40.00%

Synchronous design: 40.00%

Asynchronous design: 0.00%

******* DATA OF COMPONENT >>>vlsi<<< ******

Number of elements: 40

Component type: asic

Mount type: smd

*************** >>DFT ALTERNATIVES<< (3) ********************

DFT TYPE . PRICE DPM RATE

COMPLEXITY

nodft, 300.00 2000

60000 gates, 300 pins, 52.0 weeks des. eff.

bound_scan, 330.00 2200

64000 gates, 304 pins, 54.0 weeks des. eff.

bound_scan, selftest, 370.00 2300

70000 gates, 304 pins, 57.0 weeks des. eff.

******* DATA OF COMPONENT >>>ram64k<<< ******

Number of elements: 4

Component type: ram

Mount type: smd

*************** >>DFT ALTERNATIVES<< (2) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 5.00 500

65536 gates, 19 pins, 0.0 weeks des. eff.

bound_scan, selftest, 6.00 500

65536 gates, 23 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>ram256k<<< ******

Number of elements: 8

Component type: ram

Mount type: smd

*************** >>DFT ALTERNATIVES<< (2) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 15.00 500

256000 gates, 21 pins, 0.0 weeks des. eff.

bound_scan, selftest, 17.00 500

256000 gates, 25 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>ram1m<<< ******

Number of elements: 16

Component type: ram

Mount type: smd

*************** >>DFT ALTERNATIVES<< (2) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 60.00 500

1000000 gates, 25 pins, 0.0 weeks des. eff.

bound_scan, selftest, 65.00 500

1000000 gates, 29 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>resi1<<< ******

Number of elements: 800

Component type: res

Mount type: axial

*************** >>DFT ALTERNATIVES<< (1) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 2.00 50

0 gates, 2 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>capa1<<< ******

Number of elements: 100

Component type: cap

Mount type: axial

*************** >>DFT ALTERNATIVES<< (2) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 5.00 10

0 gates, 2 pins, 0.0 weeks des. eff.

nodft, 2.00 100

0 gates, 2 pins. 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>deskew<<< ******

Number of elements: 100

Component type: analog

Mount type: smd

*************** >>DFT ALTERNATIVES<< (1) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft. 15.00 1500

0 gates. 2 pins. 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>edge<<< ******

Number of elements: 3

Component type: edge_con

Mount type: smd

*************** >>DFT ALTERNATIVES<< (2) ********************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 12.00 35000

200 pins

bound_scan, 14.00 35000

204 pins

******* DATA OF COMPONENT >>>pcb<<< ******

Number of elements: 1

Component type: board

Mount type: b_board

*************** >>DFT ALTERNATIVES<< (3) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 200.00 30000

0 tpads, 5000 nodes, 12 layers, 2 sides, 0.0500 wire sep. (mm)

ict, 220.00 30200

5000 tpads, 5000 nodes, 12 layers, 2 sides, 0.0500 wire sep. (mm)

bound_scan, 220.00 30500

0 tpads, 5004 nodes, 12 layers, 2 sides, 0.0500 wire sep. (mm)

******* DATA OF BOARD >>>vbs_board<<< ******

DFT types: bound_scan, board_st, nodft,

** DESIGN EFFORTS (in weeks)**

DESIGN	VERIFICATION	LAYOUT	PROTOYPE	COMPONENTS
120.0	56.0	61.0	40.0	76.0

** ITERATION FACTORS**

DESIGN	VERIFICATION	LAYOUT	PROTOYPE
0.7	0.5	0.8	0.9

Number of pre/post iterations: 1.5 / 1.2

Number of prototypes / material cost per prototype: 3 / 81000.00

** PRODUCTION DATA **

Expected production volume: 5000

Production prepare cost: 5300.00

Solder join repair cost: 2.00

Component replacement cost: 5.00

Number of solder joins: 15100

Defect rate of solder joins: 25.00

Defect rate of pick&place: 300.00

** DEFECT SPECTRUM **

digital : 0 dpm

analog : 150000 dpm

passive : 0 dpm

board : 30000 dpm

edge_con : 105000 dpm

pla : 0 dpm

ram : 0 dpm

rom : 0 dpm

micro : 0 dpm

asic : 0 dpm

res : 40000 dpm

cap : 1000 dpm

solder : 65000 dpm

pick_and_place : 301200 dpm

*************** TEST CLUSTER LIST (3) *******************

Cluster1: ram256k, ram64k, ram1m,

Cluster2: bsc, vlsi,

Cluster3: resi1, capa1, deskew, edge, pcb,

*********** TEST COMPLEXITY ***************

Combinational design: 20.00%

Pipeline structure: 40.00%

Synchronous design: 40.00%

Asynchronous design: 0.00%

******* DATA OF COMPONENT >>>vlsi<<< ******

Number of elements: 40

Component type: asic

Mount type: smd

*************** >>DFT ALTERNATIVES<< (3) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft. 300.00 2000

60000 gates. 300 pins. 52.0 weeks des. eff.

bound_scan. 330.00 2200

64000 gates. 304 pins. 54.0 weeks des. eff.

bound_scan. selftest. 370.00 2300

70000 gates, 304 pins, 57.0 weeks des. eff.

******* DATA OF COMPONENT >>>ram64k<<< ******

Number of elements: 4

Component type: ram

Mount type: smd

*************** >>DFT ALTERNATIVES<< (2) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 5.00 500

65536 gates, 19 pins, 0.0 weeks des. eff.

bound_scan, selftest, 6.00 500

65536 gates, 23 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>ram256k<<< ******

Number of elements: 8

Component type: ram

Mount type: smd

*************** >>DFT ALTERNATIVES<< (2) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 15.00 500

256000 gates, 21 pins, 0.0 weeks des. eff.

bound_scan, selftest, 17.00 500

256000 gates, 25 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>ram1m<<< ******

Number of elements: 16

Component type: ram

Mount type: smd

*************** >>DFT ALTERNATIVES<< (2) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 60.00 500

1000000 gates, 25 pins, 0.0 weeks des. eff.

bound_scan, selftest, 65.00 500

1000000 gates, 29 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>resi1<<< ******

Number of elements: 800

Component type: res

Mount type: axial

*************** >>DFT ALTERNATIVES<< (1) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft. 2.00 50

0 gates, 2 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>capa1<<< ******

Number of elements: 100

Component type: cap

Mount type: axial

*************** >>DFT ALTERNATIVES<< (2) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 5.00 10

0 gates, 2 pins, 0.0 weeks des. eff.

nodft, 2.00 100

0 gates, 2 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>deskew<<< ******

Number of elements: 100

Component type: analog

Mount type: smd

*************** >>DFT ALTERNATIVES<< (1) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 15.00 1500

0 gates, 2 pins, 0.0 weeks des. eff.

******* DATA OF COMPONENT >>>edge<<< ******

Number of elements: 3

Component type: edge_con

Mount type: smd

*************** >>DFT ALTERNATIVES<< (2) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft. 12.00 35000

200 pins

bound_scan. 14.00 35000

204 pins

******* DATA OF COMPONENT >>>pcb<<< ******

Number of elements: 1

Component type: board

Mount type: b_board

*************** >>DFT ALTERNATIVES<< (3) ********************

DFT TYPE PRICE DPM RATE

COMPLEXITY

nodft, 200.00 30000

0 tpads, 5000 nodes, 12 layers, 2 sides, 0.0500 wire sep. (mm)

ict, 220.00 30200

5000 tpads, 5000 nodes, 12 layers, 2 sides, 0.0500 wire sep. (mm)

bound_scan, 220.00 30500

0 tpads, 5004 nodes, 12 layers, 2 sides, 0.0500 wire sep. (mm)

******* DATA OF COMPONENT >>>bsc<<< ******

Number of elements: 1

Component type: asic

Mount type: smd

*************** >>DFT ALTERNATIVES<< (1) *******************

DFT TYPE PRICE DPM RATE

COMPLEXITY

bound_scan, selftest, ict, nodft, 130.00 2000

6400 gates, 104 pins, 24.0 weeks des. eff

Planning Testable VLSI Design Under Economic Aspects

Jochen Dick
Siemens AG
DI AP 22
Postfach 700079
D-8000 München 70

0. Abstract

This paper presents a concept of planning testable designs for VLSI circuits. The approach is based upon a test economics model. A test planning support system will be introduced. This system aims at identifying the cost-optimal test method for a specified VLSI design.

1. Introduction

The increasing integration level of ASIC's lead inevitably to embedded circuits. For testing purposes, problems occur with low accessibility. Test cost increase as accessibility to the test device decreases. So people started to develop methods for increasing the accessibility only for testing purposes. These methods, called "Design-for-Testability" methods, have the aim to increase the accessibility of the circuit, in order to reduce the test cost and test generation cost, by adding extra logic into the circuit. But the disadvantage of the hardware overhead must be weighted against the advantage of more testability. The most suitable DFT method for a specific design must be found. We chose the cost as weighting unit, because it is probably the only unit, to which all advantages and disadvantages can be transformed. Also, the minimisation of the overall cost is probably the best way to make a product successful. Beside weighting the cost, the requirements coming from the circuit specification must be considered for choosing a test method.

To predict the arising cost, a test economics model was developed [Dic89b]. The model considers all costs, which arise for VLSI design and production, and which are influenced by DFT. Based on the test economics model, a test planning support system (TPSS) will be developed as a design tool. It aims at supporting VLSI designers for planning testable designs under economic aspects. It should be used during the design specification and accompanying the design-entry phase.

2. A Test Economics Model

The test economics model developed is a parameterised model. It enables the prediction of all cost influenced by the test strategy. The model fits for the development of cell-based ASIC's. By cell-based ASIC's we mean semi-custom ASIC's, which are developed by the usage of a cell library. The model development was based on initial studies [Dic89a, Dea89, Dis89]. The cost forming the model are divided into 4 parts as proposed in [Dic89a].

1) Production cost :

Cost of the VLSI supplier. These cost are not calculated by a model, because the user of the model (this is the designer) can not influence the cost by controlling the production process. He has to pay the price, which is offered by the supplier, no matter what he would calculate by a cost model. Nevertheless the price is influenced by characteristics of the design, especially the gate number.

2) Design cost :

Cost related to the design and development of cell-based ASIC's. They include engineering cost, the cost for using computers and the charges for using a outside design center.

3) Test cost :

Cost related to testing purposes. These include the cost for fault simulation and test pattern generation. ATE cost as part of the production are only considered, if the ASIC price is influenced by the size of the test set. Cost for an incoming test are usually related to board cost.

4) Time-to-Market cost :

Cost related to reduced production volume and penalty for non-performance caused by a development-schedule slippage.

3. Concept of A Test Planning Support System

The Test Planning Support System (TPSS) aims at supporting a VLSI designer for planning testable designs under economic aspects. This system should enable to find cost-optimal solutions of designing VLSI's. The system is based on the parametrised test economics model, which allows to predict the cost impact of applying DFT methods. To use the model, information about the design, the DFT methods under consideration and the design environment is needed.
Using the TPSS we assume a 2-step method of designing for testability. This method follows strongly the style of designing VLSI's :

- In the first step, the circuit must be partitioned into testable units (TU). This partitioning process is driven by design partitioning aspects, the size and the accessibility of the TU's. A TU is identified by 2 attributes : The internal DFT method for a TU is homogeneous, and the unit is directly accessible, which

means, that every i/o of the TU can be controlled/observed independently through a specified function (i.e. scan path).

- In the second step, the testability of the TU's itself must be planned (internal DFT). This step aims at ensuring a required fault coverage by applying a DFT method to the TU. Some DFT methods fit only for specific designs (i.e. scan path requires synchronous design), and therefore a decision on DFT must be made quite often in the specification phase.

The TPSS should be used for both steps. As well the partitioning process as the decision on DFT should be supported by the TPSS. In addition, the TPSS should enable the economic evaluation of new DFT methods or special methods.

3.1 Test Partitioning

Hierarchical design and a mixture of random logic and regular structures on one VLSI are state-of-the-art design techniques. For testing, we should take advantage from these partitioning techniques, because the test methods for regular structures usually differ from the test methods for random logic, and partitioning for testing reduces the complexity for testing in the same way as it reduces the complexity for the design process (principle of "divide and conquer").The goal of test partitioning is to derive testable units (TU's), which can be tested independently. To achieve this goal, the embedded TU must be directly accessible for testing purposes. There are two cases to fulfil this requirement :

1) The inputs/outputs of the TU are directly accessible through circuit i/o's or can be made accessible through a user-defined function.

2) The inputs/outputs of the TU are not directly accessible. Then the circuit must be provided with extra test logic to make the i/o's directly accessible.

For partitioning the random logic, the overhead, which is caused in case 2, must be weighted against the advantage achieved for testing. This weighting will be done by using the economics model, in order to derive a cost-optimal test partitioning solution. The partitioning process of the circuit is also driven by applying different test methods. In the same way we can use the cost model to derive the cost-optimal solution (what is cheaper : partitioning the circuit to apply the cost-optimal test method to every block, or using a unified, none-optimal test method and saving the extra cost for partitioning ?). Basis for the test partitioning is the design partitioning. TU's are composed of design blocks, so that the TU-architecture can be easily extracted from the hierarchical netlist.

3.2 Support of DFT Method Selections

Once the design is partitioned into TU's, and the accessibility of the TU's is assured, a test method must be derived for every TU. This process will also be supported by using the test economics model. The test method should enable to derive a test set, which achieves the required fault coverage. The test method

includes as well the DFT method (in this case "internal DFT") as the methods for test pattern generation. For some DFT methods (especially structured DFT methods), the application must be considered from the beginning. Structured DFT methods require a synchronous design. If the designer does not intend to develop the design in the required style, the decision, whether to use a structured DFT method or not, must be made during the specification phase. If the requirements on the clock logic are not fulfilled, the application of structured DFT methods after having designed the circuit would probably lead to a complete redesign of the whole logic. The decision on which method to use, can be made after the logic design. But the designed circuit must be prepared for applying structured DFT methods. So we have two decision phases for the internal DFT. In the first phase, which is part of the design specification phase, we have to plan, whether to apply structured DFT or not. In the second phase we plan, which DFT method to apply for the TU's. This planning can be done after the logic design-entry. The advantage is, that test planning based on the test economics model is more accurate, if we have detailed information about the design. This information does not exist before having designed the circuit on logic level. Another advantage of test planning after the logic design-entry is, that the information, which is needed for using the test economics model, can be extracted automatically from the netlist.

4. Conclusions

A concept for planning testable designs under economic aspects was presented. A test planning support system was presented, which will give advice based upon the economic implications of any design decision.

The test economics model presented here considers only the cost concerning the VLSI design and production. But for some DFT methods, the costs of higher levels of assembly are of significance (i.e. boundary scan). In order to enable test planning for boards and systems, we will extend the test economics model for predicting the cost implications for systems.

5. References

Dea89 : Dear, Dislis, Lau, Miles, Ambler :
Hierarchical Testability Measurement and Design For Test Selection by Cost Prediction.
Proc. ETC 1989

Dic89a Dick :
ESPRIT-2318 (EVEREST) : Activity Report
No. SIE X0004 AD MP

Dic89b Dick :
ESPRIT-2318 (EVEREST) : Activity Report
No. SIE 0008 GL TN

Dis89 Dislis, Dear, Miles, Lau, Ambler :
Cost Analysis of Test Method Environment
Proc. ITC 1989

An Economics Based Test Strategy Planner for VLSI Design

C. Dislis**, J. Dick*, A. P. Ambler**

* SIEMENS-Nixdorff-Informationssysteme, 8000 Munich 83, Germany
** Brunel University, Department of Electrical Engineering and Electronics, Uxbridge, Middlesex, UK
(under contract to SIEMENS-Nixdorff)

Abstract

The problem of making informed testability choices for a partitioned design is addressed in this paper, and an industrial software tool to aid the decision making process is described.The system uses an economics model to evaluate test methods, in combination with a set of user defined limits. A degree of automatic test strategy selection is also incorporated.

1. Introduction

This paper describes the design and implementation of a test strategy planning system developed jointly by SIEMENS and Brunel University under the ESPRIT EVEREST project. The system was designed for use in an industrial environment and is intended to aid the designer in making informed decisions about suitable DFT methods to be used in the design. The aim is not only to make the circuit testable, but also to do this in an economically optimal way. In order to achieve this, the effects of design and test decisions are evaluated using an economics model, specifically developed by SIEMENS for semicustom gate array design. Previous economics modelling work has shown that cost modelling is a viable method for test strategy selection [Dislis89, Dear88]. This paper will describe the structure of the test strategy planning system, the methods used to describe and categorise DFT methods and the use of cost modelling techniques for DFT decision making. The application of these methods to an example circuit will be discussed. The use of economics as the deciding factor in test strategy selection removes much of the uncertainty associated with it, as a large degree of subjectivity in assessing the relative importance of decision variables is replaced by objective financial measures.

It is now widely accepted that the route to testable VLSI circuits lies in making testability choices during the design process. There is a wide range of design for testability methods for a designer to choose from, each with its own advantages and drawbacks.The problem of which combination of methods to choose for a given application however, is not a trivial one, and has been addressed in the past [Abadir89]. If the right testability decisions are made in the early stages of the design process, a significant amount of redesign may be avoided, and a higher quality of product achieved. The system described here is unique in that it uses economic data to guide the decision making process. The final decision will depend on a large number of parameters, some of which may fall outside the immediate interests of the designer. The designer must also have in-depth knowledge of a large number of DFT techniques, which may not always be the case. An automatic system which formalises this process and lets the designer make informed test decisions by using stored knowledge and evaluation techniques is therefore likely to fulfil a need in this area. This is the aim of the Test Strategy Planning System described here. In an industrial environment, project decisions are driven by economic considerations. The cost of testability choices can be predicted by evaluating a cost model.

2. The Structure of the Test Strategy Planning System.

Figure 1 shows the outline of the test strategy planning system. The design description is acquired either directly from the user or from an existing netlist, and is built in a hierarchical fashion in order to allow test strategy decisions to be made at several stages of the design process. A variety of essential economic data is also acquired at the same time.

TEST STRATEGY PLANNER STRUCTURE

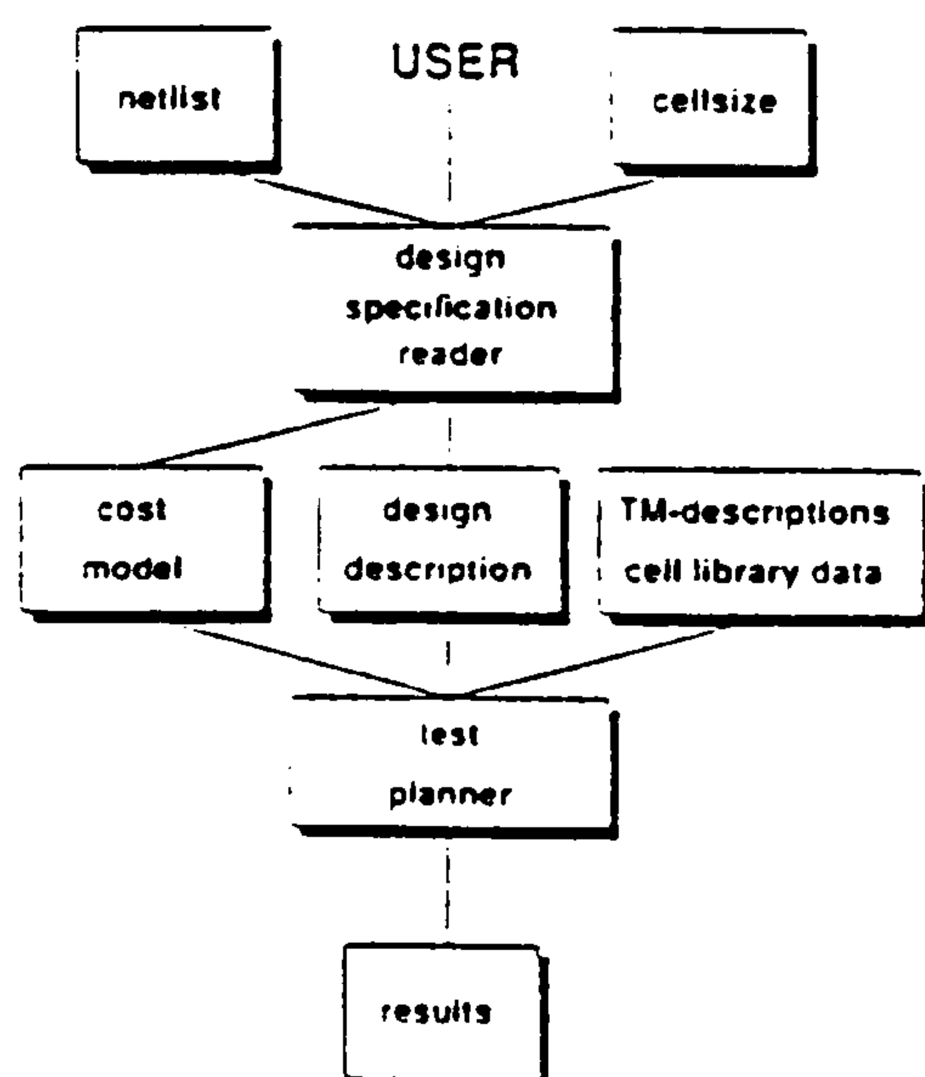


Figure 1: System architecture

OVERALL COST		
Design Cost	Engineering cost	Design Complexity
		Productivity of Design Environment
	Equipment Cost	
	Design Centre Cost	
Production Cost	Production Unit Cost	
	NRE Charges	
Test Cost	ATPG Cost	
	Manual TPG Cost	
	Test Application Cost	

Table I

The data is used to automatically update an economics model prior to the test strategy selection process. The test strategy planner uses stored knowledge on test methods as well as the design data and the existing cost model to evaluate a variety of test strategies. The test strategy control can be left entirely to the user, or alternatively, a degree of automatic planning may be employed to accelerate the selection.

3. Use of Cost Modelling Techniques

The economics model used was based on previous economics modelling work [Varma84, Dislis89], but was tailored to the needs of semicustom cell design. The cost model categorises primary parameters supplied by the user into design independent (mostly global company costings), design dependent (which will change with the design), test method dependent, and constant factors. A set of equations is then calculated using the supplied values. Table I illustrates the main parts of the model.

Design costs are modelled in terms of and equipment required, the cost of using an external design centre if this option is taken, and manpower, which is a function of the complexity of the system, and the productivity of the design environment. Productivity is modelled in terms of the designer's experience, the performance of the CAD system, and the functionality of the cell library.

Production costs for are linked to the design complexity (measured in gate equivalents, grids or mm^2), mostly with a linear relationship within certain complexity ranges. In this case, gate equivalent count was used as a complexity measure. A test strategy may influence the gate count and therefore the production cost per unit. In addition, non recurrent engineering (NRE) charges must be added to the production cost. Unlike previous models [Varma84, Dislis89], yield effects are not modelled separately, as they are included in the pricing policy of the vendor.

Test costs are separated into test pattern generation and test application costs. Test application costs are

linked to the number of test patterns in linear or stepwise ranges, similar to the way gate count is linked to production cost. This pricing is normally provided by the vendor, and removes the need for the separate modelling of ATE costs present in previous models. Test pattern generation costs are estimated as follows: the cost of automatic test pattern generation is estimated, together with the maximum achievable fault cover. If the achievable fault cover is less than the required value (a fundamental requirement) then the cost of achieving it using expensive manual tpg is estimated. Using structured DFT methods often means that ATPG techniques alone are effective for producing the required fault cover.

The model is coded in a manner which makes it easy for the user to alter. This may be done in two ways: during the test strategy planning process, the user can examine and alter individual parameters to evaluate their effects. For more permanent changes (for example to reflect a change in the vendor's pricing) the 'generic' (or template) cost model files are stored as text, so it is a simple matter to make changes using a text editor.

Group 1: parameters which are used by the cost model for an economic evaluation.	
Equivalent gate count	number of extra functions
sequential depth	number of test patterns
performance complexity	achievable fault cover
additional pin count	TPG method
originality	pin compatibility (possible shared use of test pins)
Group 2: Design implications.	
accessibility impact	self test / no self test
test method type (testability/accessibility)	
Group 3: Design Requirements.	
suitable design class (eg PLA, RAM)	suitable design style (eg synchronous, flip flop design, latch design)

Table II

4. Test Method Categorisation and Description

The test method description database needs to take account mainly of the economic effect of test methods, so that an economic evaluation may be produced, but also of the suitability of test methods for the particular design so that fundamental requirements (in terms of performance, maximum pin count etc) are still met. Parameters are described either as single values or text strings, or as equations. The parameters to be described are categorised in three groups, and are shown in table II.

The first group of parameters are used in the cost model for an economic evaluation of the method. The performance complexity and the originality parameters are related to design costs, as is the number of extra functions (if any) introduces by the test structures. The pin compatibility information is used in the calculation of the final pin overhead of the plan. In the design implications group of parameters, the test method type has to be specified. At the end of the test strategy planning process, a test strategy must make every block testable (on its own), and every block i/o line accessible, so that test patterns may be propagated to it. This is a requirement of the system. The test methods are therefore categorised into testability enhancing (internal) and accessibility enhancing external). For example, some self test methods make a block testable but not accessible, and methods such as scan path on the i/o lines only, provide accessibility but do not improve the actual testability of the block. Note that there is some overlap in this categorisation, as some testability enhancing test methods (eg certain scan options) will also provide accessibility to the block. The impact on the accessibility of the block i/o lines is also stored here, as it is useful in guiding the test selection algorithms. The third group contains the design requirements of the test method. For a test method to be applicable to a block, it has to be suited to the type and style of the design and also

needs to fulfil basic requirements in terms of fault cover, maximum gate count and pin count. Data in this section is used to check that design requirements are met.

4.1 Test Method Description - an example

The following section provides an example of the coding of the test methods description. The method shown here is scan path for enhancing the testability of random sequential designs. It does not necessarily improve the accessibility to the i/o lines of the block. The information is summarised in table III.

Test Method Name	int_scan
Test Method Type (ext./int.)	internal
Suitable Design Classes	random sequential
Self Test	no
Assures data-in accessibility	no
Assures control-in accessibility	no
Assures clock-in accessibility	no
Assures out accessibility	no
Assures bus accessibility	no
Performance implication	1.1
Test pattern generation method	Combinatorial ATPG
Sequential depth	0
Achievable fault coverage	calculated by the cost model
Number of testpatterns	calculated by the cost model
overhead_formula	2.5 * gate count of 1 DFF
originality impact	1
inpin overhead	2
outpin overhead	1
bipin overhead	0
pin compatibility class	1
Design Style Requirements	synchronous flip-flop design
Number of functions	0

Table III

4.2 Test Method Application

When a test method is applied to a block, its design and economic effects are evaluated. The test method is first checked for compatibility with the design type and style, and for falling within user defined limits. If this check is successful, the economic effects are evaluated by using the test method description parameters for the cost model, taking into account any cell library support for the method, and also the possibility of testing a number of functional blocks in parallel. Once the method is applied, its effects on the accessibility of the block are also noted, and the accessibility of other block i/o is recalculated. The recalculation is based on existing information of transparent 'paths' through functional blocks. Thus, and by the use of the cost model, global, as well as local implications of the test method are assessed. Testability and accessibility enhancing methods are applied separately. Changes to the design description are not permanent, but are current only for the duration of the test strategy planning process. It is up to the designer to implement the test structures suggested by the test strategy planner.

5. The Test Strategy Planning Process

The system provides the necessary functions for the user to evaluate a variety of test strategies, and also to use a degree of automated planning. The cost implications can be examined by examining the cost model. At the end of the process, a test strategy description is produced, with details of the test method chosen for every block, together with a complete copy of the cost model for later reference. The design description itself is not altered.

5.1 Manual Test Strategy Planning

Within the test strategy planning package, the user has the following choice of commands with which to create and evaluate test strategies for the design.

apply	apply test method to a given block
auto_plan	automatic test strategy planning
access	show line accessibility
show_tp	show test strategy
show_ci	show circuit information
stack	place cost model values on stack
report	print cost data of the test strategies on the stack
dump_cm	print evaluated cost model
file 'filename'	store cost model to specified file
new 'parameter' = value	evaluate the cost model using the new value for the specified parameter
tell 'parameter'	print the value of the specified parameter for test strategies on the stack
expand 'parameter'	expand a parameter to its primary parameters
table	create a table of the values of one parameter based on variations of another. Used for graphical output.
q	quit

Table IV

The 'apply' command allows the user to apply individual test methods to a block. One testability and one accessibility method may be applied to a block at any time. Automatic test strategy planning (auto_plan) is described in detail in section 5.2. The accessibility of testable units (TUs) is made visible through the 'access' function by displaying all TU-connections, which are not controllable or observable by either primary i/o's, the transparency of neighbouring TUs or an applied test method. The function is based upon the design data and a pre-calculation of the accessibility implications. This command is provided to inform the user about parts of the design where the accessibility is limited, in order to target these parts for further investigation. Accessibility is not given as a value, but as a yes/no flag, and may take the form of accessibility from the circuit's primary inputs("controllability"), or accessibility to the circuit's primary outputs ("observability").

The rest of the commands allow the user to examine the test strategy in detail, by being able to show circuit information (show_ci), as well as the chosen methods for each block (show_tp), print the evaluated cost model to the screen (dump_cm) or to a file (file), and examine the value of single parameters (tell). It is possible to compare up to five strategies by placing them on a stack (stack). The cost model data for all strategies on the stack can be examined, either collectively (report), or individually (tell). It is also possible to assign new values to individual parameters (new 'parameter'='value), or alter the value of a parameter through a specified range and observe the result on another parameter graphically (table). An example of this would be when the user would like to evaluate the effect that production volume would have on the relative costs of design, manufacture and test.It is also possible to find out which factors a parameter is dependent on, by using the 'expand' function.

5.2 Automatic Test Strategy Planning

The test strategy planning process can be automated, in order to save time or even provide a suitable starting point for further refinement of the plan. The aim is to find the cheapest combination of test methods which will make the design both testable and accessible, while staying within user defined limits. One algorithm is currently implemented in the system, and further improvements are planned. The order in which blocks are targeted for test method application is important. This is because once a method is chosen for a block it remains active (ie it is not re-evaluated) while methods for subsequent blocks are assessed. One of the aims of the algorithm chosen is to avoid backtracking, while incorporating some

optimisation into the plan.

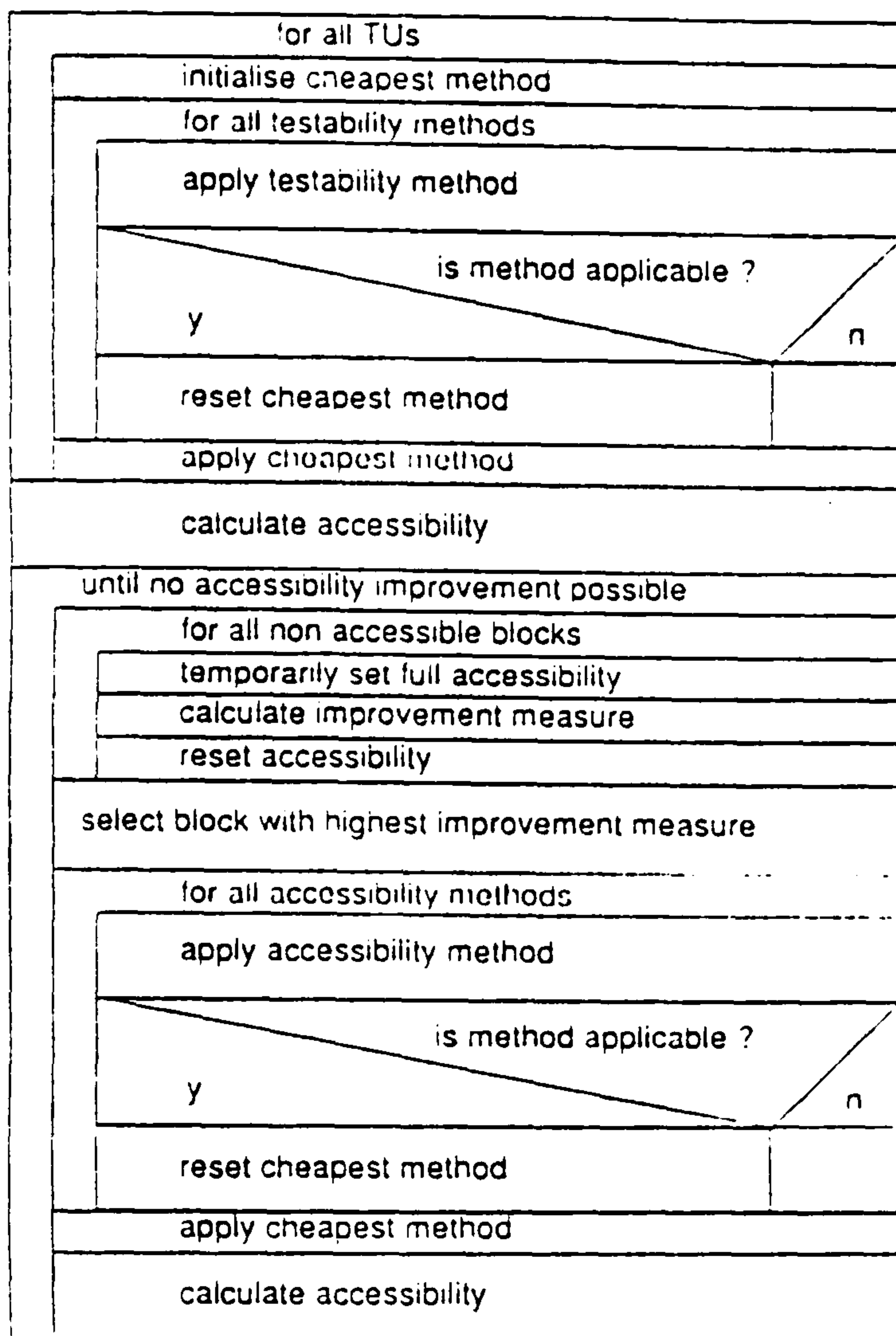


Figure 2: Automatic Test Strategy Planning

The automatic test strategy planning algorithm supplied with the system is summarised in figure 2 and works on the following principles: The first requirement is to make every block testable. This means that the block would be easily testable if it were tested in isolation, with perfect access to all its i/o lines. On this basis, a cost optimal testability method is chosen for every block in the circuit, by evaluating all suitable methods for each. At the end of this process, every block should be testable, but the accessibility of some blocks might also have increased.

The next requirement is one of accessibility to all i/o lines of every block, in order to be able to propagate the test patterns required. In order to determine where the accessibility problems are situated, the accessibility of i/o line groups is established. This information is used to determine whether any accessibility test methods need to be applied. This may or may not be the case, as many testability enhancing methods have the side effect of improving accessibility as well. There may be several candidates for the application of accessibility methods, and the aim of the algorithm is to find the block which has the maximum impact on the accessibility of the circuit, taking into account existing (already applied) test methods and the transparencies of paths through functional blocks. The improvement to the overall accessibility of the design is assessed by temporarily making each non accessible block fully accessible and propagating this effect in order to evaluate its impact on the accessibility of the whole circuit. Each block is given a rating on that basis, and the highest rated block is chosen. Suitable accessibility test methods are then evaluated, and the cost optimal one is chosen. The accessibility evaluation is repeated, and the algorithm terminates when the requirement of full accessibility is met, or when no further improvements can be made. The system informs the user of the methods currently evaluated, as well of the progress of the block selection process. The final test plan can be examined, altered or updated like any other.

Possible improvements on this algorithm would include a method for assessing the order that testability enhancing methods are applied in. This could be targeted at breaking feedback paths as a first heuristic, with subsequent decisions being taken based on the number of 'paths' through the circuit that the particular block controls. The larger the number of paths, the greater the testability improvement on the overall design would be. A small amount of backtracking could also be used, although this would impair the speed of the system, and its advantages and disadvantages still need to be evaluated.

6. Test Strategy Planning - an Example.

Figure 3 shows a circuit example which is used to illustrate the process.

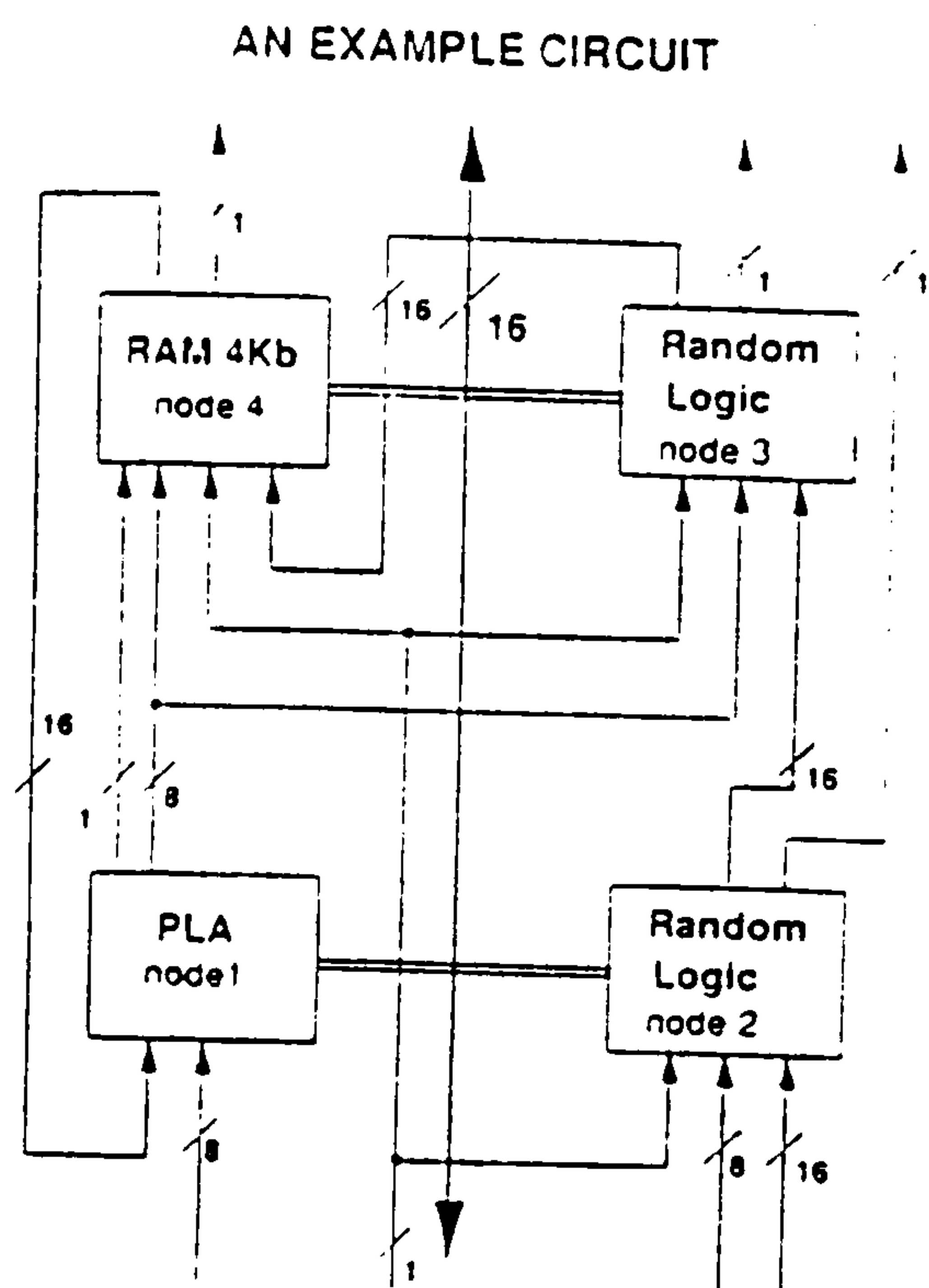


Figure 3: Circuit example

The design description is fully hierarchical down to gate level, but here only two levels of hierarchy are used: the overall circuit and the circuit sub blocks. These are described in terms of functionality (allowing suitable test methods to be chosen), as well as connectivity, with attributes to identify control, data and bus lines. In this case, the blocks are a PLA with 24 inputs, 9 outputs and 50 product terms, a 4Kb RAM and two sequential random logic blocks, of 5000 gates each. Information is also given on block transparency which allows the system to derive the accessibility of lines for test purposes. The objective of the automatic test strategy selection process is to ensure all blocks are individually testable and that all lines are accessible so that test patterns can be propagated to the testable blocks.

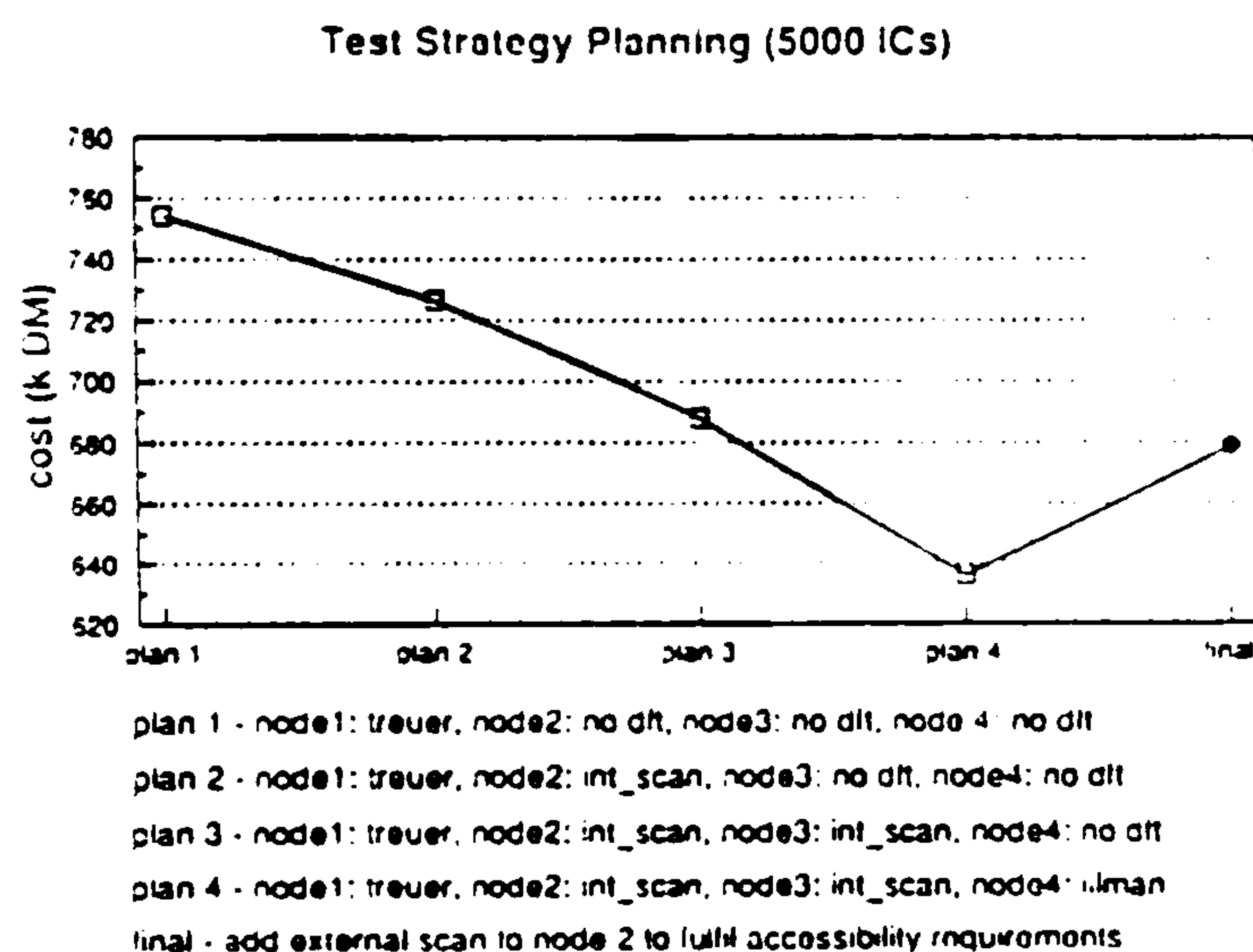


Figure 4: Test Strategy Selection for Production Volume of 5000 ICs.

Figure 4 shows the results of the test strategy plan at different stages for a production volume of 5000 units. In this case four PLA methods [Zhu88] were put through the cost model and the one chosen was a self test method using cumulative parity comparisons [Treuer85]. The RAM test methods included different implementations of standard memory test algorithms, as well as pseudo-random methods. The method chosen was an LFSR based pseudo-random method [Illman86]. Internal scan was used as the most cost effective method for making the sequential random blocks testable. An evaluation of the accessibility status after the inclusion of the testability enhancement methods revealed an accessibility problem on the group of lines connecting node 2 to node 3, and an external scan path, which is classed as an accessibility enhancing method was used to modify this. The accessibility method actually incurs a financial penalty, but this is due to the fact that accessibility improvement is a requirement of the algorithm and is not directly accounted for in the cost model. Cost calculations are made on the assumption that the blocks are fully accessible for test purposes. The financial penalty of adding an accessibility method is comparatively less for larger production volumes, as can be seen in figures 5 and 6. This is mainly

due to the fact that the relative importance of extra development costs decreases with increasing production volume. Figures 7 and 8 illustrate the cost breakdown of the latter two cases, indicating the relative importance of different cost areas.

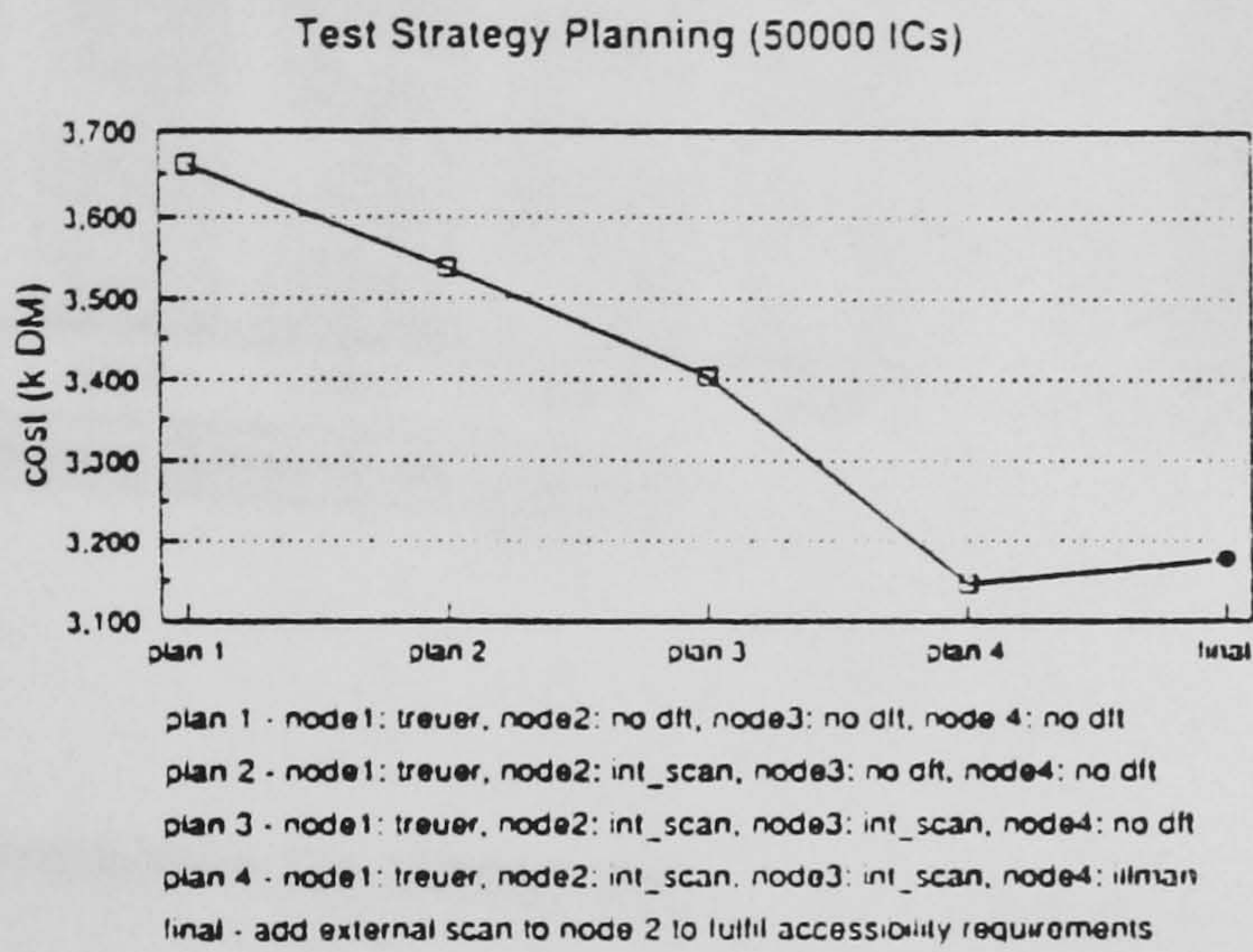


Figure 5: Test Strategy Selection for Production Volume of 50000 ICs.

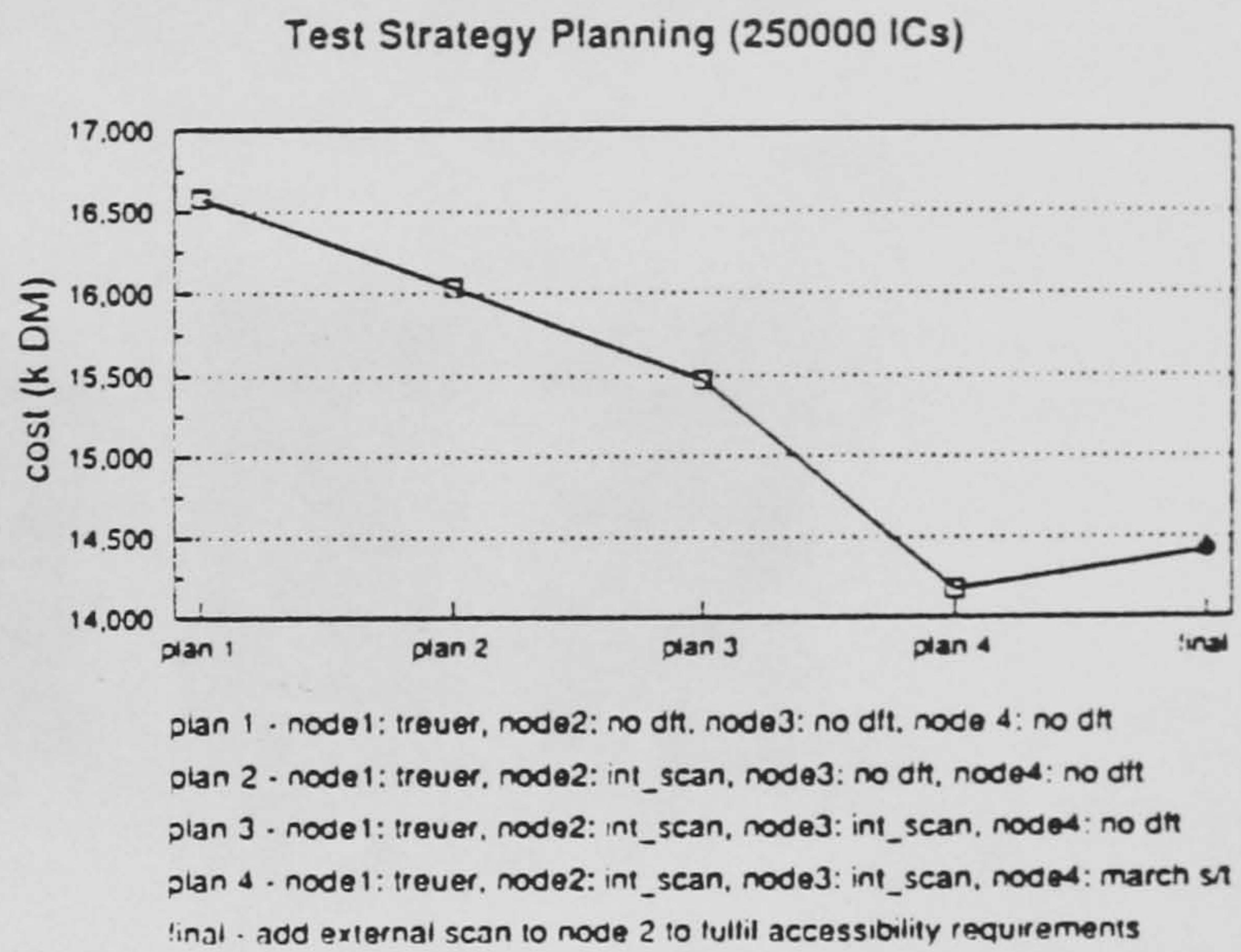


Figure 6: Test Strategy Selection for Production Volume of 250000 ICs.

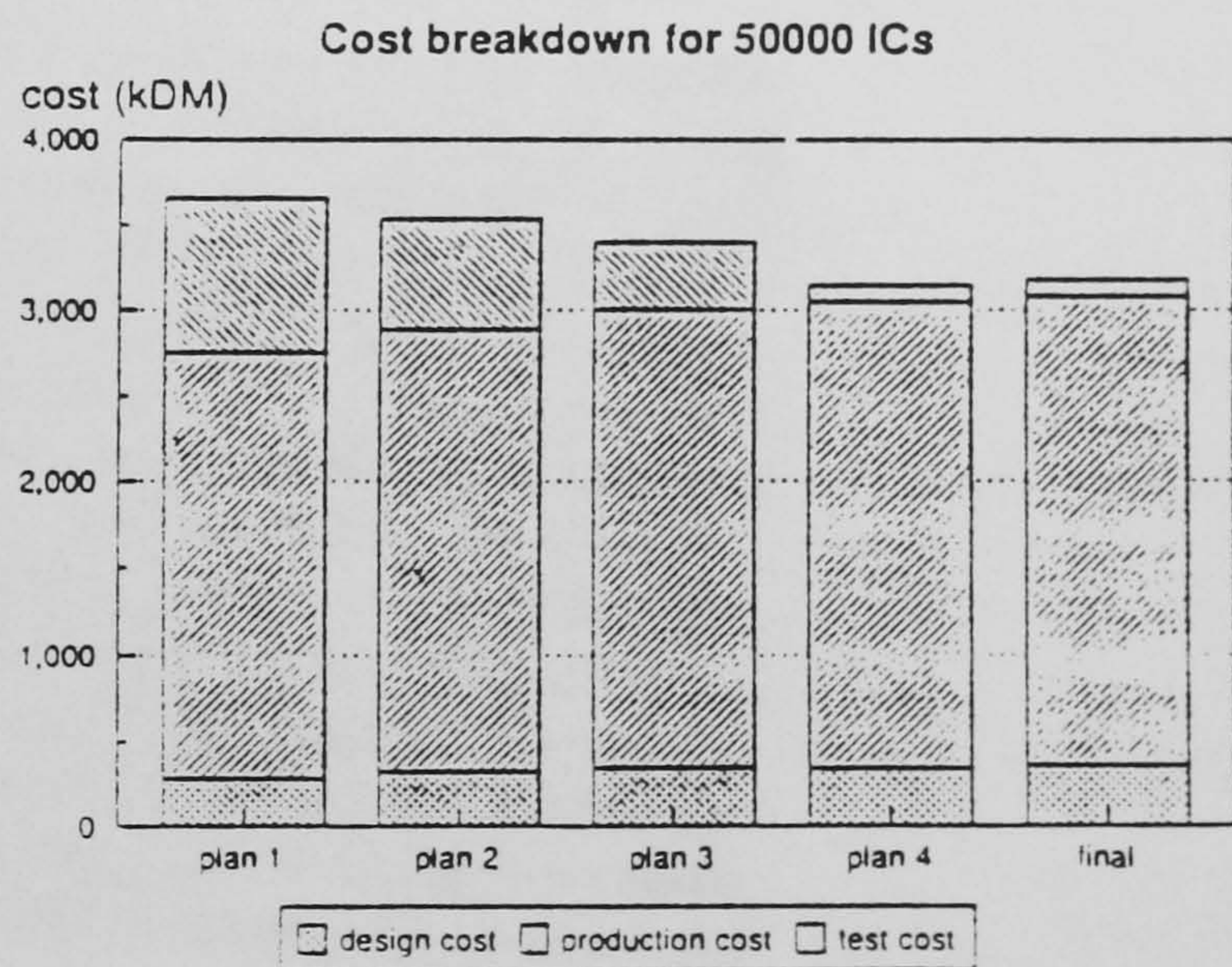


Figure 7: Cost breakdown for 50000 ICs.

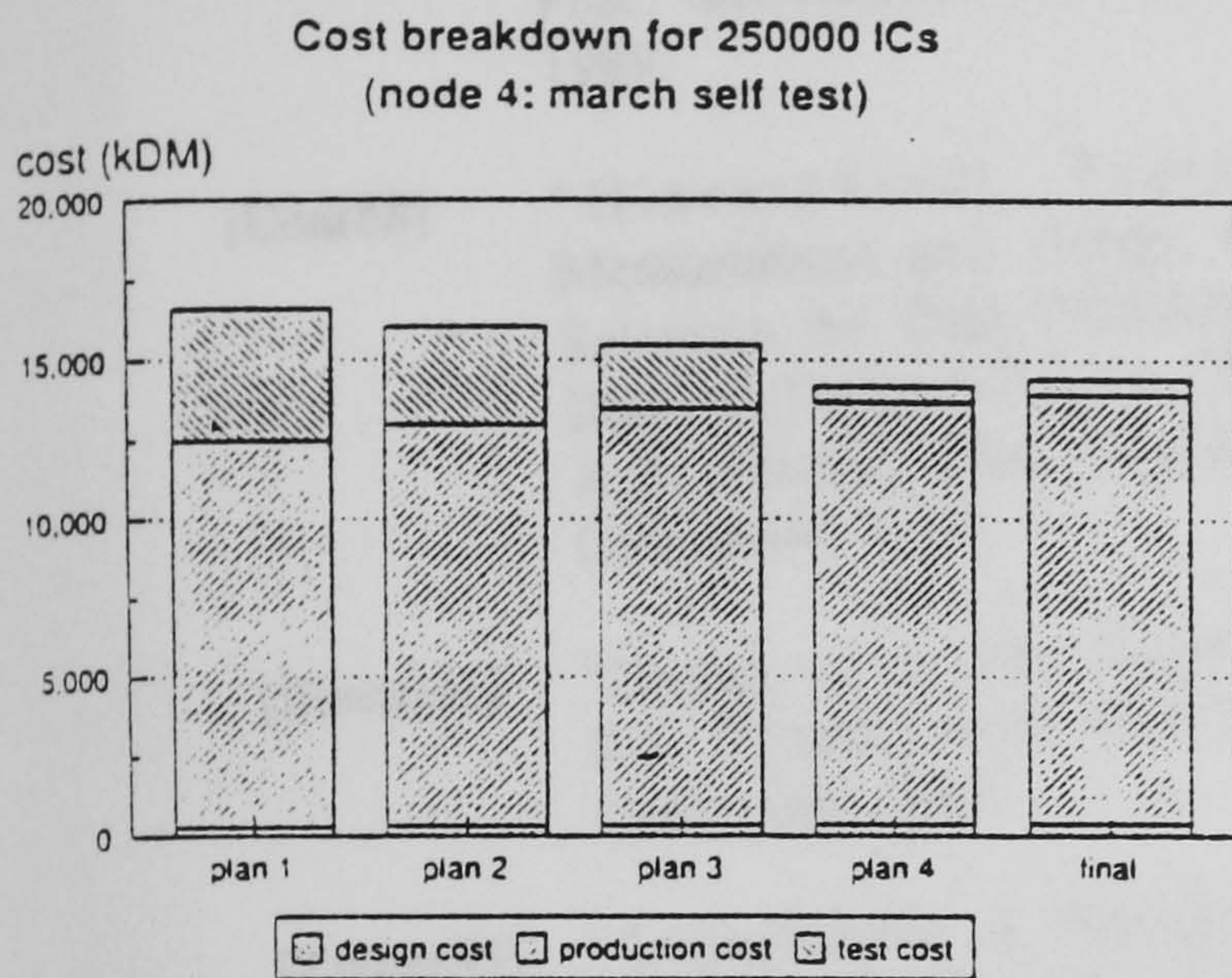


Figure 8: Cost breakdown for 250000 ICs.

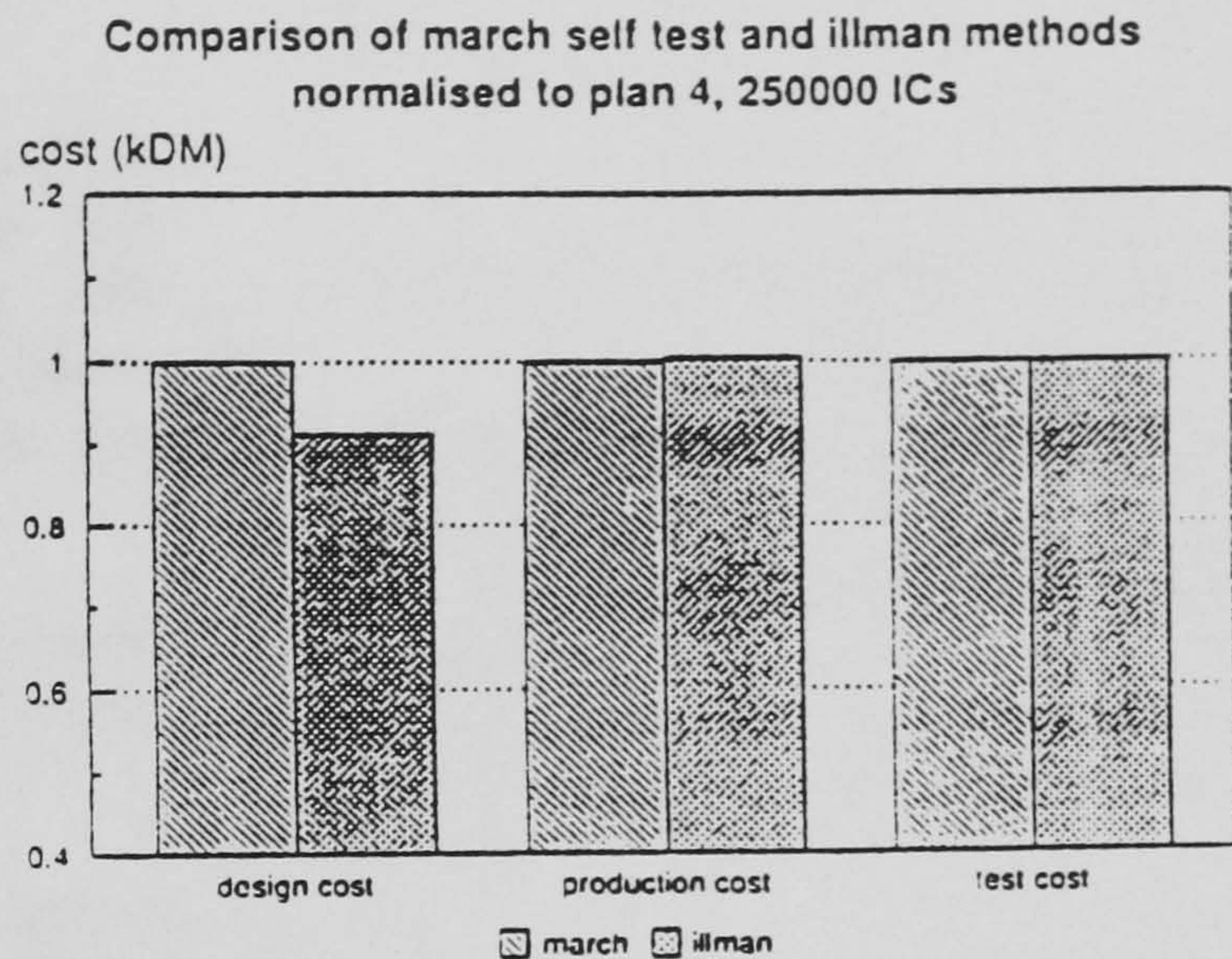


Figure 9: Comparison of memory test methods

In the case of the 250000 IC volume, the plan changes at stage 4, with a different test method for the memory element being chosen. In this instance, a self test implementation of the march algorithm was found more cost effective. Figure 9 shows the reasons the march method was chosen for the memory block, by comparing it to the previous plan. The results are normalised to the cost of the march method. Although the Illman method results in lower design costs, these are overtaken by the increase in production cost, which is small in percentage terms but results in a large real cost due to the increased manufacturing volume. It is possible therefore that for the same design, a variety of test method will be optimal, based on factors external to the design itself.

7. Conclusions

The system described here is still under evaluation, but we believe it is a useful tool for aiding designers in enhancing the testability of their designs. It was one of the aims of the system to make the selection of test methods speedy and easy by incorporating a measure of automatic planning, while still allowing full flexibility to the designer for evaluating test plans manually. The system is easily reconfigurable in terms of the cost modelling and test method description to suit different users. A full EDIF interface is also planned. The concept of test strategy planning is now being taken through to test strategy planning for VLSI based systems, based on the principles of economic evaluation and test strategy selection.

References

[Abadir89] "TIGER: Testability Insertion Guidance Expert System", M. Abadir, Proc. IEEE International Conference on Computer Aided Design, 1989.

[Dislis89] "Cost Analysis of Test Method Environments", C. Dislis, I.D. Dear, J.R. Miles, S.C. Lau, A.P. Ambler,

Proc. International Test Conference, 1989.

[Dear88] "Hierarchical Testability Measurement and Design for Test Selection by Cost Prediction", I.D. Dear, C. Dislis, S.C. Lau, J.R. Miles, A.P. Ambler, Proc. European Test Conference 1988.

[Illman86] "Design of a Self Testing RAM", R.J. Illman, Proc. Silicon Design Conference 1986.

[Treuer85] "Implementing a Built-In-Self-Test PLA Design", R. Treuer, H. Fujiwara, V.K. Agrawal, IEEE Design and Test of Computers, Vol. 2, April 1985.

[Varma84] "An Analysis of the Economics of Self Test", P. Varma, A. P. Ambler, Proc. IEEE International Test Conference, 1984.

[Zhu88] "Analysis of Testable PLA Designs", X. Zhu, M. A. Breuer, IEEE Design and Test, August 1988.

2.3

DOM: A defect occurence model for evaluating the life cycle costs of test strategies

Jochen Dick, Erwin Trischler and Anthony P. Ambler† – Siemens-Nixdorf-Informationssysteme AG, 8000 Munich 83, FRG. † Brunel University, Department of Electrical Engineering and Electronics, Uxbridge, UK.

The costs of testing VLSI based systems (VBSs) have reached an alarming size. In order to reduce these costs, the mix of test procedures (so called "the test strategies") is planned and optimised in the early stage of a product's life cycle. Today, this optimisation procedure is done locally for each life cycle phase. This paper will present a defect occurence model (DOM) as part of a test economics model, which allows to make a global optimisation of the test related costs.

2.3.1 Introduction

This chapter describes a defect occurence model (DOM) to calculate the impact of design, manufacture, test and operation/ageing on the defect distribution for VLSI based systems (VBSs) at each stage of the system's life cycle. Test costs are mainly a function of the quality, the complexity, the technology and the design style of the product and of the test strategy [Dis91]. The modification of e.g. the test strategy impacts the quality and yield of all following phases of the life cycle. So, the quality and therefore the test costs of a life cycle phase depends

The author(s) wish to acknowledge the support of ACM/SIGDA sponsorship in the organisation of the First International Workshop on the Economics of Design and Test

on the test procedure of the previous phases. Today test costs are typically optimised seperately for each life cycle phase and production stage. Instead of evaluating the test economics for various quality levels of intermediate production stages, quality requirements are fixed for each production stage. So, this is a local optimisation rather than a global optimisation. But the goal of testing should be to get a required field quality, achieved by the most economical mix of tests. DOM allows to evaluate the defect occurence during the entire life cycle of the production and thus to provide life-cycle-wide, economically optimised test strategies. In addition, different defect types are evaluated seperately. This differentation is needed due to the fact that different test methods exist to cover different defect types.

2.3.2 The Life Cycle Phases

To define the defect occurence for each phase of the life cycle of a VBS, the phases are classified by their influence on the defect occurence into the following classes:

- A **manufacture** process adds new defects.
- The **process control** reduces the addition of new defects during manufacture.
- **Ageing** adds new defects.
- **Test** decides on defective and non defective devices and can decrease the number of defects.
- **Operation** increases the number of defects and decides on defective and non defective devices.
- **Repair** eliminates defects, but may also cause new defects.

The diagnosis of defects is part of the process control phase (in order to optimise the process systematically) as well as the repair phase.

The life cycle of a VBS product can now be modeled as a chain of phases [Dav82]. The life cycle model for a specific product depends now on the structure of the production- and the test strategy, i.e. the combination of test phases and repair phases. Figure 1 shows the model of a typical board production as part of the life cycle.

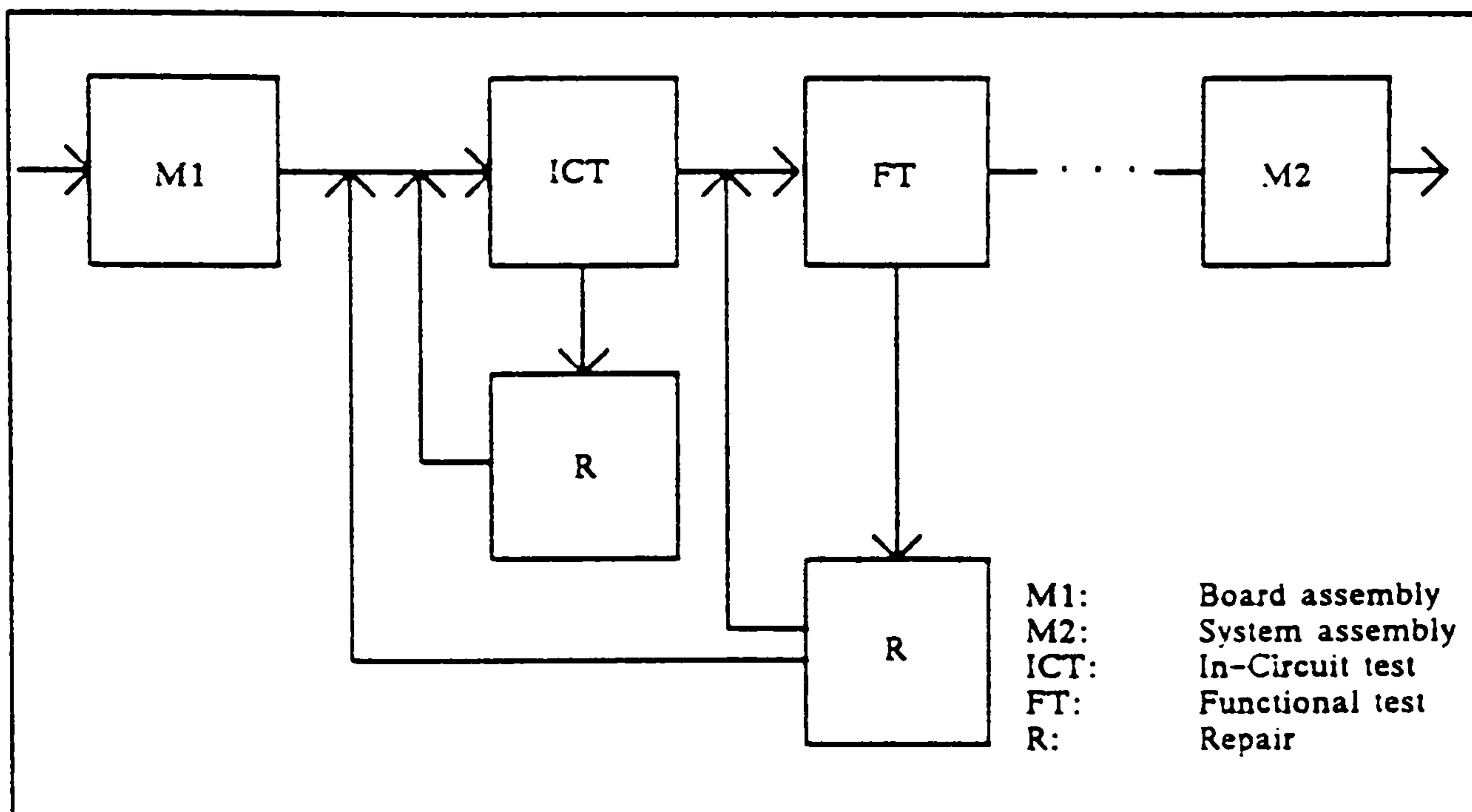


Figure 1: Phase model of a board production

2.3.3 The Defect Types

In the IC testing world a fault model is used to describe several defect types. Based on the defined fault model test patterns are generated and fault coverages are determined. The defect level of the final device is derived from the fault coverage and the yield of the IC production process[McC88]. A shortcome of using a single fault model is the neglect of the effect of defect types, which are not covered by the fault model, for deriving the quality level. For evaluating the defects over a product's life cycle, we need to distinguish between several defect types for the following reasons:

1) For assembled devices, there are no test methods which cover all defect types with a single test.
2) In the IC world, the fault coverage of e.g. a parametric test is not defined, because usually all possible tests are executed. This is not the case for the different faults of assembled devices.
3) In different phases of the product's life different defect types are relevant. To allow a cost evaluation for the entire life cycle, one must distinguish between the different defect types in different life cycle phases.

In Table 1 some typical defect types occuring after PCB assembly of computer boards are presented. In the right column test methods are listed which cover the accompanied defect type.

Defect Types	Test methods, which cover defect
Open in the edge connector area	Open-short test, boundary scan+self test
Open in the component area	In-circuit test, boundary scan
Open in the interconnection area	In-circuit test, boundary scan
Short in the edge connector area	Open-short test, boundary scan+self test
Short in the component area	In-circuit test, boundary scan
Short in the interconnection area	In-circuit test, boundary scan
Resistor defects	Open-short test, functional test
Static component faults	Functional test, boundary scan+self test
Dynamic component faults	Functional test, boundary scan+self test
Functional dynamic board faults	Functional test, self test
Calibration faults	Calibration test, self test
Memory faults	Memory test, self test
Parametric faults	Parametric test

Table 1: Defect types of computer boards and accompanied tests

2.3.4 The Defect Occurence Model (DOM)

DOM is a description of the average number of defects occuring after each phase of the life cycle and for each defect type. The average number of defects for each type is described by the vector

d_i: defect vector after phase i

This vector is modified by a test and repair through multiplication of the diagonal matrix T

T_i: test transparency of test in phase i

Each element in the matrix describes the test transparency [McC88] of the accompanied defect. Test transparency for a specific defect is defined as (1 - defect coverage) of the test. The defect vector after the test can be derived from

$$d_i = T_i \cdot d_{i-1} \quad (1)$$

where d_{i-1} is the defect vector of the incoming parts in phase i.

A phase, in which the number defects increases (e.g. manufacture), modifies the defect vector **d** through the addition of the new defects vector

nd_i: new defects vector in phase i

$$\mathbf{d}_i = \mathbf{d}_{i-1} + \mathbf{nd}_i \qquad (2)$$

A DOM for a specific product is now described by defining the phase model and describing for each phase the test transparency matrix **T** and the new defects vector **nd**. DOM is then used for modeling the test economics in two ways:

1) Calculation of yields and test qualities

The costs of test and repair depend on the input yield and the test quality. Input yield and test quality of a test affects the number of parts going to repair and therefore the repair cost in this phase. But test quality also affects directly test engineering cost (in terms of more or less complex test programs) and test application cost (in terms of different test equipments and test times). If the defect behaviour meets the conditions of the poisson distribution, the input yield Y_i can be derived from DOM:

$$Y_i = e^{-|\mathbf{d}_i|} \qquad (3)$$

where Y_i is the output yield and $\mathbf{n}_i$ is the vector of the number of possible defects.

Equation (3) requires independency and a uniform distribution of the defects. The lines | | define the sum over all defect types and not the vector length.

2) Calculation of input and output numbers for each phase

The costs - and therefore the test economics - for each phase depend on:

- the number incoming parts; this number impacts mainly test and repair costs.
- the required number of outgoing parts; this number impacts mainly manufacturing costs.

See figure 2 for an example. The number of input parts of a phase depends on the number of output parts of the previous phase. The output number of a phase depends on:

- for a testing phase (TA_{i+1}):
  the input number (N_{INi+1}), the relation of good and bad incoming parts (input yield, Y_i) and the test quality (Q_{Ti+1}).

$$N_{PASSi+1} = N_{INi+1} \bullet \frac{Y_i \bullet Y_{i+1}}{1 - Y_{i+1}} \tag{4}$$

$$N_{FAILi+1} = N_{INi+1} - N_{PASSi+1} \tag{5}$$

- for a manufacture phase (M_i):
  the required output volume (N_{REQ}) and the manufacture yield (Y_i).

$$N_{OUTi} = N_{REQ} \bullet \frac{1}{Y_i} \tag{6}$$

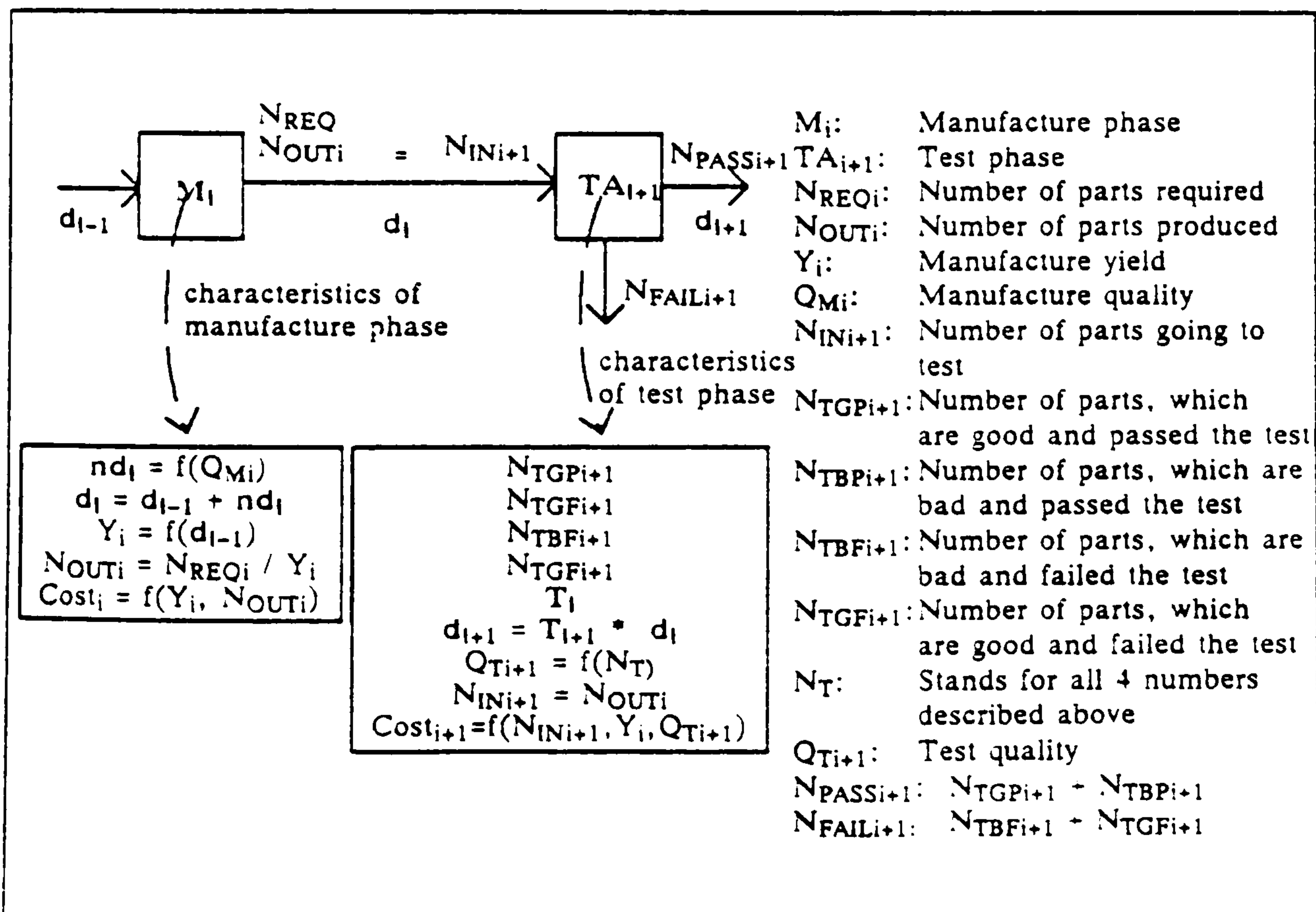


Figure 2: Defect occurencies and number of parts for a manufacture and test phase

2.3.5 Conclusions

DOM is a very effective model to describe yields, volumes and the influence of life cycle phases on these parameters. Based on these parameters, the test related costs can be modeled in a very compact way. The main advantages of DOM are:

- Several test strategies with different quality levels during the manufacture process can be compared and economically evaluated.
- The model enables to cover all failure types in one model.

2.3.6 References

[Dav82] B. Davis: The Economics of Automatic Testing McGraw-Hill Book Company, 1982

[Dis91] C. Dislis, J. Dick, A.P. Ambler: An Economics Based Test Strategy Planner for VLSI Design Proc. Europ. Test Conference, 1991

[McC88] E.J. McCluskey, F. Buelow: IC Quality and Test Transparency Proc. Int. Test Conference, 1988

5.1

Test cost modelling techniques and costs

C. Dislis, A.P. Ambler and J. Dick† – Brunel University, Uxbridge, Middlesex. UK.
†Siemens-Nixdorf, Munich, Germany.

5.1.1 THE COST OF TESTING AND THE COST OF NON-TESTING

This chapter will discuss the various ways cost modelling can be used to help make the best use of resources, increase product quality, create strategies for test, or even identify where problems are likely to occur. It will attempt to address some of the different ways cost modelling has been used in the past, describe current work by the authors, and suggest some ideas for the future.

Although ICs and IC systems have become increasingly complex and their cost has effectively dropped, the cost of test has in fact increased as a result of the increase in complexity and can account for as much as 55% of the total product cost. The cost of test in reality increases faster than linearly for a linear increase in circuit complexity [Turino90]. This increase can in fact be in the range of 35% to 55%, due to the increased complexity and the use of new packaging technologies such as surface mounting, which restrict accessibility.

The author(s) wish to acknowledge the support of ACM/SIGDA sponsorship in the organisation of the First International Workshop on the Economics of Design and Test

The accessibility problem is even more evident in ICs where the complexity and gate count have risen much faster than the number of pins on the package. As a result, testing to a high fault cover is becoming increasingly more expensive, unless special design for test provision is made. It is also becoming increasingly difficult to test to a satisfactory level, simply because of the time constraints in testing large circuits. For example Daniels and Bruce [Daniels85] when describing self test methods for Motorola microprocessors, mention that a complete test of all instructions under all conditions of the MC6800 would take 2 million years to execute. The MC6800 consisted of approximately 4000 transistors, compared to the 68,000 transistors of the MC68000.

Although design for testability helps to tackle the testing problem, there still persist many misconceptions surrounding the cost of test and as a result, many devices and systems are inadequately tested. One such practice is the use of verification patterns for testing, thus bypassing the often time consuming test pattern generation step. However, design verification vectors typically provide a fault cover of between 40% and 70%, while it is known that fault coverage greater than 98% is needed to ensure adequate quality [McCluskey88]. High test cover at component test is particularly important for low process yields. For example, at 90% fault cover, 6.7% of bad ICs will be passed as good for a yield of 50%, while 14.9% of bad ICs will be passed as good if the process yield drops to 20%. These figures drop to 0.7% and 1.6% respectively, if the fault cover is increased to 99% [Racal-Redac89]. Test decisions at the component stage influence subsequent stages and therefore quality considerations need to be incorporated in any evaluation of the true cost of test, and are obviously important in determining the cost effectiveness or otherwise of DFT methods.

Area overhead as a result of design for test is often thought to be a major factor in increasing costs. However, this is not always the case, as it is possible that the test savings can outweigh area overhead related costs. The problem then is to find the optimal level of design for testability that will result in overall savings. Figure 1 shows an example of costs vs test related area overhead, which clearly shows that there is an optimum area overhead value (greater than 0), for which overall costs are minimised.

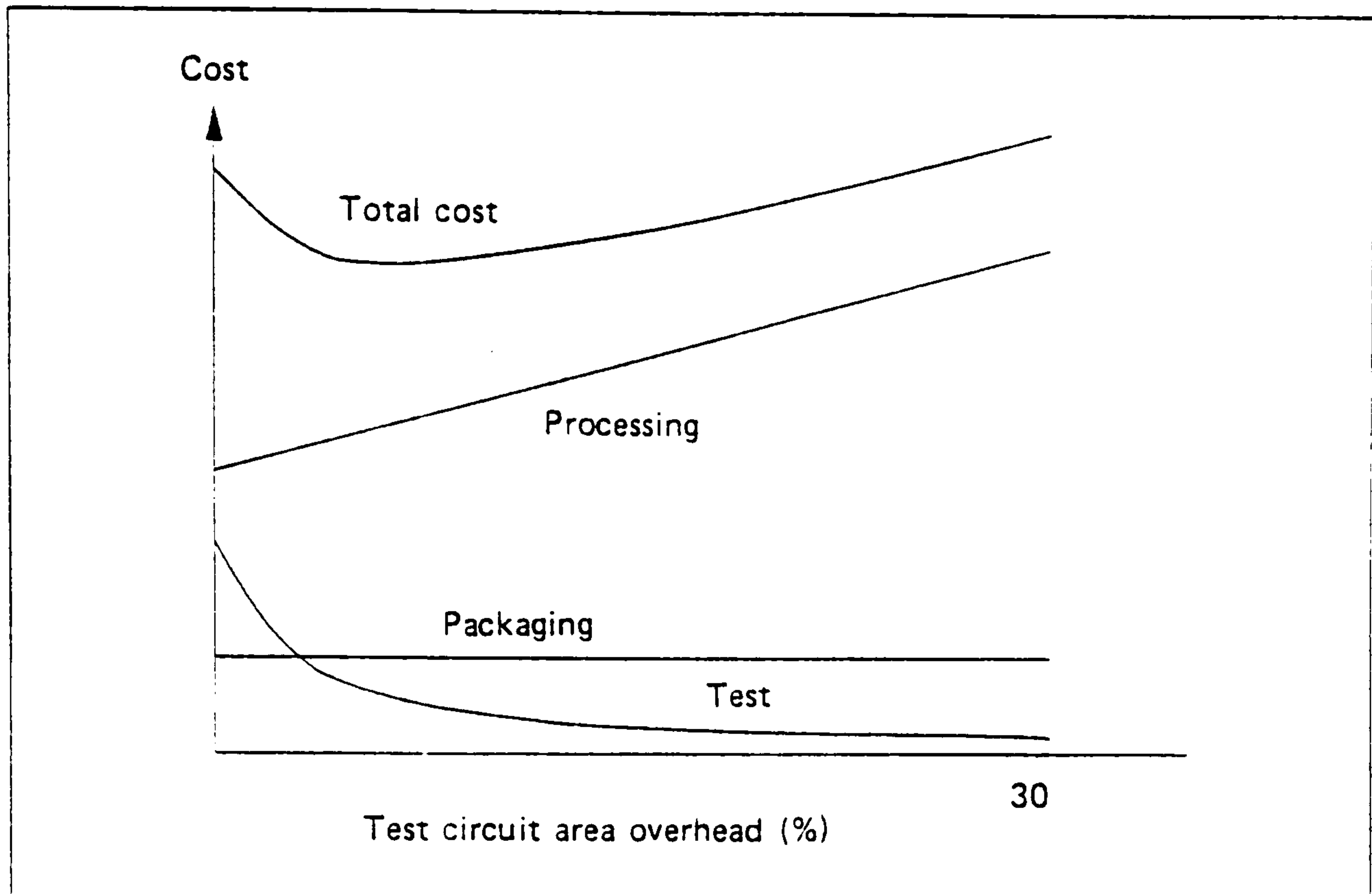


Figure 1. Costs associated with test area overhead

5.1.1.1 Product quality

However, the question of testing is more wide ranging. On leaving the supplier, devices and boards become part of possibly large and expensive systems, whose reliability depends to a large extend on the quality of test used for the ICs they contain. Faults that escape to subsequent stages, for example from chip level to board level, become increasingly difficult to discover and fix, due to the increased complexity of the system. Also money has been spent in terms of packaging, materials, effort and time, on a product that is defective. The later this product fails, the more expensive a failure becomes. If a chip fails at component test, it is simply discarded. If the same chip fails at board test, diagnosis and repair have to be performed. It is the same at system test, only this time the process is likely to be more time consuming and therefore more expensive. However, if the system fails in the field, down time costs can be extremely large, and diagnosis will often involve an engineer being called on site. Additionally, if the board and system tests are not geared towards

testing the chips in the system, but consider mainly assembly and interconnect faults, then very few of the faults that escaped component test are likely to be detected. In the case where a chip test is used at board test, it should be a different test to the one used at component level. If not, then most of the faults that escaped previously will again not be detected, and will possibly result in failures in the field. In the cases of highly critical applications such as satellite and military systems, failure can not only be expensive, but also catastrophic.

A much quoted rule of thumb states that costs for the detection and repair of a fault increase tenfold at each stage of its lifecycle. Therefore, a faulty IC might cost 1$ to discover at IC test, 10$ at board test, 100$ at system test and 1000$ in the field. This is definitely a generalisation, but provides a useful pointer to the importance of ensuring a high fault cover as early as possible. This discussion on the cost of non testing does not even begin to address the effect on a company's reputation of shipping unreliable products.

In fact, complex legal aspects of product liability also need to be taken into account. The legal liability of the manufacturer and the engineer cannot be overlooked. Product liability claims (largely for loss of property or personal injury) in the United States have tripled in the large decade, and the percentage of cases where the jury has ruled in favour of plaintiffs has increased substantially, as has the amount of the average award [Brown91]. Although these statistics encompass a variety of technology products, it is clear that it is in the interests of manufacturers to improve product quality and minimise litigation.

5.1.1.2 Cost savings due to DFT

Design for testability can no longer be thought of-as an optional (and expensive) extra. It can not only improve the quality of a product, but can also offer real cost benefits. For example, the use of DFT can reduce the need for expensive ATE, and eliminate test bottlenecks where several projects have to use the same test equipment. The use of DFT for a particular product can free a heavily used ATE for another project, thus cutting down on time to market for both cases. Design verification time

as well as test pattern generation time can also be reduced as a result of DFT.

The savings that can be made depend on the type of product, the volume of production, and on how long a particular device or system will be in production for. In a low volume, wide variety situation, the majority of savings will be made on non-recurring engineering costs such as test equipment, test fixtures and test generation and programming. In a high volume, low variety case, the major savings will be in recurring areas, such as on going test and troubleshooting costs [Turino90]. This is to be expected, as non-recurring costs are amortised over the production volume and are therefore less significant in the high volume case.

5.1.1.2.1 An example - scan path testing

Scan path testing is a well known DFT method, although not as widely used as it might be, due to designer's worries that it is expensive. Scan path increases controllability and observability, and, as it reduces the sequential test problem to a combinatorial one, it allows the use of automatic test pattern generators, making it possible to achieve near 100% fault cover in a shorter time and at significantly reduced cost. The use of scan path not only makes IC testing easier, but it can be also be used for design debugging, system test and field service, and allows a better quality product to be shipped to the customer.

The use of scan path however, is not without penalties, and the trade-offs require careful analysis in order to avoid costly mistakes. The following are some of the reasons that scan path is considered expensive [Dear91]:

Scan path can incur some additional design effort, the extend of which depends largely on the designer's experience and the availability of suitable CAD tools. There is also some area overhead associated with scan, which ranges from around 4% to 20%, and can lead to yield and reliability degradation. Additional device pins can sometimes create a problem, especially in cases where use of the next package size is required resulting in a penalty in terms of space and cost. Test application time can also increase with scan, although this can be eased by the use of a scan tester.

The cost advantages and cost penalties of scan can be compared on economic grounds to produce a quantitative evaluation. The graph in figure 2 shows an example of one such evaluation. In this case, the cost of test always favours the use of scan. This evaluation did not include the benefits of the scan design philosophy to the board, system, and field test stages.

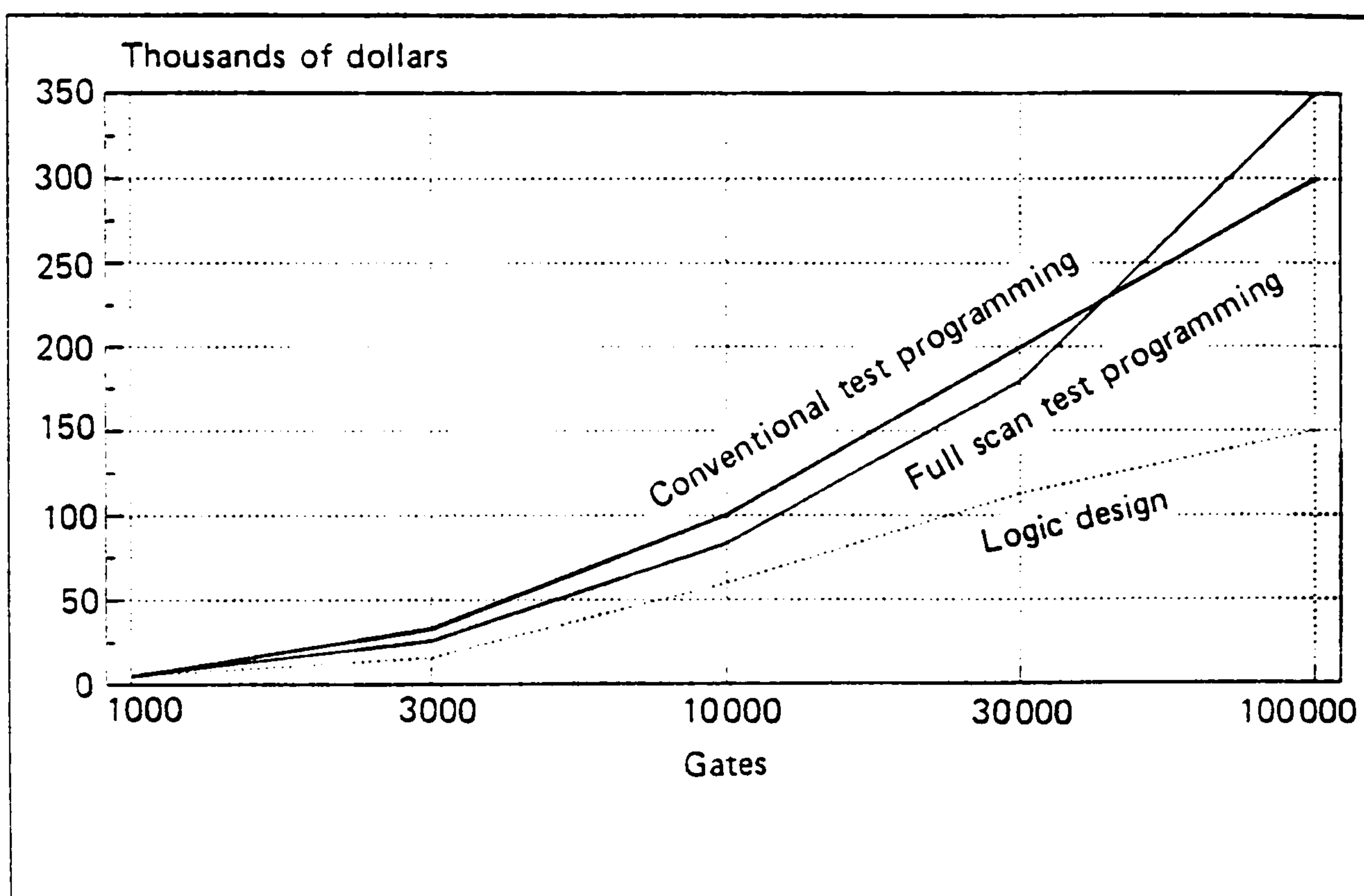


Figure 2. Typical ASIC engineering costs

5.1.1.3 Time to market

A considerable saving that can result from the use of DFT techniques is in terms of time. Adopting design for testability methods can considerably reduce development times and therefore cut the time-to-market. Design verification and test program generation can be cut by, typically, between 5% and 15%.

Fast development time has real financial benefits for the manufacturer, although the exact economic advantage depends on the state of the market

and the current competition [Smith91]. In general, early introduction of a product means that the product sales window is increased. If a product is introduced early, it seldom becomes obsolete any sooner, and the early start ahead of the competition also provides the manufacturer with more pricing freedom. The early introduction can result in a larger customer base and an increased market share, particularly in the case where switching to another manufacturer's product can be an expensive exercise. Figure 3 shows the financial benefits of operating in a longer sales window [Smith91].

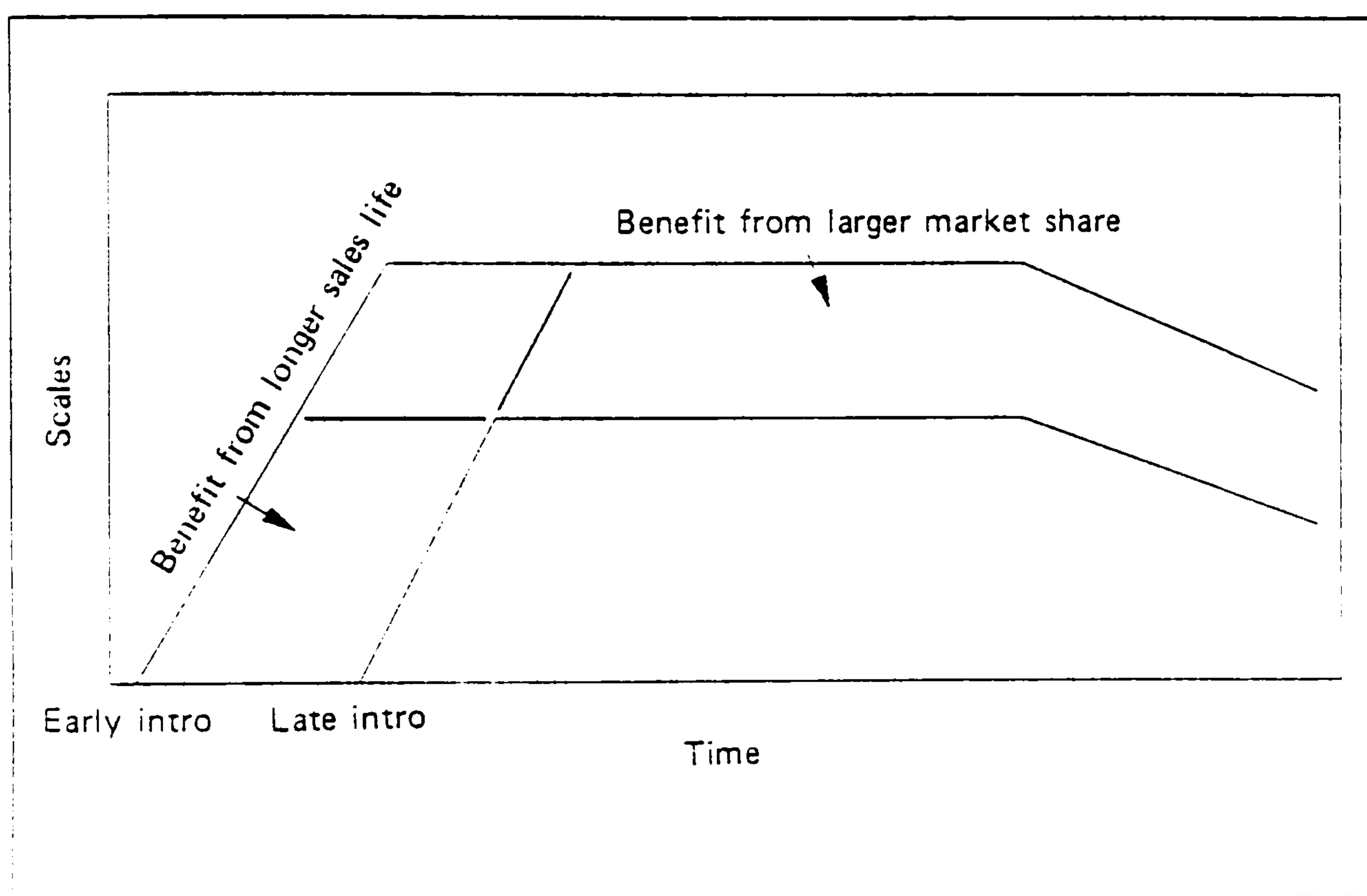


Figure 3. The benefits of early product introduction

5.1.2 MODELS TO PREDICT THE COST OF TEST

The prediction of the costs of test is an important consideration for companies, as they are likely to influence the cost of the final product (be it at chip, board or system level), testing strategies and company policies. There exists however, a great variety of parameters which will influence these costs, and there are also differences in test strategy between

companies. These costs can be modelled, and a number of papers have been published describing the use of economics models for a variety of applications. However, both the scope of the model and the application of it need to be carefully defined.

By far the most common use of test cost modelling is in examining the costs of using automatic test equipment and determining the optimal tester use. Existing models use techniques of various levels of sophistication in order to compare test strategies, evaluate the cost of test application and determine product quality [Taschioglou81]. In attempting to satisfy these objectives there are several important considerations, such as examining the optimum level of test required by costing the effects of escaped faults at each stage, as well as taking the complexity of the product under test into consideration [Myers83].

Economic models can also be used to examine not only test strategies but also the performance of ATE systems. Most economic analyses performed in order to determine a test strategy in terms of the mix of ATE equipment used, analyze the cost effectiveness of different testers. Another way to look at the problem is to analyze the cost effectiveness of enhancements (in terms of CPU, mass storage, and tester throughput) to a specific tester [Huston83]. Other aspects of test can also be examined by the use of cost modelling techniques. The prediction of test pattern generation costs is one particularly difficult problem, and empirical models can be used to approach it, ultimately linking test pattern generation costs to circuit complexity and required fault cover [Goel80].

Another area of interest where the use of economics models can prove is in the evaluation of design for test aspects and costs. This is often done by considering the increased die area and associated production costs, which can be offset against reduced tester use or the use of a cheaper, less powerful system [Pitman84]. There are models which can be used to evaluate the area overhead resulting from design for test methods ([Ohletz87], [Miles87])

What these techniques have in common is that problems are viewed in an isolated manner. For example, ATE use may be addressed in isolation from test pattern generation and from the design for test method used. An

alternative method is to attempt to unify the test cost modelling by considering the whole design, production and test cycle for a system, and the costs associated with improving the testability of the design [Varma84], [Dear88], [Dislis89].

5.1.3 USING ECONOMICS MODELLING FOR TEST STRATEGY PLANNING

Economics modelling can be automated for embedding into systems for making decisions on various test aspects. One such system has been developed jointly by SIEMENS and Brunel University and is currently under industrial evaluation [Dislis91]. It uses global cost modelling methods based on industrial data to aid designers in choosing the optimal mix of structured DFT methods for a particular application at IC level. A knowledge base of DFT methods specific to functional blocks holds economic and design information. Test decisions can be guided by the user, or be taken by a selection of automatic test strategy planning algorithms, all based on cost evaluation of alternative strategies. The technique can be extended to examine many other test related aspects. This work is now been extended to VLSI based systems, in order to create a complete testability advisor. The system was developed with the needs of industry in mind, and there is built in flexibility in the implementation of the cost model used, as well as the selection of test methods in the knowledge base. Speed of calculation was also an issue, so backtracking was avoided as much as possible. As the system was designed to operate in the early stages of the design, it is up to the designer to implement the suggested measures, although help is provided in the implementation.

5.1.3.1 The Structure of the Test Strategy Planning System.

Figure 4 shows the outline of the test strategy planning system. The design description is acquired either directly from the user or from an existing netlist, and is built in a hierarchical fashion in order to allow test strategy decisions to be made at several stages of the design process. A variety of essential economic data is also acquired at the same time.

The data is used to automatically update an economics model prior to the test strategy selection process. The test strategy planner uses stored knowledge on test methods as well as the design data and the existing cost model to evaluate a variety of test strategies. The test strategy control can be left entirely to the user, or alternatively, a degree of automatic planning may be employed to accelerate the selection.

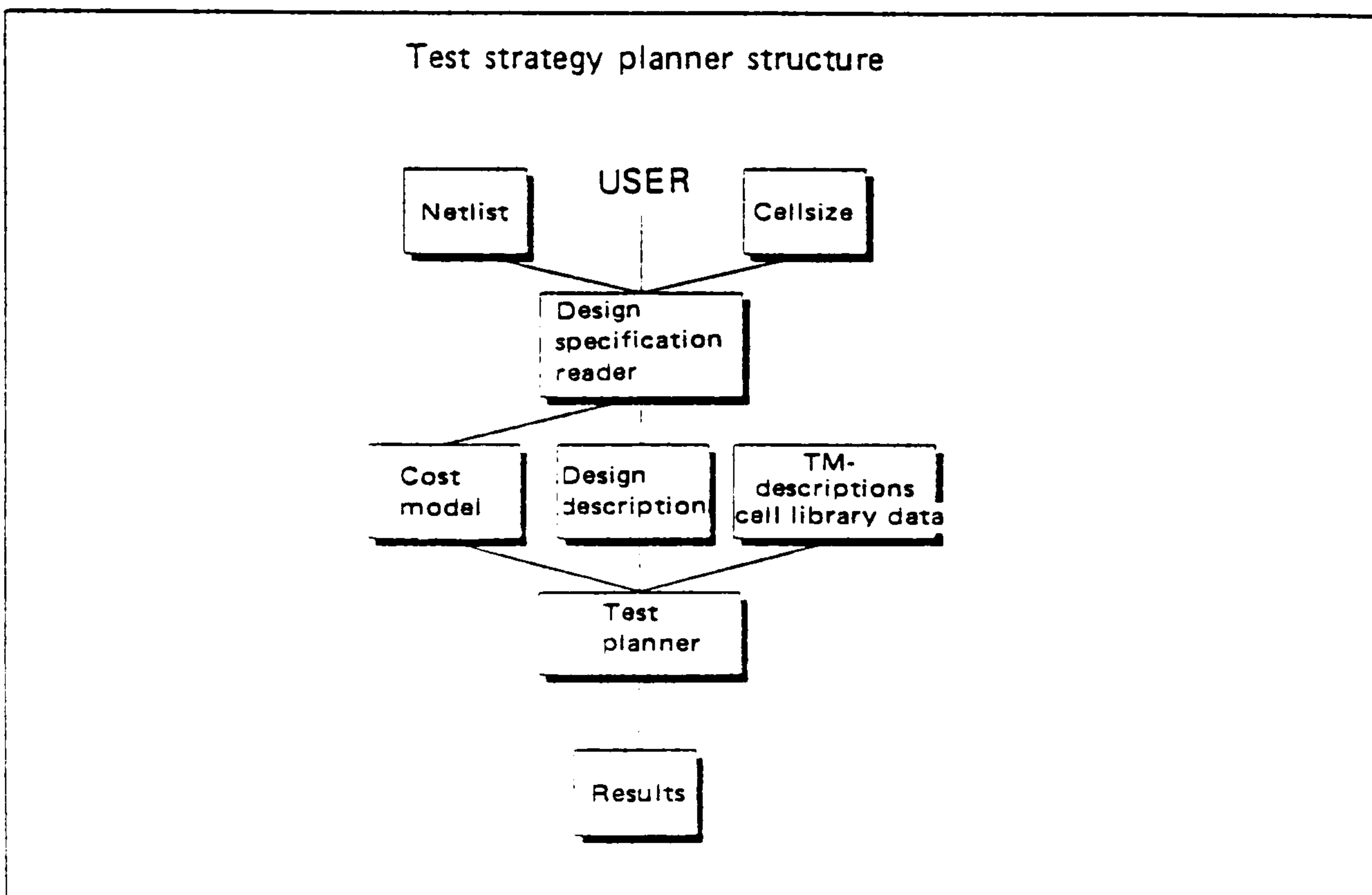


Figure 4: System architecture

5.1.3.2 The cost model

The cost model categorises primary parameters supplied by the user into design independent (mostly global company costings), design dependent (which will change with the design), test method dependent, and constant factors. A set of equations is then calculated using the supplied values. Design costs are modelled in terms of and equipment required, the cost of using an external design centre if this option is taken, and manpower, which is a function of the complexity of the system, and the productivity of the design environment. Productivity is modelled in terms of the

designer's experience, the performance of the CAD system, and the functionality of the cell library.

Production costs for are linked to the design complexity (measured in gate equivalents, grids or mm²), mostly with a linear relationship within certain complexity ranges. In this case, gate equivalent count was used as a complexity measure. A test strategy may influence the gate count and therefore the production cost per unit. In addition, non recurrent engineering (NRE) charges must be added to the production cost. Unlike previous models [Varma84, Dislis89], yield effects are not modelled separately, as they are included in the pricing policy of the vendor.

Test costs are separated into test pattern generation and test application costs. Test application costs are linked to the number of test patterns in linear or stepwise ranges, similar to the way gate count is linked to production cost. This pricing is normally provided by the vendor, and removes the need for the separate modelling of ATE costs present in previous models. Test pattern generation costs are estimated as follows: the cost of automatic test pattern generation is estimated, together with the maximum achievable fault cover. If the achievable fault cover is less than the required value (a fundamental requirement) then the cost of achieving it using expensive manual tpg is estimated. Using structured DFT methods often means that ATPG techniques alone are effective for producing the required fault cover.

5.1.3.4 The test strategy planning process

The user has the option to apply and evaluate test methods for specific blocks manually, which allows a lot of flexibility. However, it was necessary to also automate this process, and a number of algorithms are supplied with the system. These work on the following principles: The first requirement is to make every block testable. This means that the block would be easily testable if it were tested in isolation, with perfect access to all its i/o lines. On this basis, a cost optimal testability method is chosen for every block in the circuit, by evaluating all suitable methods for each. At the end of this process, every block should be testable, but the accessibility of some blocks might also have increased.

The next requirement is one of accessibility to all i/o lines of every block, in order to be able to propagate the test patterns required. In order to determine where the accessibility problems are situated, the accessibility of i/o line groups is established. This information is used to determine whether any accessibility test methods need to be applied. This may or may not be the case, as many testability enhancing methods have the side effect of improving accessibility as well. There may be several candidates for the application of accessibility methods, and the aim of the algorithm is to find the block which has the maximum impact on the accessibility of the circuit, taking into account existing (already applied) test methods and the transparencies of paths through functional blocks.

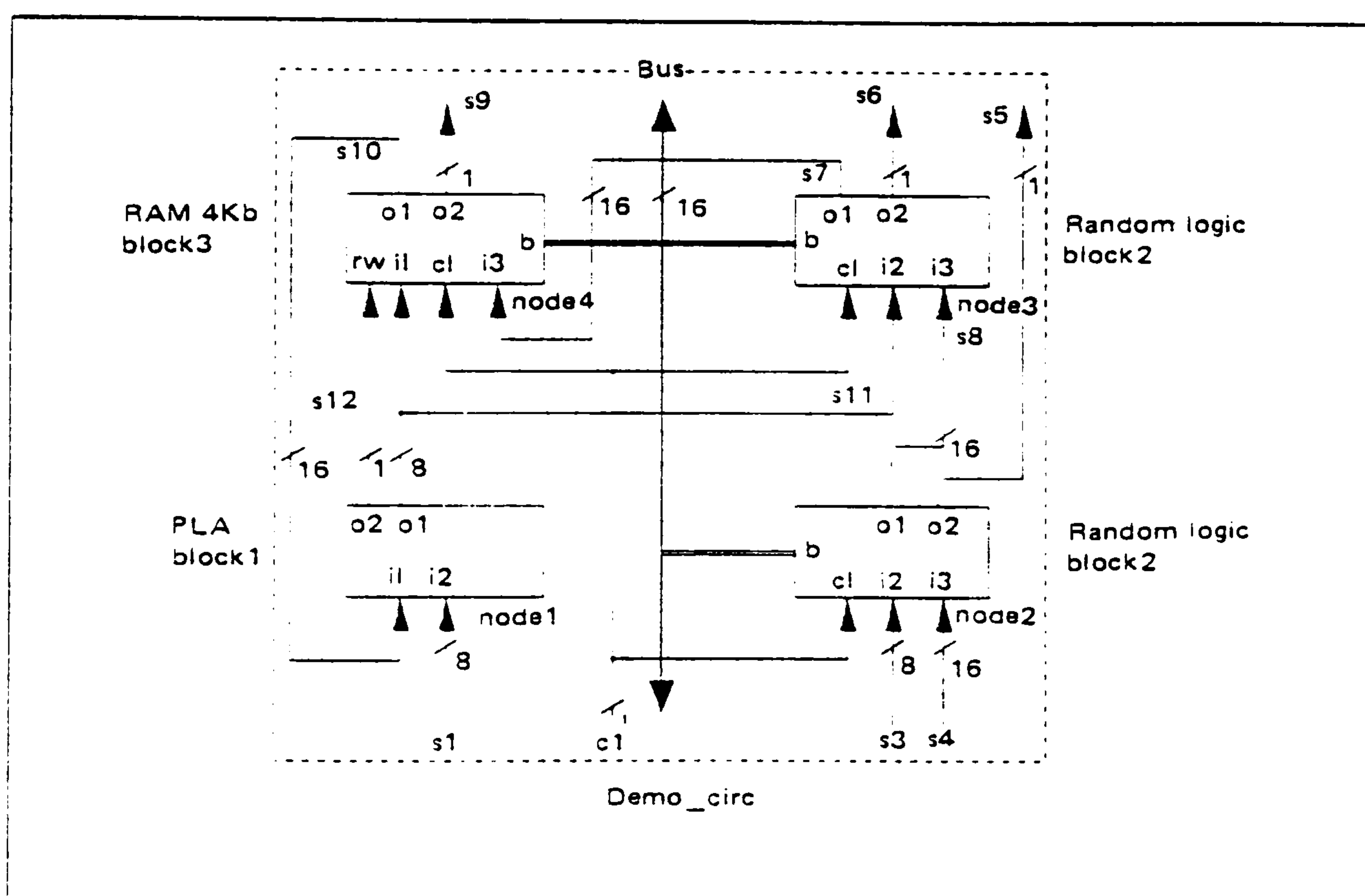


Figure 5. An example circuit

Figure 5 shows an example circuit, and figure 6 shows example costs of the automatic test strategy planning process as each block is tackled. It should be noted that the production volume is a sensitive parameter, altering the relative strategy costs. The test strategy results are obviously valid only for the specific circumstances, and the test methods

([Treuer85], [Illman86],Zhu88]) are block specific and their descriptions are part of the system [Dislis91].

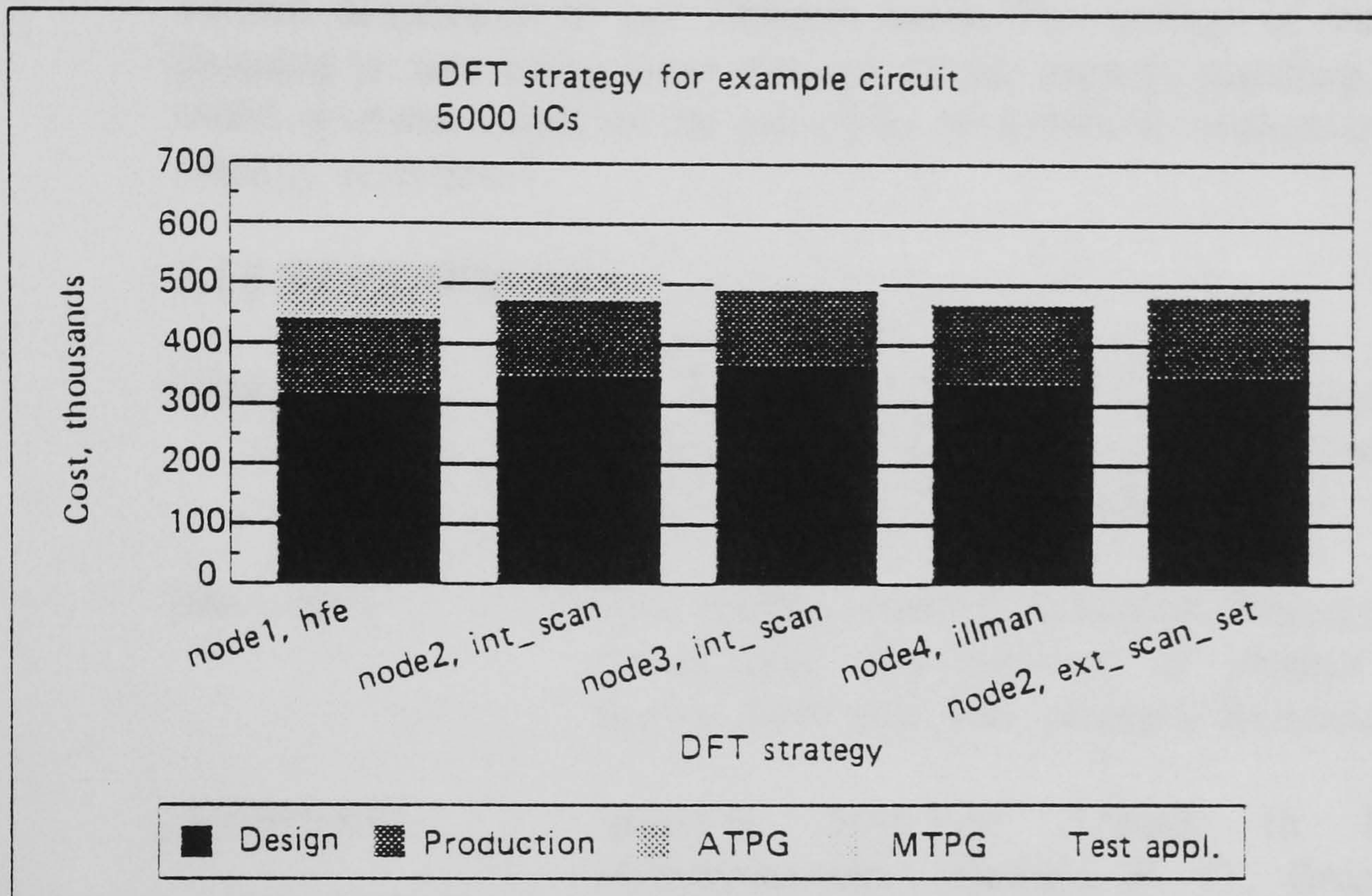


Figure 6. DFT strategy for 5000 ICs

5.1.4 CONCLUSIONS

This chapter introduced some of the factors that affect test costs and test decisions, and discussed the way DFT can affect them. Some of the difficulties of test were highlighted, and the need for a high fault cover was illustrated. The prediction of test related costs is an important consideration, and a variety of models which attempt to analyze test costs from different viewpoints were reviewed; one of these specifically for the evaluation of alternative (broad) DFT methods. Finally, the possibility of using economic modelling techniques for detailed decision making on the choice of the optimal combination of test methods for a given circuit was discussed. A system for test strategy planning using economic factors was then described, in order to illustrate the industrial validity of the technique.

The system described here is still under evaluation, but we believe it is a useful tool for aiding designers in enhancing the testability of their designs. It is easily reconfigurable in terms of the cost modelling and test method description to suit different users. The concept of test strategy planning is now being taken through to test strategy planning for VLSI based systems, based on the principles of economic evaluation and test strategy selection.

5.1.5 REFERENCES

[Abadir89] 'TIGER: Testability Insertion Guidance Expert System', M. Abadir, Proc. IEEE International Conference on Computer Aided Design, 1989.

[Brown91] 'The product liability handbook. Prevention, risk, consequence and forensics of product failure', Brown, Sam (ed), Van Nostrand Reinhold, 1991

[Daniels85] 'Built-In Self-Test Trends in Motorola Microprocessors', Daniels, R. G., Bruce, W.C., IEEE Design & Test of Computers, Vol. 2, No 2, April 1985, pp 64-71

[Dear88] 'Hierarchical Testability Measurement and Design for Test Selection by Cost Prediction', I.D. Dear, C. Dislis, S.C. Lau, J.R. Miles, A.P. Ambler, Proceedings of the European Test Conference, 1988.

[Dear88] 'Hierarchical Testability Measurement and Design for Test Selection by Cost Prediction', I.D. Dear, C. Dislis, S.C. Lau, J.R. Miles, A.P. Ambler, Proc. European Test Conference 1988.

[Dear91] 'Economic Effects in Design and Test', I.D. Dear, C. Dislis, A.P. Ambler, J. Dick, IEEE Design and Test of Computers, vol. 8, no. 4, December 1991.

[Dislis89] 'Cost Analysis of Test Method Environments', C. Dislis, I.D. Dear, J.R. Miles, S.C. Lau, A.P. Ambler, Proc. International Test Conference, 1989.

[Dislis91] 'An Economics Based Test Strategy Planner for VLSI Design', C. Dislis, J. Dick, A.P. Ambler, Proceedings of the European Test Conference, 1991.

[Goel80] Goel,P. 'Test Generation Costs Analysis and projections' Proceedings of the IEEE 17th Design Automation Conference, 1980, pp 77-84.

[Huston83] Huston, R.E. 'An Analysis of ATE Testing Costs' Proceedings of the IEEE International Test Conference, 1983, pp 396-405

[Illman86] 'Design of a Self Testing RAM', R.J. Illman, Proc. Silicon Design Conference 1986.

[McCluskey88] 'IC Quality and Test Transparency', McCluskey, M.J., Proc. IEEE International Test Conference, 1988, pp 295-301

[Miles88] J. R. Miles, PhD Thesis, Brunel University, UK, 1988.

[Myers83] Myers, M.A. 'An Analysis of the Cost and Quality Impact of LSI/VLSI Technology on PCB Test Strategies' Proceedings of the IEEE International Test Conference, 1983, pp 382-395

[Ohletz87] Ohletz, M. J., Williams, T. W., Mucha, J. P. 'Overhead in scan and Self-Testing Designs' Proceedings of the IEEE International Test Conference, 1987, pp 460-470

[Pittman84] Pittman, J. S., Bruce, W. C. 'Test Logic Economic Considerations in a Commercial VLSI Chip Environment' Proceedings of the IEEE International Test Conference, 1984, pp 31-39

[Racal-Redac89] Private communication, Racal-Redac, 1989

[Smith91] 'Developing products in half the time', Smith, P.G., Reinertsen, D.G., Van Nostrand Reinhold, 1991

[Taschioglou81] Taschioglou, K. P. 'Applying Quality Curves for Economic Comparisons of Alternative Test Strategies' Proceedings of the IEEE International Test Conference, 1981, pp 331-339

[Treuer85] 'Implementing a Built-In-Self-Test PLA Design', R. Treuer, H. Fujiwara, V.K. Agrawal, IEEE Design and Test of Computers, Vol. 2, April 1985.

[Turino90] 'Design to Test - A definitive guide for electronic design, manufacture, and service', Turino, Jon, Van Nostrand Reinhold, New York, 1990

[Varma84] 'An Analysis of the Economics of Self Test', P. Varma, A. P. Ambler, Proceedings of the IEEE International Test Conference, 1984.

[Varma84] 'An Analysis of the Economics of Self Test', P. Varma, A. P. Ambler, Proc. IEEE International Test Conference, 1984.

[Zhu88] 'Analysis of Testable PLA Designs', X. Zhu, M. A. Breuer, IEEE Design and Test, August 1988.

Economic Effects in Design and Test

I.D. DEAR
C. DISLIS
A.P. AMBLER
Brunei University

J. DICK
Siemens-Nixdorf

Because of misconceptions and myths about the cost of test, many devices and systems are inadequately tested. Focusing on application-specific ICs, the authors discuss the economics of test and show how economic analysis leads to test that pays back. They also present the EVEREST test strategy planner, a design tool that aids in the selection of DFT structures during ASIC design, using cost as a primary selection paramerer.

It can be argued that not enough information is available about the economics of electronic circuit design, either about specifically design-related issues or about test issues. However, the information that is available can be illuminating and interesting. Davis has written a book about test costs.[1] Also, Fey and Paraskevopoulos have written about the design costs of integrated circuits.[2,3]

Area overhead is often perceived to be a major factor in test cost increase. Figure 1 provides an example of costs as a function of test-related area overhead. However, certain questions arise from this relation, such as, what level of test area overhead leads to minimum costs?

The available information about the economics of circuit design must be studied carefully because it cannot relate, other than in a very broad sense, to any one set of circumstances. The wide range of factors that can affect test costs can vary widely from one installation to another and from one design to another. However, an economic analysis of individual test costs often points the way to more efficient use of test tools and DFT techniques and eventually relates to improved product quality when it is realized that good test can pay back.

In this article, we discuss various test cost issues. We also present a test cost model that has been used as a pure spreadsheet model and is now being used as part of a test-strategy planning tool.

Test economics

As might be expected, the cost of test, if nothing is done to ease the problem, increases faster than linearly for a linear increase in circuit complexity.[4] While total product cost has decreased over the past few years, test cost has risen as a percentage of this total product cost to more than 55% in some cases. The extent of the increase depends on the product size, technology, and complexity.

Unfortunately, many myths surround test costs. Test pattern generation has not been generally accepted as a possible answer when observed from the perspectives of fault coverage and CPU time. Significantly large CPU times for fault simulation are often cited as obstacles to effective test. Though there are a variety of solutions to such difficulties, the solutions have not been widely accepted. Among the many reasons given for not using such DFT methods as scan and built-in self-test are the familiar problems of performance penalty, area overhead, and pin-count overhead.

Because of these misconceptions, many devices and systems are inadequately tested. Silicon vendors say that up to 50% of "working" application-specific ICs do not work in the target system: The designs have not been specified or verified correctly because of lack of simulation,[5] and they have not been tested adequately. Design verification tests for devices typically provide 40-70% fault coverage, despite warnings that greater than

0740-7475/91/1200-0064$01.00 © 1991 IEEE

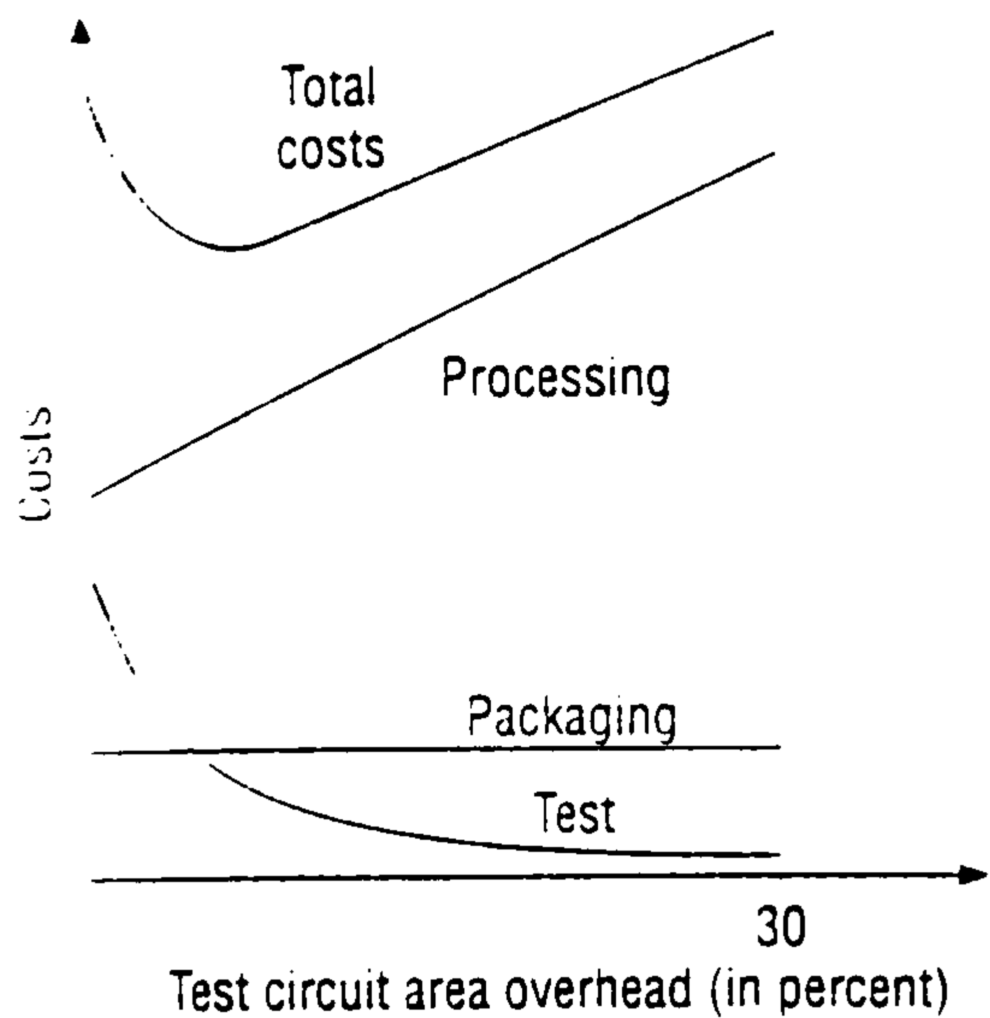


Figure 1. Plot showing costs and test-related area overhead.

98% fault coverage is necessary to ensure adequate quality.[6]

To the uninitiated, publicity showing that the cost of test is rising dramatically only reinforces the opinion that test is not cost effective. Concern about the real cost of test leads to shortcuts and to a naive reliance on the results of perhaps questionable test-generation and fault-simulation methods.

Because it is a negative quantity, the cost of test is easier to focus on than the benefits, especially when an "over-the-wall" mentality prevails, as it often does when the design team is asked to use its resources and budget to improve the lot of the test department. To make realistic choices, we must understand the problem of testability in more detail, and in terms related to the whole company, not just to individual departments. Let's consider scan path use as an example in the next section.

Scan path testing

Scan path test methodology is well understood,[7] and we will not expand on it here. Scan path use increases observability and controllability. It reduces the test problem from sequential circuit test to test of a combinational circuit plus a shift register. Automatic test pattern generation capability is available commercially for combinational logic, giving the possibility of near-100% stuck-at fault coverage and thus a higher quality device under test. Scan can be used at all stages in a system design, from initial design debugging and device production test to system test and field service.

However, often-cited penalties of using scan include

- additional design effort
- additional circuitry (4-20% overhead)
- additional device pins, sometimes requiring use of the next package size, which takes up more space and costs more
- possible increase in test-application time
- degradation in reliability and yield

An industry survey[3] shows that very few ASIC designs incorporate scan. Often-cited reasons are area overhead; degradation in performance, reliability, and yield; and the necessity for design constraints.[3]

Some of the negatives above relate to design philosophy and performance issues that we do not discuss here. The wide range of software tools and support available for scan design includes scan design rule checkers, automatic scan-conversion tools, and automatic test pattern generation. The other more directly cost-related issues require some detailed analysis for a true comparison. The factors involved include the following:

- The balance between increased gate count and the impact the increased count might have in increased design time, greater processing costs, and decreased yield (fewer devices per wafer) and reliability
- The balance between easier test pattern generation and higher fault coverage on the one hand and potential increase or decrease in test time (serial application of tests, which might be aggravated or eased by the absence or presence of a scan tester) on the other

Such factors can be compared on straight economic grounds for a quantitative evaluation of scan as a test method. The qualitative issues are for individual design teams to evaluate.

Nevertheless, the cost of test, with or without scan, very much favors the use of scan,[3] as Figure 2 shows. The graph in Figure 2 does not take into account the

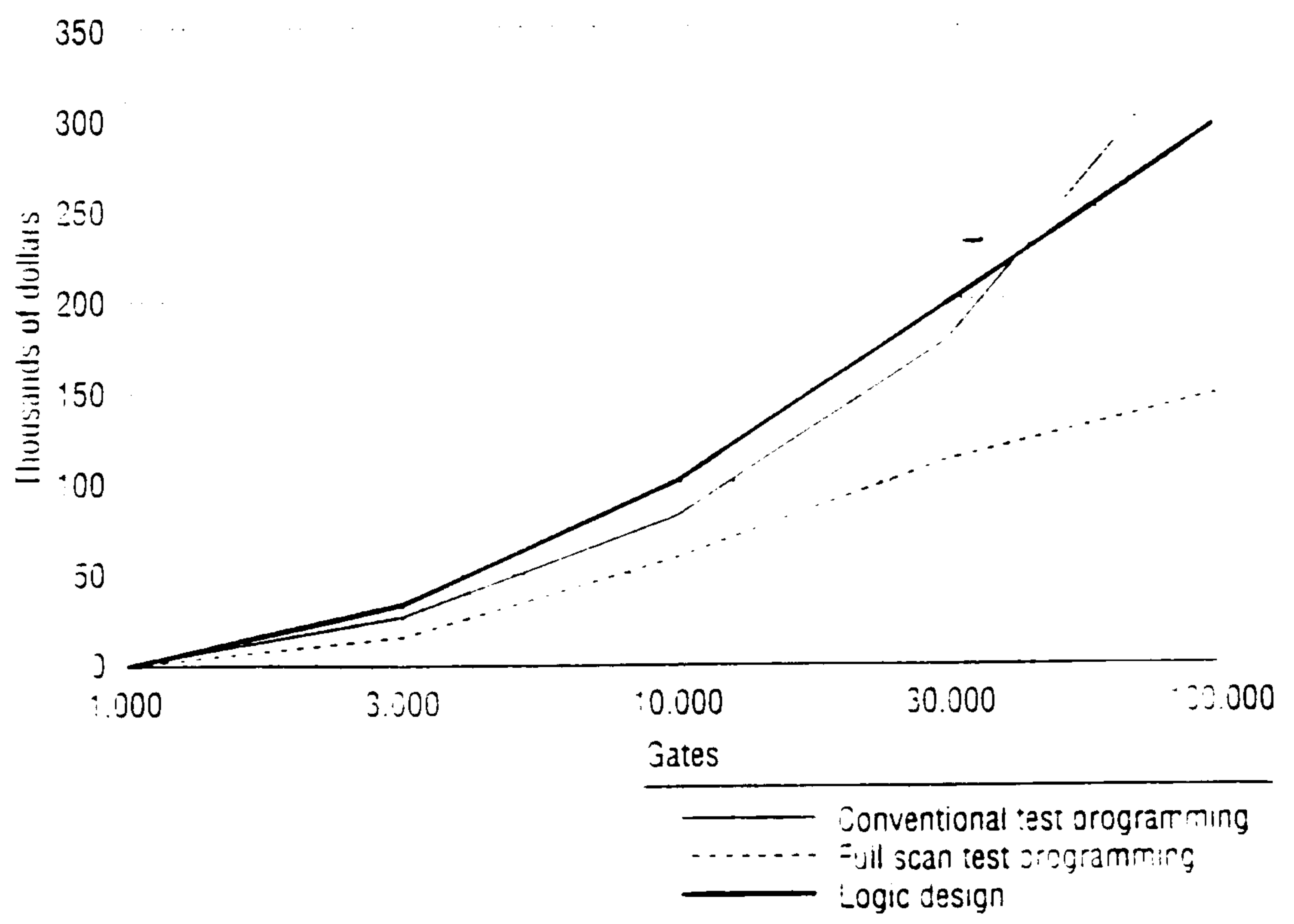


Figure 2. Typical ASIC engineering costs.

benefits to the board, system, and field test stages when the entire design includes the scan design philosophy. Dervisoglu[10] demonstrated the benefits of systemwide scan test.

The effectiveness of improved testability—and hence scan—from the device level to subsequent test stages is simply put by the "rule of tens."[1] This rule holds that leaving test to the later stages of system integration and poor-quality test at the device level lead to very large cost increases in field failures. According to the rule, if detecting a fault at the device level costs, say, $1, then detecting the same fault at the board level costs $10; at the system level, $100; and in the field, $1,000. Analyses of the real costs based on actual examples show this rule of thumb to be reasonable. (The figures for testing a design flaw could follow a similar pattern and be a lot higher.)

Need for economic analysis

In addition to the need for a straightforward economic analysis of the costs of test, consider the dilemma of designers who need to choose a DFT strategy for their ASIC. Even for designers completely familiar with the available test techniques, the choice can be difficult. For example, Zhu and Breuer[11] discuss 20 BIST methods for programmable logic arrays. Given that designs can include PLAs and other structures as well, the list can become daunting.

Factors that must be considered when deciding on the test method for any design include, but are not be limited to, area overhead, pin count overhead, test length, fault coverage, and automatic test equipment requirements. They also include such design-dependent factors as design time, circuit size, performance impact, and test pattern generation costs.

The models we consider here extend to over 100 parameters, and we cannot discuss them in detail. When constructing a global economics model, we found it necessary to consider all these parameters rather than a subset—perhaps including only area overhead, test time, and a few other parameters—to put any true savings into perspective. There are cases in which a simple subset can be usefully applied, but we have found that the enlarged model is useful in the initial stages to prevent preconceived ideas about the relative importance of the parameters coming into play. A fuller discussion of the necessary parameters can be found in the literature.[12-14]

All factors vary with circuit style, so we need some procedure to compare and contrast the disparate values to produce a meaningful judgment. When confronted by what could be conflicting issues of longer test time versus larger area overhead versus increased pin count requirements, how can we reconcile them satisfactorily?

Of course, there are many cases in which the variables listed are not variables. For example, there may be design constraints such as package size limitations that fix area overhead or pin count (as in an aerospace design). However, how to effectively compare and contrast the effects of all the variables associated with a test strategy remains an issue.

Varma, Ambler, and Baker[12] showed the effects of analyzing many different test-related parameters using cost as the basis for comparison. Graphs of specific cases showed how the relative benefits of BIST versus scan versus no DFT varied with production quantity. In the cases studied, the changes were quite marked: BIST was the preferred method for lower production volumes, and no DFT was best for higher production volumes.

Further studies[15] brought different results: Higher production quantities *increased* the benefits of using DFT compared with not using DFT. Such work only emphasizes the requirement for economic analysis of any given design, manufacture, and test scenario.

We emphasize here, as we will again later, that because of the variety of issues considered, any figures we give in this article relate *only* to the examples from which they arise. The figures will not necessarily bear any resemblance to figures from comparable examples in other companies.

Different test-related quantities can be effectively compared using cost as the basis for comparison, allowing for true quantitative comparison. Any other form of comparison can only be subjective.

Test-related parameters

The effects of adding testability to a design are widespread. To assess the impact of a DFT strategy, we must consider both the direct impact on test costs (such as test pattern generation) and also such factors as the reduction in ATE complexity and increase in design time and cost. Let's consider the main tangible cost areas affected by test decisions: design, manufacture, test generation, and test application. Note that costs can vary greatly among products and companies. We offer a guide to the possible benefits or penalties of DFT.

Area overhead

The area overhead penalty is often cited as a principal argument against DFT. The argument that DFT increases chip area and decreases yield, hence increasing production cost, is well understood. However, no general figure can be set as the maximum allowable area overhead associated with DFT. Designers are often left to make the best assessment they can. Sometimes a limit is imposed on the design from another source. The origins of this limit are often hard to ascertain, leaving unanswered the question of whether the cost saving of the area overhead has been properly considered.

Most companies in the fabrication industry keep a close watch on the chip yield of their process lines. Depending on the type of device produced, the technology, and so on, these companies use models to calculate the yield of devices, given

a chip area. Abundant research has been devoted to this field. Even when designers have access to these models, they must calculate closely the area overhead of the whole chip before assessing the effect on the yield and the consequent effect on the production cost.

Predictive measures. The effect of DFT methods on circuit area is important, and designers should be able to predict it as early as possible. Miles[16,17] developed a method for accurately predicting the area overhead for macrocells when DFT is added.

The functional area of the macrocell is determined by parameterizing the floor plan of a given circuit structure before and after implementing a specific test strategy. Thus, designers can determine a parameterized model for any size cell. The prediction model also includes additional routing area and wasted area due to pitch mismatching when DFT cells are added. Using a probabilistic model, Miles[17] considers the possible effect on global routing area overhead when DFT is added to a part or all of a device.

Figure 3 compares, using the Miles approach, between the overheads for a PLA with scan and a PLA with built-in logic-block observer (BILBO) registers applied to inputs and outputs. The results show that the relative overheads decrease as the PLA size increases, which is the expected trend.

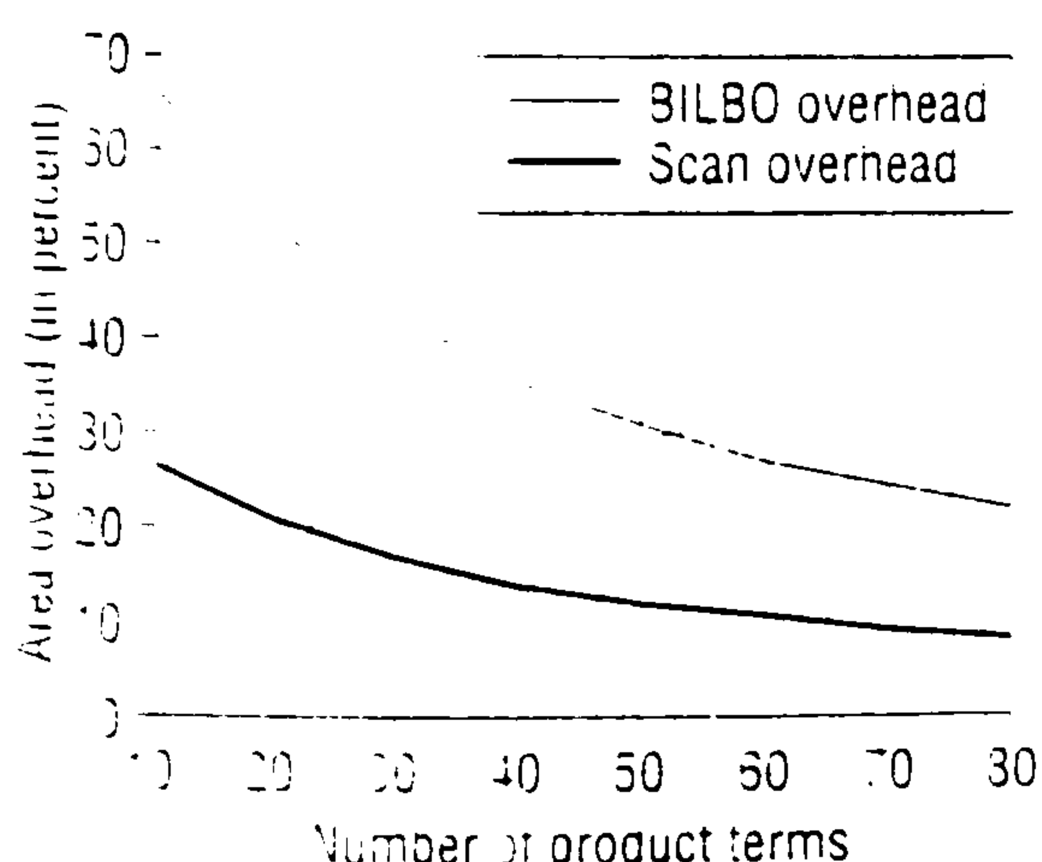


Figure 3. Area overhead predictions for scan path and BILBO test circuitry applied to PLAs (10 inputs, 10 outputs).

Simplification of production costs. In some cases, the complexity of the modeling process required can be reduced considerably. Companies commonly have specialist outside agents (silicon vendors) manufacture their semicustom chip designs. For a given production volume and package type, the production cost per unit is usually based on a simple gate-count model.

Of course, the actual costs quoted vary from manufacturer to manufacturer. If a company places many designs and or large production volumes through the manufacturer, lower costs per unit can be expected.

Figure 4 shows a typical production cost curve. The addition of DFT logic could increase the pin count beyond that allowed for a particular package type in the vendor's production model. Thus the appropriate adjustment must be made.

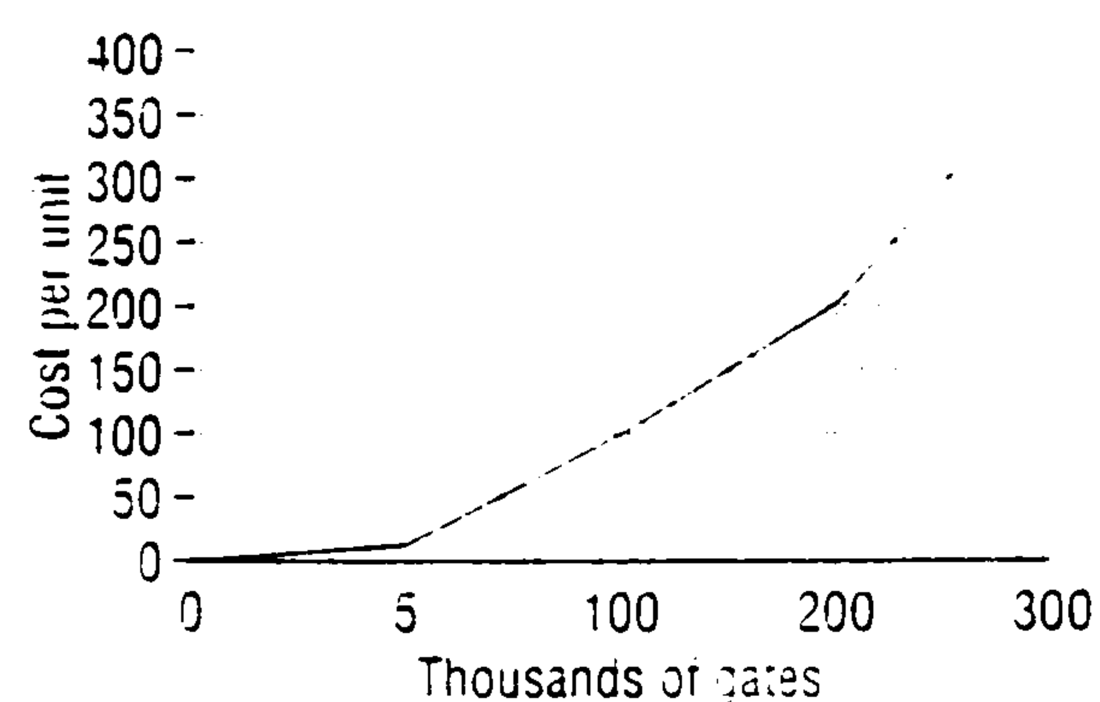


Figure 4. Manufacturing cost versus gate count.

Design time modeling

Several factors influence the design-time increase with the addition of DFT. Fey[18] and Fey and Paraskevopoulos[19] produced an empirical model from studies carried out at Xerox Corporation. The model calculated the efficiency of the design department based on a set of characteristics related to the design environment and the design itself. This work highlighted such factors as

- the designer's familiarity with the design
- design originality
- type of technology
- CAD tools available
- the gate count's effect on the design time

The DFT time penalty is often singled out as affecting time-to-market. However, if the design environment has been set up to encourage DFT and designers are skilled in that area, the penalties are much reduced.[1,20] Also, time-to-market depends not only on design time but also on test generation time. DFT methods may increase the time required for the first design, but they can reduce the time for test generation to almost zero if automatic test pattern generation tools are used. DFT methods also reduce the probability of redesigns entailed by untestable designs. Taking this argument to the extreme, the use of a structured DFT approach can often increase the efficiency of the design center and reduce the amount of rework required. The effect on design time and test generation time is important. With modern design systems, often the test department is the bottleneck.

Test pattern generation costs

Test pattern generation cost is a nonrecurring engineering cost. Thus, the smaller the production volume, the greater the effect this cost can have. DFT overhead cost is a constant, reoccurring cost per system or device. This is the main reason for the common observation that increasing DFT is more attractive for smaller production volumes.[12,21] The fall in production volumes can be linked to the increase in integration and consequent increased testing costs[20]—but don't forget the alternative cases[15] referred to earlier.

Although the calculation of accurate test generation costs is very difficult, it has always attracted a high level of interest. Goel carried out one of the first detailed studies of automatic test pattern generation.[22] He predicted the number of test vectors (effort) against fault coverage for ATPG and showed how test lengths can be associated with test generation cost. Goel produced an empirical model, but its ac-

plication was limited to combinational blocks. However, Varma, Ambler, and Baker[12] used this model to show that DFT methods still had an advantage when system test and field test are considered. They reinforced their argument by adopting very optimistic figures for devices with no DFT.

This type of model can be extended to take into account a scenario in which ATPG is performed to obtain the maximum level of fault coverage possible with specific ATPG systems, when sequential designs are considered. Such a model needs to account for both the limitations of the ATPG package and the characteristics of the design. As in the combinational case presented by Goel, the gate count, number of inputs and outputs, and the average fan-in or fan-out of the circuit can be good measures of the circuit size and complexity. For sequential circuits, the sequential depth and number of storage elements in the circuit can give a measure of the circuit's sequential nature. The limitations of the ATPG system used will be affected by the algorithm used, the amount of machine memory, the operator's skill, and so on.

Using historical data, we can predict the number of test patterns, test generation costs, and the maximum achievable fault coverage for a general design.[14] Historical data must be used from time to time to update such a predictive strategy.

A similar method can be applied to manual test pattern generation. Manual TPG is normally required to increase the fault coverage from the maximum achievable level obtained with ATPG to that required for the design, which is often about 98% fault coverage of all testable faults. Figure 5 shows a typical cost curve. In the general case, the increase in fault coverage is assumed to be linear with the number of manual test vectors or generation costs. The gradient affects the sequential characteristics of the circuit.

Such a general modeling approach must be treated with care. Historical data must be used consistently, and the limitations of such a general modeling prediction with specific designs must be understood. Normally, with the use of DFT such as scan path or BIST, a model can produce more accurate test generation cost predictions as the circuit's complexity is reduced. If pseudorandom or exhaustive tests are used, only fault-simulation costs are required. In some cases, fault-simulation costs can be considerable. If "canned" tests (prestored tests for commonly used circuit building blocks) are used, the cost of test generation and possibly fault simulation can be saved.

Test-application cost

The test-application cost is primarily affected by the number of test vectors per device, the number of devices, and the type of automatic test equipment (ATE) used. The type of DFT adopted can have a strong effect on the number of test vectors as well as the type of vectors to be applied. In some cases, the DFT method can impose restrictions on the type of ATE used or vice versa. The argument put forward for BIST is valid: the need for an inexpensive tester to act as control only for the self-test logic. However, the test time or throughput costs can still be the critical factor for high production volumes if exhaustive testing is performed by self-test logic.

Many trade-offs can be made between test lengths and DFT methods, and individual examples support such trade-offs. In general, the type of system or device and the production environment influence the way the trade-offs swing. For example, scan path testing results in application of long sequences of test vectors of narrow width. The width depends on the number of different scan paths and the amount of parallel test application possible. Thus, for scan path testing to be efficient, the ATE must have a large memory behind some pins with very fast application speeds. When this type of ATE is not available, the application time and cost remove scan path testing as a possible DFT approach, unless the production volumes are small. Investing capital to purchase new ATE is often a corporate decision and needs to be based on more than one design.[1-20]

The use of partial scan[24] often removes the need for an expensive scan path tester and reduces the total number of test vectors to be applied, thus offering a cost-effective test application solution. Also, the setup time for probes can be an important factor. If partial scan can reduce the problem of probing, significant production test time can be saved.

Prediction of test-application costs. Very accurate test lengths can be predicted when a set of test vectors is functionally independent, as is the case with exhaustive testing, pseudorandom tests, or canned tests, and with regular structures such as RAMs, where algorithms with well-known test lengths are used.

Test length prediction is crucial to predicting test-application times. As already mentioned, test lengths can be predicted in a way similar to TPG costs. Figure 5 shows a typical result for a sequential circuit. The ATPG system is an efficient way to produce test vectors to obtain initial fault coverage. The predictions of such a model shows the following: DFT reduces sequential depth so ATPG can attain high fault coverage. Thus, less manual TPG is needed.

We must consider any setup time the ATE requires before testing as well as the application time of all the test vectors. The

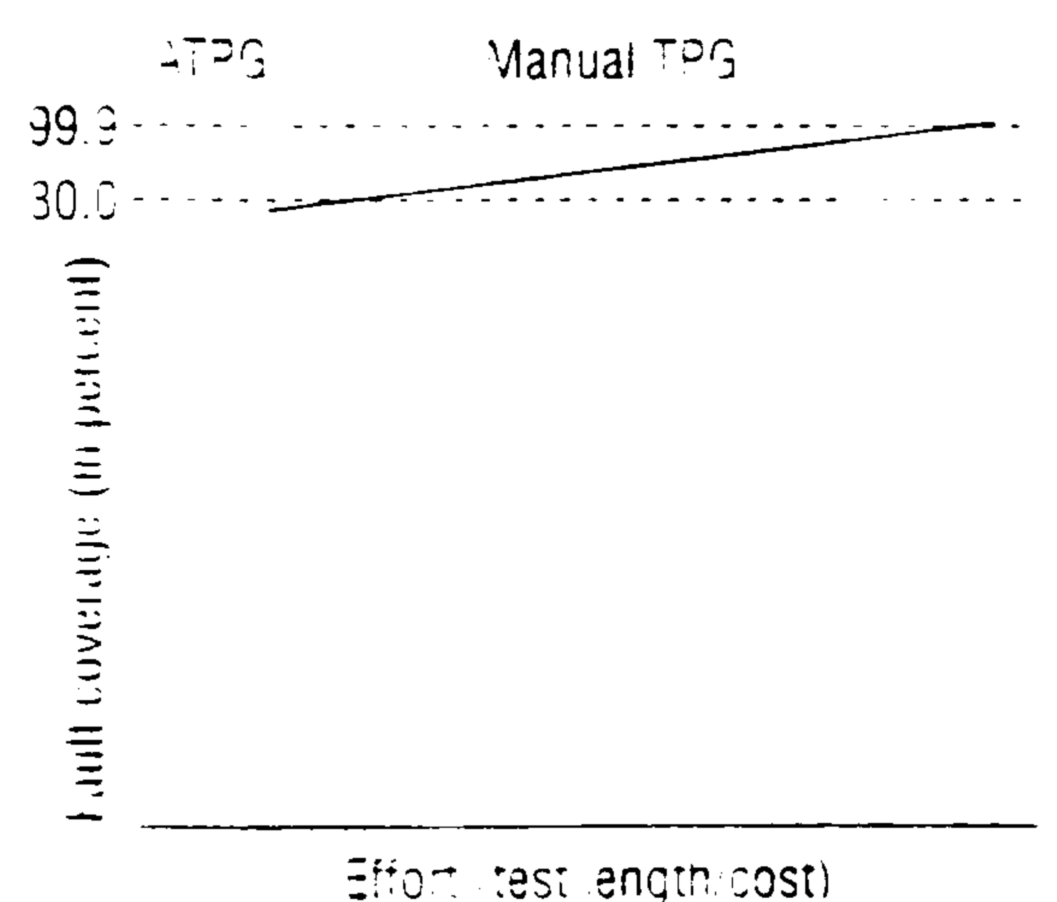


Figure 5. Fault coverage versus test effort (cost, test vectors).

application of test vectors is not a linear function of the number of test vectors. If all the test vectors can be loaded into the pin memory, the test-application time is negligible compared with the load time. If some test vectors need to be reloaded because the test vector size exceeds the pin memory size, the additional test-application time is steplike.

Once the total test time per device has been calculated, the application cost can be calculated from the hourly rate of running the ATE and the application speed of the tester. The running cost of the ATE should include labor and maintenance as well as the payback for the purchase of the ATE. The capital repayment will depend on the initial cost of the ATE and its working life. Thus, different companies may have very different test-application costs for the same devices.[1-23]

A practical simplification of test-application costs. To obtain an accurate accounting of the test-application costs for a design based on the number of test vectors, we can extend the argument for using outside manufacturers of devices from a production cost case to test application costs. A manufacturer may have several different ATEs, and the one used for the particular design can affect the cost curve. The customer's bargaining position will also influence the cost per unit.

Figure 6 shows an example cost curve. As can be seen from the curve, in many cases the manufacturer gives a certain amount of test vectors free and then charges a fixed cost per batch of test vectors. Often this batch size represents the memory capacity of the ATE used, for the reasons discussed earlier.

Overall economic effects of DFT

Consideration of the issues discussed should help in making objective DFT decisions. For example, a common question concerns the maximum area overhead affordable for a given test method. Figure 7 shows the result of a sample analysis of the problem. The results are normalized with respect to the cost of the unmodified design. The trends are specific to the design analyzed, and we show them here only to illustrate the usefulness of cost modeling. In the economics modeling, we used the simplifications made to the prediction of test-related factors.

Figure 7 is based on a combinational block containing 90 gates. The block occupies 0.001 sq cm of a 0.25-sq cm IC with a production volume of 1,000 ICs.

In this case, when self-test is used, the maximum affordable area overhead before self-test becomes uneconomical is around 60%. This is the crossover point. The crossover point for scan is only around 35%. This difference decreases with increasing production quantities (in line with predictions[12]) as the costs in the test generation sector become less significant.

We pointed out previously that the overall chip size affects the cost effectiveness of DFT methods for the circuit, because of reduced yield. However, in the test sector, there are larger savings in a large circuit. The cost effectiveness of DFT then depends not only on the number of devices produced but also on the size of the device and the area overhead of DFT methods contrasted with savings in test costs. To illustrate this point, we evaluated some ASIC designs.

We considered a range of area overhead values for this analysis.[13] The over-

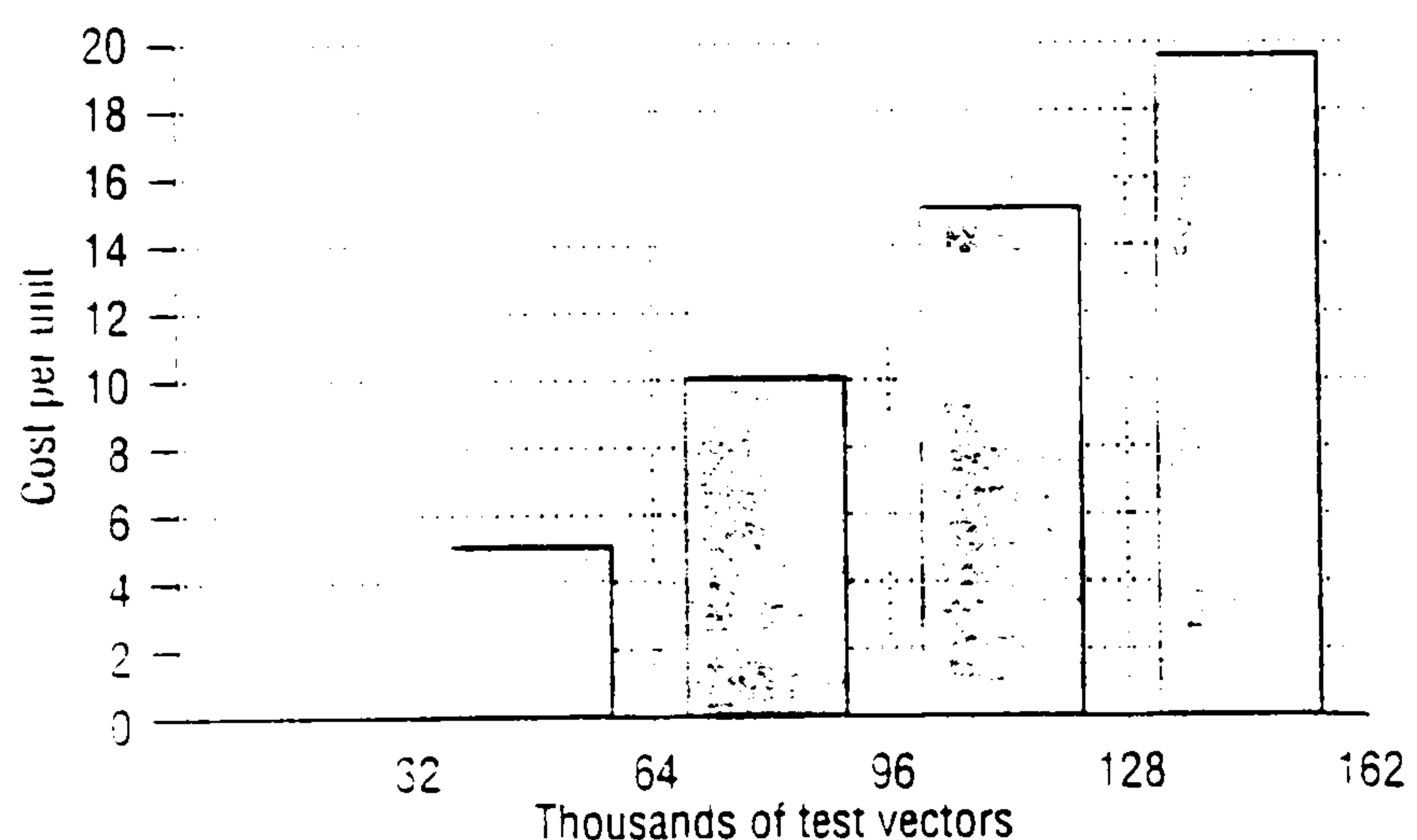


Figure 6. Test length versus application cost per unit.

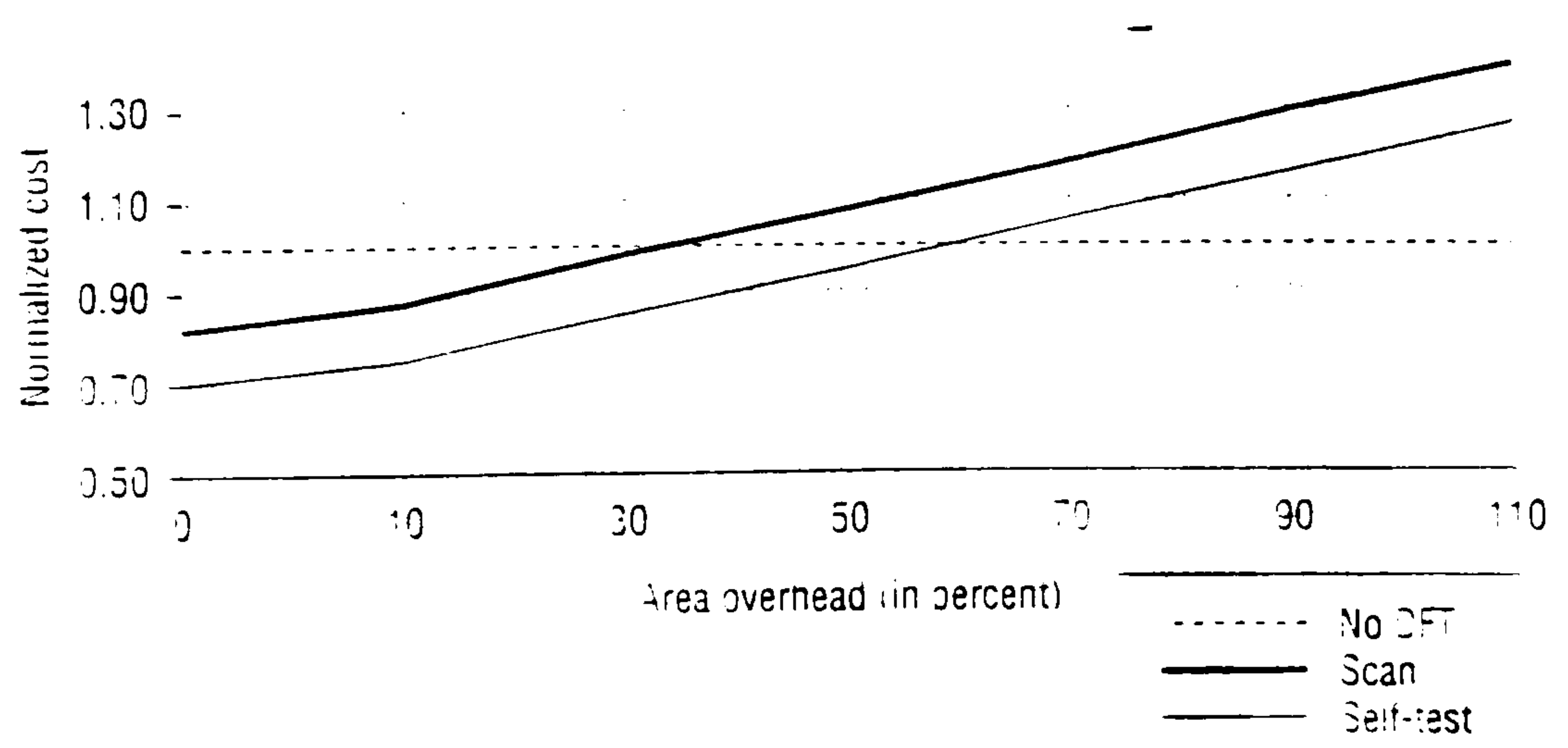


Figure 7. Cost versus area overhead for different DFT methods, assuming no overhead for no DFT and a production volume of 1,000 ICs.

head for scan was between 5% and 15%, and for self-test between 20% and 30%. Costs were again normalized to the case without DFT. The difference between the high and low overhead cases widens as the size of the chip is increased, because of reduced yield.

In the case of self-test, the crossover point (the number of chips at which self-test ceases to be cost-effective) increases as the gate count increases from 5,000 to 10,000 gates and decreases again as the gate count increases to 20,000 and then to 50,000. The reason is that as the complexity of the circuit increases, test generation costs also increase. However, because the chip size increases as well, at one point the increased manufacturing costs become higher than the savings in the test sector. In the case of scan, this effect takes place earlier, as the savings in test costs are not as high as in the case of self-test. Figure 8 shows the status for a 10,000-gate, 0.80-sq cm device; Figure 9, for a 50,000-gate, 2.00-sq cm device.

These examples for chip design, production, and test only do not take into account the potential benefits that can be accrued through reduced board and system test as a result of improved chip testability. The benefits to board and system test can be considerable.

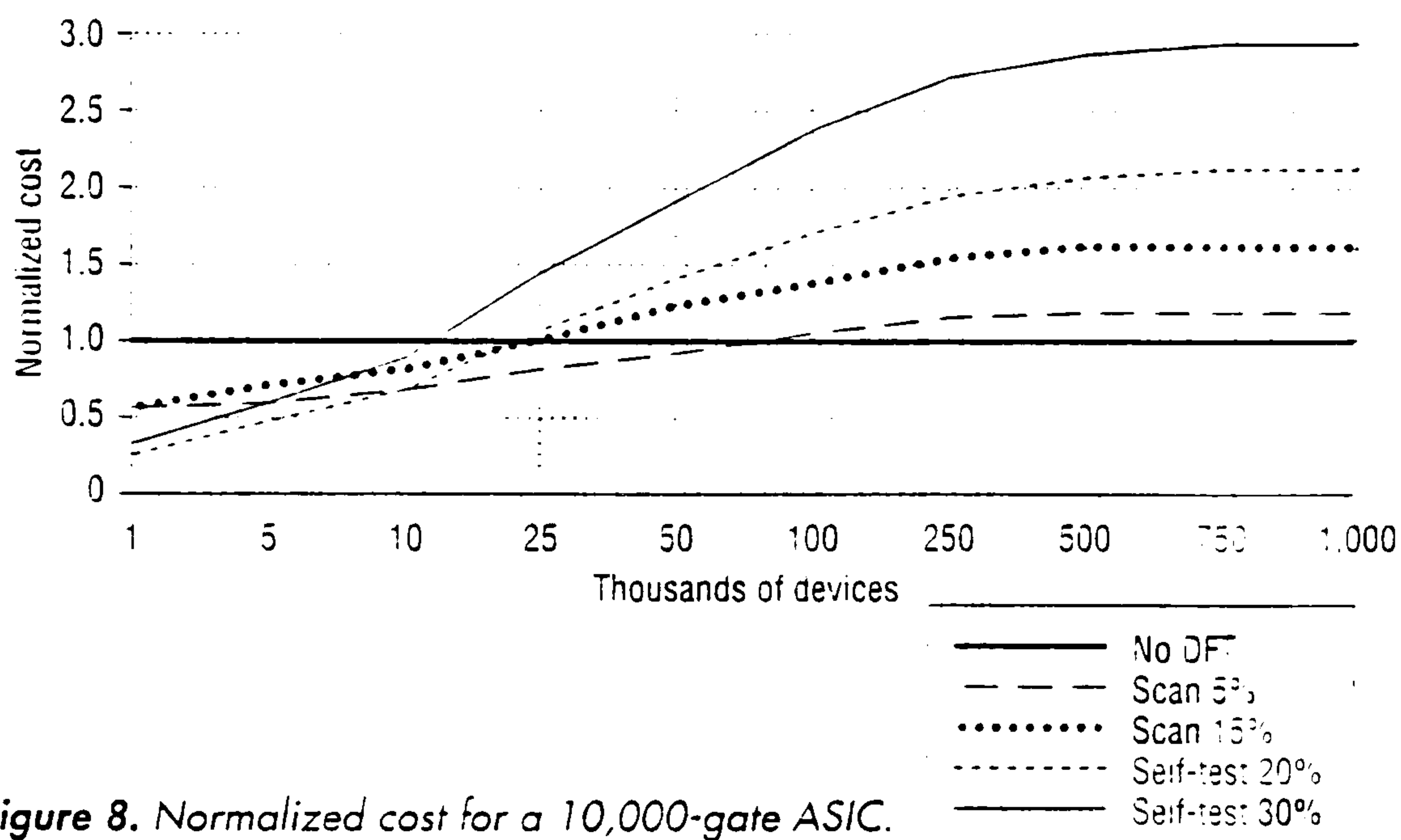


Figure 8. Normalized cost for a 10,000-gate ASIC.

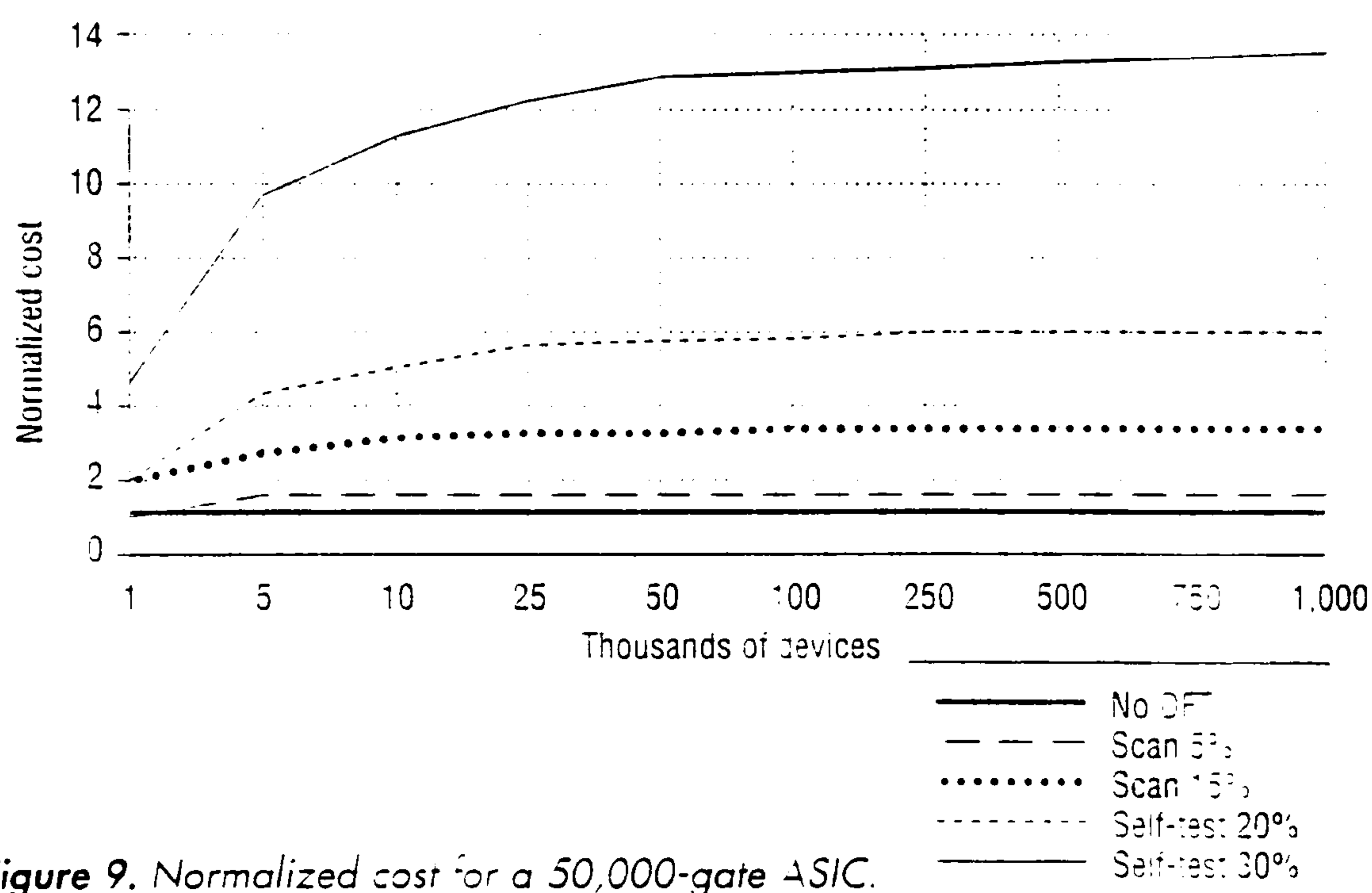


Figure 9. Normalized cost for a 50,000-gate ASIC.

EVEREST test strategy planner

Economics modeling can also be used for more specific DFT decision making, as an aid to designers seeking the optimum mix of DFT methods for a particular application. A system for such economics modeling is the EVEREST test strategy planner, which is a practical implementation of the BEAST system.[25,26] (BEAST was developed under an Alvey scheme—Alvey is a UK-sponsored information technology program.) Both test strategy planners have aims similar to those of the Tiger system.[27] However, instead of using a weighting system specified by the designer, both planners use economics modeling to help the designer make informed DFT decisions. Thus, designers do not need to understand the relative importance of all the test-related parameters.

The BEAST system was designed to work in a macrocell (RAM, ROM, PLA, and so on) environment, but its cost modeling was purposely left as general as possible. The system successfully showed that designers can use economics to select between test methods. The major problem with the system was obtaining the cost data to make it company and product specific. The required design and test parameters (for example, yield and test quality) are obviously commercially sensitive—to the extent that often departments within the same company have problems exchanging meaningful data. However, the system did show that the results of economics-based test strategy planners are sensitive to different companies' manufacturing and accounting procedures, as well as to different test strategy plans.[14,26]

The natural next step was to integrate a general-purpose system such as BEAST into a company's CAE environment and use in-house costing procedures. The EVEREST test strategy planner addressed the practical problems of implementing a usable system. We carried this work out in collaboration with Siemens-Nixdorf as part of a project sponsored by the European Strategic Programme for Research in Information Technology. We linked the resulting economics-driven test strategy planner into the Siemens-Nixdorf CAD environment using accounting information generated in-house,[14] so obtaining necessary cost data has not been a problem.

Test strategy planning system

Figure 10 outlines the test strategy planning system. The design description can be acquired either from an existing netlist or directly from the user. The description itself is hierarchical to allow test strategy decisions at several stages of the design process. A variety of essential economic data is also acquired at the same time.

The data automatically updates an economics model before test strategy selection. The test strategy planner uses stored knowledge on test methods as well as the design data and the existing cost model to evaluate a variety of test strategies. The test strategy control can either be left entirely to the user or automatically planned to accelerate the selection.

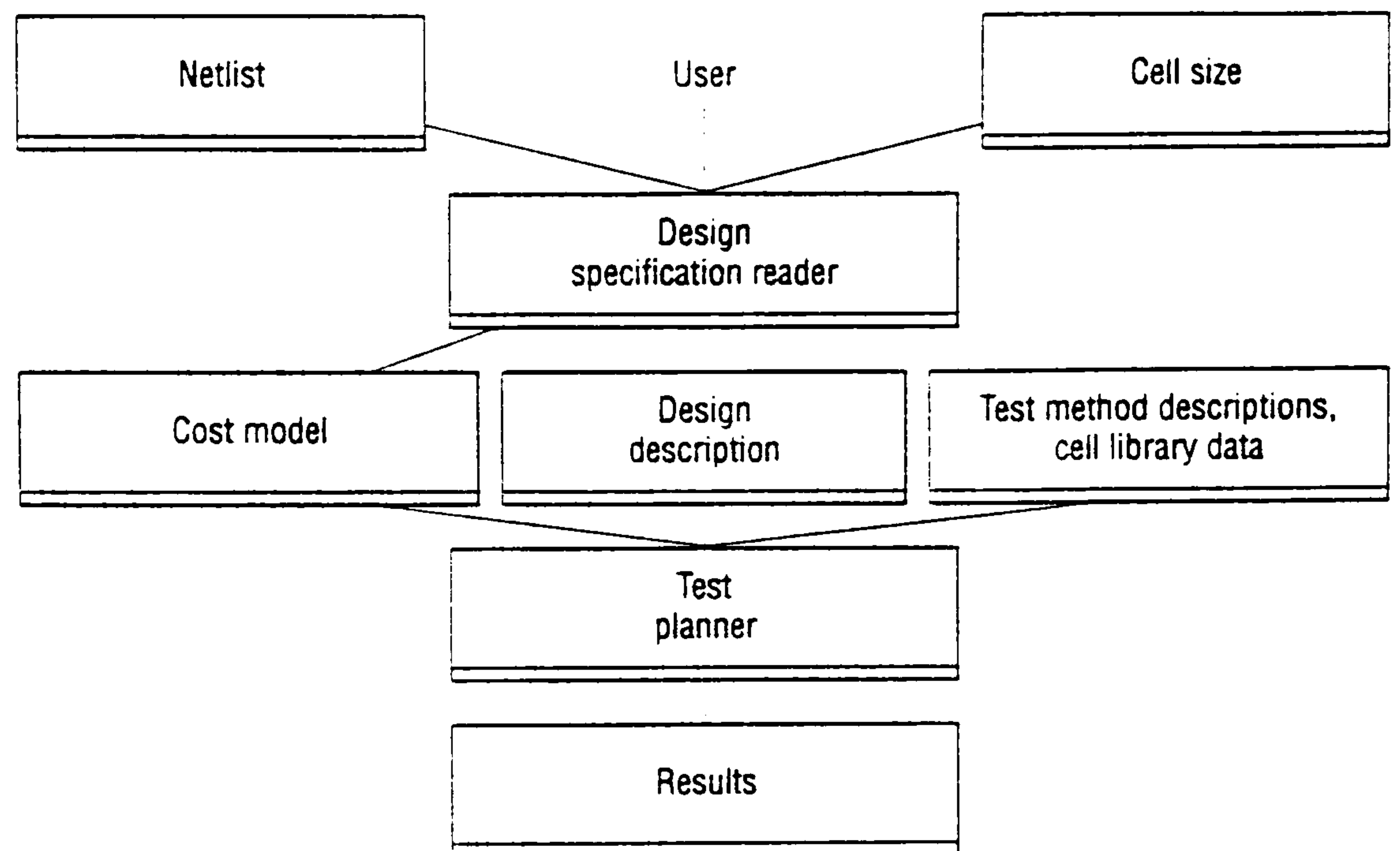


Figure 10. EVEREST system architecture.

Table 1. Cost model.

Overall cost		
Design	**Production**	**Test**
Engineering	Production unit	ATPG
Design complexity	NRE* charges	Manual TPG
Productivity of design environment		Test application
Equipment		
Design center		

*Nonrecurring engineering

Cost modeling techniques

The economics model used in EVEREST was based on previous economics modeling work[12-13] but was tailored to the needs of semicustom cell design. The cost model categorizes primary parameters supplied by the user into the following factors: design independent (mostly global company accountings), design dependent (which will change the design), test-method dependent, and constant. The system then evaluates a set of equations using the supplied values. Table 1 lists the main parts of the model.

Design costs are modeled in terms of design time and equipment required, the cost of using an external design center if this option is taken, and manpower, which is a function of the complexity of the system and the productivity of the design environment. Productivity is modeled in terms of the designer's experience, CAD system performance, and cell library functionality.

Production costs are linked to the design complexity (measured in gate equivalents, grids, or square millimeters), mostly with a linear relationship within certain complexity ranges. In this model, gate equivalent count was used as a complexity measure. A test strategy may influence the gate count and therefore the production cost per unit. In addition, nonrecurrent engineering charges must be added to the production cost. Unlike in previous models,[12,13] yield effects are not modeled separately, because they are included in the vendor's pricing policy.

Test costs are separated into test pattern generation costs and test-application costs. Test-application costs are linked to the number of test patterns in linear or stepwise ranges, similar to the way gate count is linked to production cost. This pricing is normally provided by the vendor and removes the need for the separate modeling of ATE costs present in previous models. Test pattern generation costs are estimated as follows: The cost of ATPG is estimated, together with the maximum achievable fault coverage. If the achievable fault coverage is less than the required value (a fundamental requirement), the cost of achieving it using expensive manual TPG is estimated. Using structured DFT methods often means that ATPG techniques alone are effective for producing the required fault coverage.

The model is coded in a manner that makes it easy for the user to alter. Changes are made in two ways

- During test strategy planning, the user can examine and alter individual parameters to evaluate their effects.
- For more permanent changes (for example, to reflect a change in the vendor's pricing), the user can simply change the "generic" (or template) cost model files using a text editor, because the files are stored as text.

Relatively few of the parameters in the model need to be changed from design to design. Those that need changes can easily be obtained from the CAD tool, design, and test database.

Test method categorization and description

The test method description database needs to take into account mainly the economic effect of test methods, so that an economic evaluation can be produced. But it should also assess the suitability of test methods for the particular design. Parameters are described as single values or text strings, or as equations. The parameters to be described are categorized in the three groups listed in Table 2.

The first group of parameters is used in the cost model for an economic evaluation of the method. The performance complexity and the originality parameters are related to design costs, as is the number of extra functions (if any) introduced by the test structures. The pin compatibility information is used in the calculation of the final pin overhead of the plan.

In the design implications group of parameters, the test method type must be specified. At the end of the test strategy planning process, a test strategy must make every block testable (on its own) and every block I/O line accessible, so test patterns may be propagated to it. This is a requirement of the system. The test methods are therefore categorized as testability enhancing (internal) or accessibility enhancing (external).

For example, some self-test methods make a block testable but not accessible. Also, methods such as scan path only on the I/O lines provide accessibility but do not improve the actual testability of the block. There is some overlap in this categorization, as some testability-enhancing test methods (for example, certain scan options) also provide accessibility to the block. The impact on the accessibility of the block I/O lines is also stored here, because it is useful in guiding the test selection algorithms.

The third group contains the design requirements of the test method. For a test method to be applicable to a block, it must be suited to the type and style of the design and also must fulfill basic requirements in terms of fault coverage, maximum gate count, and pin count. Data in this section is used to ensure that design requirements are met.

Example of test method description

The following example shows the coding of the test method description. The scan path method enhances the testability of random sequential designs. It does

Table 2. Test method description.

Group		
1: Cost model parameters for economic evaluation	**2: Design implications**	**3: Design requirements**
Equivalent gate count	Accessibility impact	Suitable design class (PLA, RAM)
Sequential depth	Test method type (testability/ accessibility)	Suitable design style (synchronous, flip-flop design, latch design)
Performance complexity		
Additional pin count		
Originality		
Number of extra functions		
Number of test patterns Achievable fault coverage TPG method		
Pin compatibility (possible shared use of test pins)		

not necessarily improve the accessibility to the block I/O lines. Table 3 summarizes the information.

When a test method is applied to a block, its design and economic effects are evaluated. The test method is first checked for compatibility with the design type and style and to ensure that it falls within user-defined limits. If this check is successful, the economic effects are evaluated by using the test method description parameters for the cost model, taking into account any cell library support for the method and also the possibility of testing a number of functional blocks in parallel.

Once the method is applied, its effects on block accessibility are also noted, and the accessibility of other block I/O is recalculated. The recalculation is based on existing information about transparent "paths" through functional blocks. This recalculation and the cost model are used to assess the global as well as local implications of the test method. Testability- and accessibility-enhancing methods are applied separately. Changes to the design description are not permanent but are current only for the duration of the test strategy planning process. The designer must implement the test structures suggested by the test strategy planner.

Table 3. Example of test method description.

Parameter name	Parameter value
Test method name	Int_scan
Test method type	Internal
Suitable design classes	Random sequential
Self-test	No
Assures accessibility	
Data-in	No
Control-in	No
Clock-in	No
Out	No
Bus	No
Performance implication	1.1
TPG method	Combinatorial ATPG
Sequential depth	0
Achievable fault coverage	Calculated by the cost model
Number of test patterns	Calculated by the cost model
Overhead formula	2.5 × gate count of 1 DFF*
Originality impact	1
Overhead	
In-pin	2
Out-pin	1
Bi-pin	0
Pin compatibility class	1
Design style requirements	Synchronous flip-flop design
Number of functions	0

*D-type flip-flop

Test strategy planning

The system provides the necessary functions for the user to evaluate a variety of test strategies and also to use a degree of automated planning. The user can examine the cost implications by examining the cost model. At the end of the process, a test strategy description is produced, with details about the test method chosen for every block, together with a complete copy of the cost model for later reference. The design description itself is not altered.

Example of test strategy planning

The circuit example in Figure 11 (on the next page) illustrates the planning process. The design description is fully hierarchical down to the gate level, but here only two levels of hierarchy are used: the overall circuit and the circuit subblocks. These are described in terms of functionality (allowing suitable test methods to be chosen), as well as connectivity, with attributes to identify control, data, and bus lines. In this case, the blocks are a PLA with 24 inputs, nine outputs, and 50 product terms; a 4-Kbyte RAM; and two sequential random logic blocks of 5,000 gates each. Information is also given on block transparency, which allows the system to derive the accessibility of lines for test purposes. The objective of automatic test strategy selection is to ensure that all blocks are individually testable and that all lines are accessible, so test patterns can be propagated to the testable blocks.

Figure 12 shows the results of the test strategy plan at different stages for a production volume of 50,000 units. In this case, four PLA methods[illegible] were put through the cost model. The one chosen was a self-test method using cumulative parity comparisons.[28] The RAM test methods included different implementations of standard

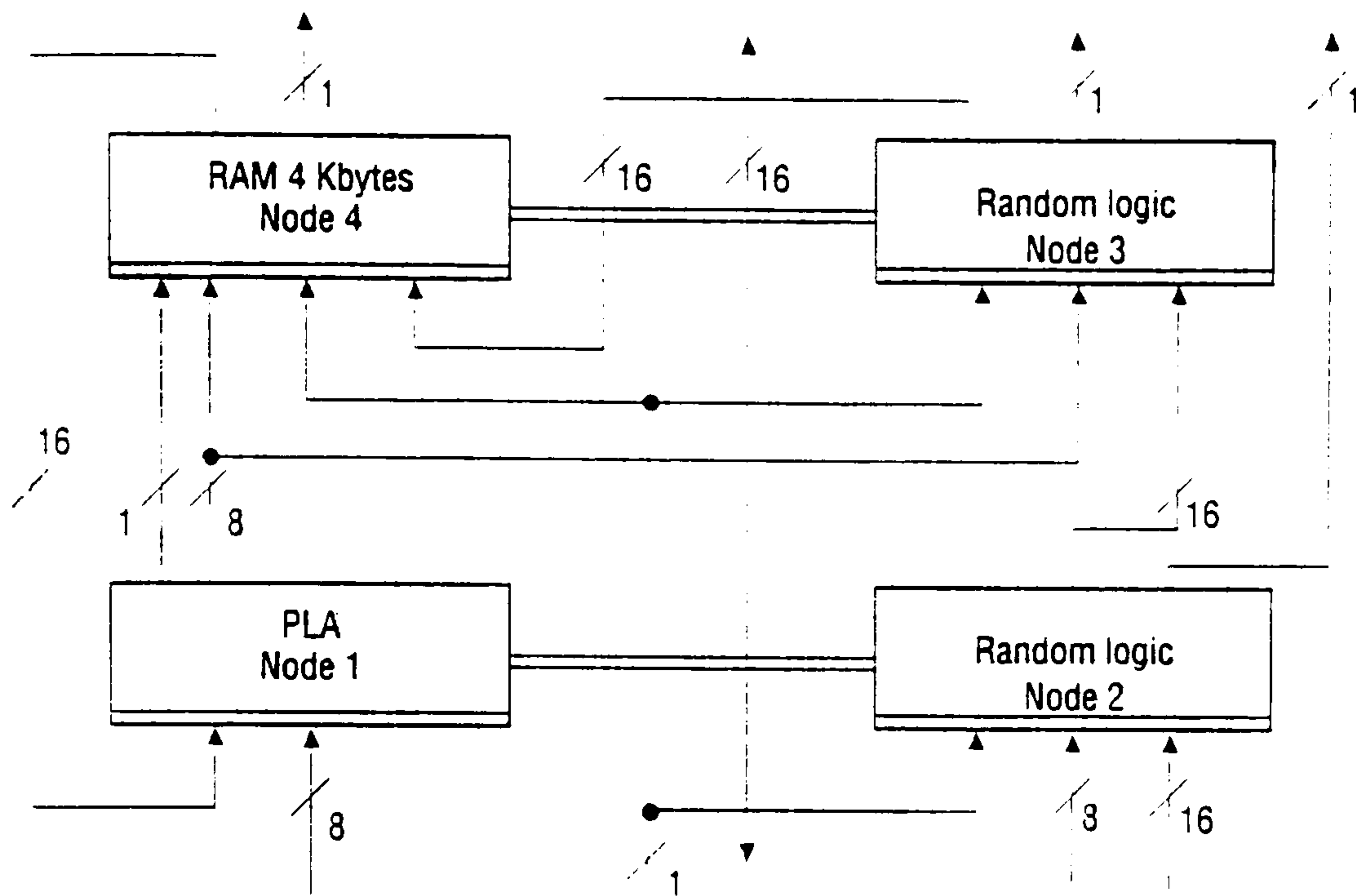
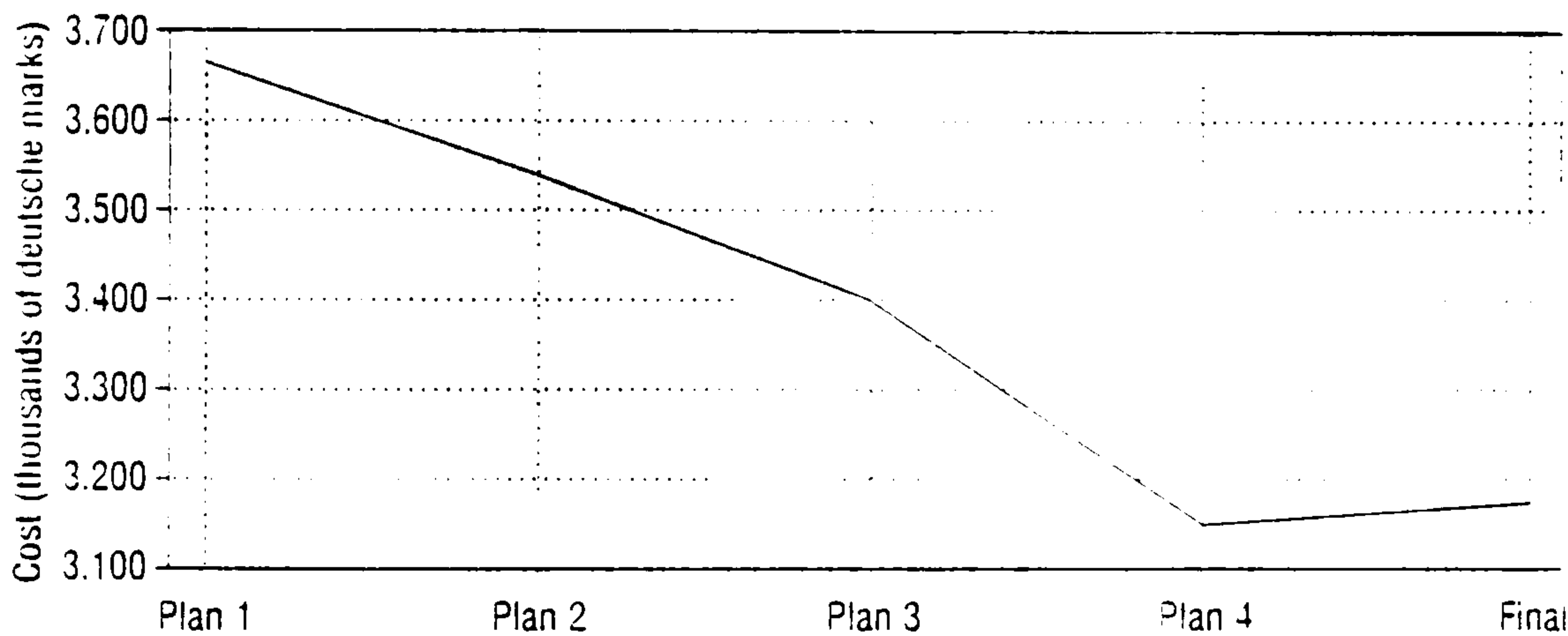


Figure 11. Circuit example.



(a)

Plan 1. Node 1: Treuer,[28] node 2: no DFT, node 3: no DFT, node 4: no DFT
Plan 2. Node 1: Treuer, node 2: int_scan, node 3: no DFT, node 4: no DFT
Plan 3. Node 1: Treuer, node 2: int_scan, node 3: int_scan, node 4: no DFT
Plan 4. Node 1: Treuer, node 2: int_scan, node 3: int_scan, node 4: illman[29]
Final. Node 1: Add external scan to node 2 to fulfill accessibility requirements.

(b)

Figure 12. Test strategy selection for production volume of 50,000 ICs: Results at different stages (a) and plan descriptions (b).

memory test algorithms as well as pseudorandom methods. The method chosen was a pseudorandom method based on linear-feedback shift registers.[29] Internal scan was used as the most cost-effective method for making the sequential random blocks testable. An evaluation of accessibility after the inclusion of the testability-enhancing methods revealed an accessibility problem on the group of lines connecting node 2 to node 3. An external scan path, which is classified as an accessibility-enhancing method, modifies this problem.

The accessibility method actually incurs a financial penalty, but this is because accessibility improvement is a requirement of the algorithm and is not directly accounted for in the cost model. Cost calculations are made on the assumption that the blocks are fully accessible for test purposes. The financial penalty of adding an accessibility method is comparatively less for larger production volumes, principally because the relative importance of extra development costs decreases with increasing production volume.

If the production volume increases to 250,000, the plan changes at stage 4 with a different test method for the memory element being chosen. In this instance, a self-test implementation of the March algorithm proved more cost-effective. This illustrates that the test strategy selected is sensitive to production volume.

We have tested larger circuits in the system. Automatic test strategy planning for a design of 250,000 gates with six combinational logic blocks, two RAMs, and one PLA takes 7 to 8 minutes running on an Apollo 4500 workstation. The runtime depends on connectivity complexity rather than gate count. The results for this circuit show trends similar to those for the previous circuit. This circuit has been used to investigate heuristics for performing quicker and more effective test strategy plans.

The EVEREST test strategy planner is currently being evaluated at Siemens-Nixdorf and is to be used by other collaborating companies under the EVEREST project. It will become a commercial product next year.

Related issues

A straight economic analysis of test-related factors can be a very useful and illuminating task, as we have shown. Previous presentations of the material have been well received by those practicing good DFT; it confirmed their belief with hard evidence of the benefits of test and by others who had not considered DFT because of their perception of likely test costs. However, in our opinion other factors cannot be measured accurately

enough, if at all, in any economic calculation. Nevertheless, they must be considered along with the measurable factors.

The effect on the company's image—how consumers and the public perceive its concern with quality—*must* be a major force in any considerations about a product's testability. For example, if an economic analysis showed that 70% fault coverage was acceptable on cost grounds, consider the effect on the company image when that inadequately tested device or system caused the destruction of, say, a civil airliner. Clearly, other factors can far outweigh cost considerations!

Marketing must also be considered in any cost-effect analysis. We have not discussed it here, but considerations of marketing are available elsewhere.[30,31] Such factors are important and can greatly affect the costs and likely productivity of the finished product, both in terms of defining the original specification and time-to-market issues. A design change or enhancement that results from testability improvements may increase the time required to get a product to market. On the other hand, the product may be delayed in getting to market because testability hasn't been designed in. Some figures suggest that shipping six months late can lead to a 33% reduction in potential profits under certain market conditions.[32] This is in addition to the other negatives of not incorporating testability!

There is also the issue of whether a merchant chip manufacturer has any interest in providing improved testability that benefits the user or customer. The economic issues in these cases are interesting. How much extra can the customer be charged for a new design "feature"? (Adding testability for the new feature increases its expense.) Another argument is that pushing the bounds of technology when a new device is being designed precludes the incorporation of testability because testability reduces potential functional area. But with no testability designed in, how can the customer adequately test its new circuit?

Not all environments are straightforward enough for economic analysis in the way we describe here. Dear, Ambler, and Maunder[33] discuss a field service environment in which intangible factors can be quite prominent. An example is the "macho" approach—a field service engineer merely replaces all boards in a system to make it function correctly in the shortest possible time, without attempting any diagnosis. This inevitably leads to expensive redundant analysis of working boards before they can be returned to stock. Although difficult to quantify, the effects of environment are significant and must be accounted for.

Board and system test

Although we have not addressed in any detail issues relating to board and system test, there have been significant efforts in this field.

The EVEREST work is now moving in this direction[34] and will allow for economics-based test strategy planning at the board and system levels. Previous work that touched on board and system test[12,33] does not appear to go into sufficient detail. Dislis et al.[13] showed results that go some way toward meeting the requirements with an analysis of the potential effects of boundary scan at the board level (see Figure 13). They discuss the issues involved in a lot more detail than we can here and include further results showing that the benefits of full scan versus boundary scan can reverse with increasing production quantities. Nevertheless, economic analyses can be made only when comparing like with like. With a board or system designed for boundary scan—so there is no other easy way to test adequately—no comparison can be made, economically or otherwise.

Board and system level analysis will have major implications for ASIC test economics. Some results show that the use of BIST for chip test only is potentially more expensive than other methods (again, results depend on complexity, production run, and so on). But use of these same devices at the board and system levels can dramatically reduce the overall test costs (through the "rule of tens" effect), as Figure 14 on page 76 shows.

Other applications

Test is not the only area in which economic measures aid decision making. The whole area of design and subsequent manufacture must not be ignored. The requirements of concurrent engineering are fer-

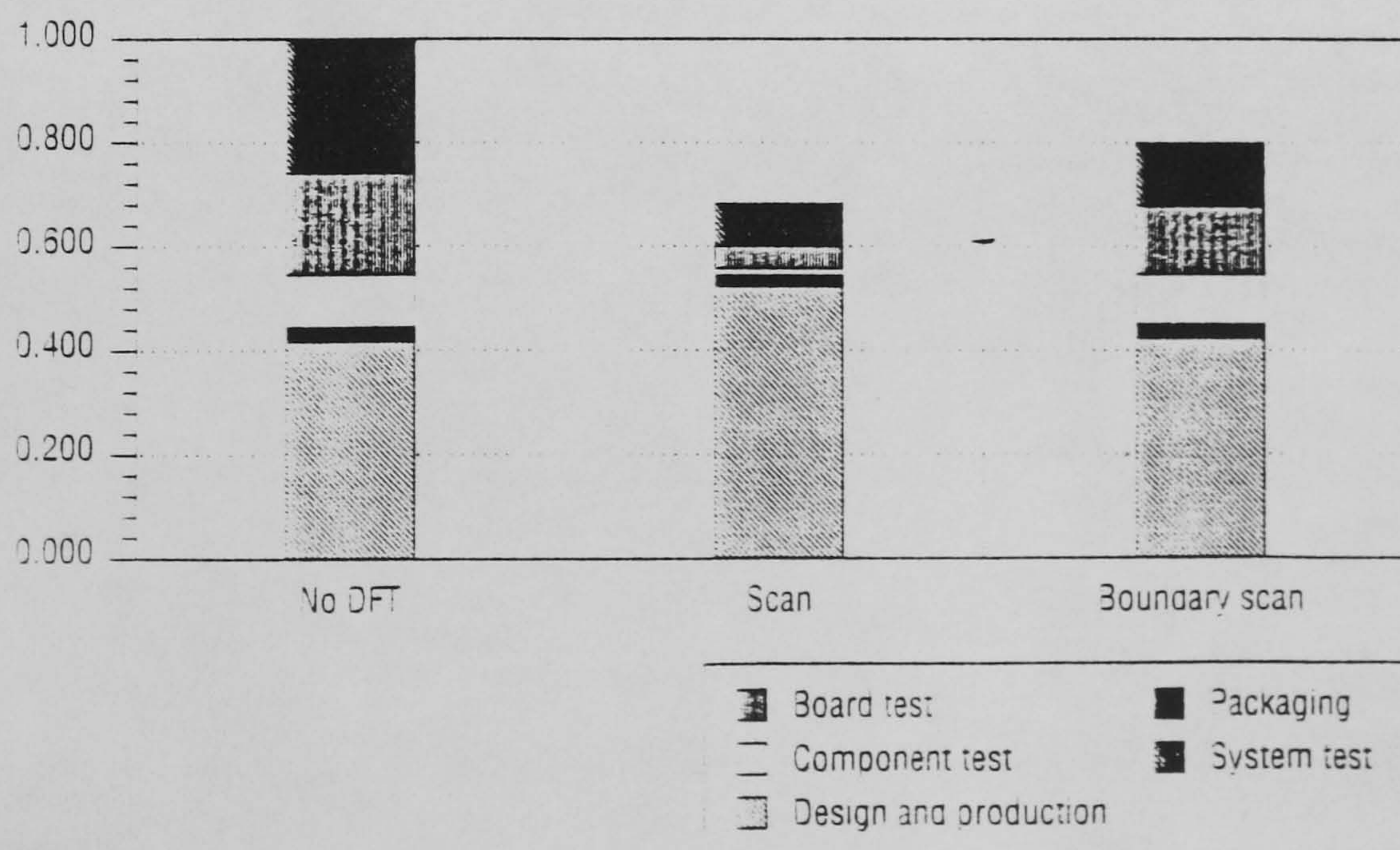


Figure 13. Economic effects of board test strategies: normalized cost for 5,000 systems.

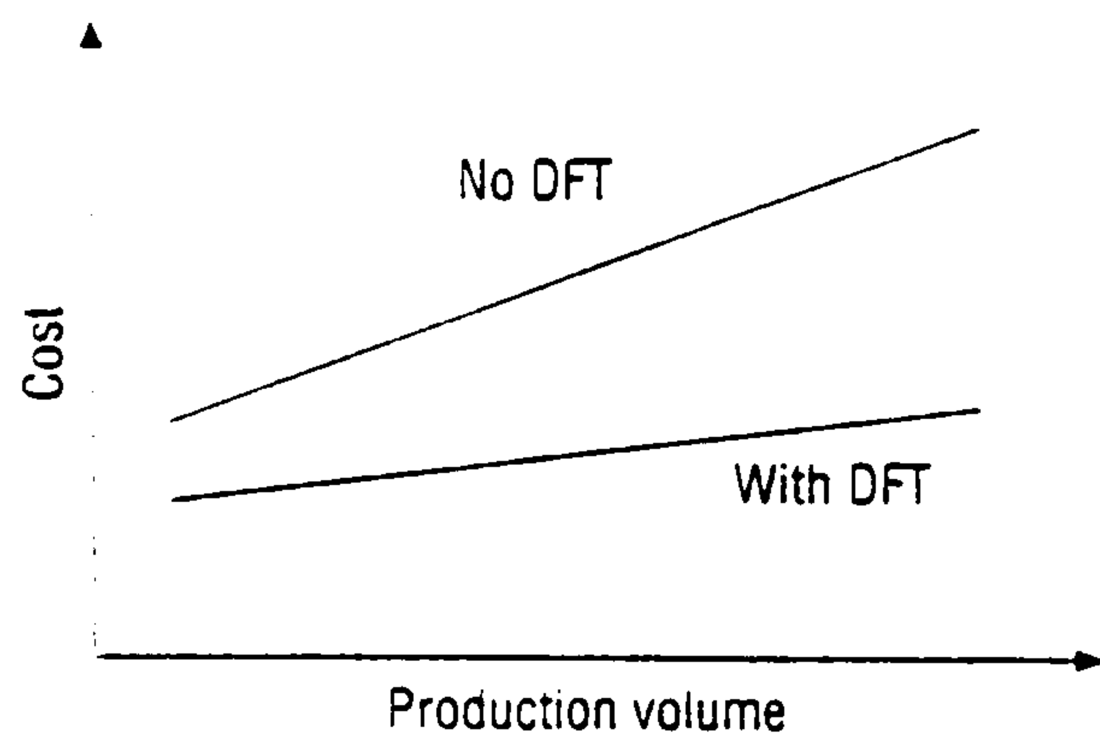


Figure 14. Cost comparison of possible board test strategies.

tile grounds for extending the methods we described. As with the test analysis, we can visit, revisit, and evaluate new technologies, new design techniques, new manufacturing methods, *and* old discarded ones. We are currently doing such work. Not only will we try to predict the final cost; we will also evaluate the potential means for comparing quantitatively disparate values.

We have presented some economic issues that relate to test decisions made during the design cycle. Our discussion focused on ASICs, and we have not addressed the wider issues of board and system test that are just as important—if not more important—to the overall economic picture.

We have shown that economic data can be very valuable in decision making at the design level and that the data can be used in an industrial environment. In its present form, the economics model in the EVEREST test strategy planner requires a lot of data and some setup time. In the collaborative programs that we have been working in, these potential drawbacks have not been sufficient to prevent our industrial partners from doing what is required—the benefits achieved can be large.

The economics model needs to be much simpler to make the EVEREST test strategy planner marketable for widespread commercial use. To this end, we are conducting a sensitivity analysis to create a reduced data set sufficient to achieve reasonable results, and also to indicate the bounds under which the reduced data set would be valid. However, the complete EVEREST model would be available to those who prefer it.

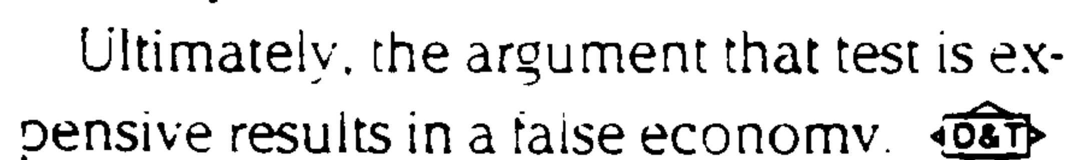
Economics models are obviously useful, but they can be difficult to create: How do we verify a model's completeness and accuracy? An effective database requires efficient management, but comparative evaluation can be used effectively. Such an approach is equally applicable to other design-related areas in which decisions need to be made. How easy it is to adapt the approach must be reconsidered in each individual case.

Other approaches to testability can and are being extended to board, system, and field test. Information about the financial impact of field test could be fed back to the designers to allow them to consider alternative and improved test features. Even better, predictive tools that look forward to likely field failures could provide data for use at initial design time, eliminating the need for rework some time after product release.

We emphasize once again that the results shown here apply to specific cases and cannot be considered generally applicable in terms of the absolute numbers shown, breakpoint positions, or even broad trends. The design environment, technology choice, equipment, and geographic location affect the data used in any model. The model itself is not sacrosanct either.

The detailed model needs to be fine-tuned to each individual location and requirement, not only because of technical differences but also because of differing accounting procedures. Nevertheless, a model can be used as the basis from which to build a more specific example, and this has been done easily and successfully in a number of cases.

Ultimately, the argument that test is expensive results in a false economy.



Acknowledgments

We thank the United Kingdom Alvey Directorate, the European Economic Commun... ESPRIT Directorate, and Siemens-Nixdorf ... their generous financial support of this wor...

References

1. B. Davis, *The Economics of Automatic Testing*, McGraw-Hill, New York, 198...
2. C. Fey and D. Paraskevopoulos, "Economic Aspects of Technology Selection: Level of Integration, Design Productivity, and Development Schedules," in *VLSI Handbook*, J. Di Giacomo, ed., McGraw-Hill, 1988, pp. 25.3-25.26.
3. C. Fey and D. Paraskevopoulos, "Economic Aspects of Technology Selection: Costs and Risks," in *VLSI Handbook*, J. Di Giacomo, ed., McGraw-Hill, 1988, pp. 26...-26.20.
4. J. Turino, *Design to Test*, second ed., Van Nostrand Reinhold, New York, 1990.
5. J. Huber and M. Rosneck, *Successful ASIC Design the First Time Through*, Van Nostrand Reinhold, 1991.
6. E. McCluskey and F. Buelow, "IC Quality and Test Transparency," *Proc. IEEE Int'l Test Conf.*, IEEE Computer Society Press, Los Alamitos, Calif., 1988, pp. 295-301.
7. M. Abramovici, M. Breuer, and A. Friedman, *Digital Systems Testing and Testable Design*, Computer Science Press, Rockville, Md., 1990.
8. I. Dear, "A Test Strategy Planning Methodology Driven by Economic Parameters," PhD thesis, Brunel University, Uxbridge, UK, 1991.
9. B. Wilkins, *Testing Digital Circuits*, ... *Introduction*, Van Nostrand Reinhold, 19...
10. B. Dervisoglu, "Using Scan Technology for Debug and Diagnostics in a Workstation Environment," *Proc. IEEE Int'l Test Conf.*, IEEE CS Press, 1988.
11. X. Zhu and M. Breuer, "Analysis of Testable PLA Designs," *IEEE Design & Test of Computers*, Vol. 5, No. 4, Aug. 1988, pp. 14-28.
12. P. Varma, A. Ambler, and K. Baker, "Analysis of the Economics of Self-Test," *Proc. IEEE Int'l Test Conf.*, 1984, pp. 2...
13. C. Dislis et al., "Cost Analysis of Test Methods in Environments," *Proc. IEEE Int'l Test Conf.*, 1989, pp. 875-883.
14. C. Dislis, J. Dick, and A. Ambler, "Economics Based Test Strategy Planner for VLSI Design," *Proc. Second European Test Conf.*, VDE-Verlag, Berlin, 1991, pp. 437-446.
15. A. Ambler et al., "Economically Via Automatic Insertion of Self-Test Featu...

for Custom VLSI." *Proc. IEEE Int'l Test Conf.*, 1986, pp. 232-243.

16. J. Miles, A. Ambler, and K. Totton, "Area and Performance Estimation of Design for Test Methods," *Proc. IEEE Int'l Conf. Computer Design*, 1988.
17. J. Miles, *Cost Modelling for VLSI Circuit Conversion to Aid Testability*, doctoral dissertation, Brunel Univ., Uxbridge, UK, 1988.
18. C. Fey, "Custom LSI/VLSI Chip Design Productivity," *IEEE J. Solid State Circuits*, Vol. SC-20, No. 2, Apr. 1985, pp. 555-561.
19. C. Fey and D. Paraskevopoulos, "A Techno-Economic Assessment of Application-Specific Integrated Circuits Current and Future Trends," *Proc. IEEE*, Vol. 75, No. 6, June 1987, pp. 829-841.
20. J. Di Giacomo, ed., *VLSI Handbook*, McGraw-Hill, 1988.
21. I. Dear and A. Ambler, "Predicting Cost and Quality Improvements as a Result of Test Strategy Planning," *Proc. IFIP Workshop on Fast Prototyping*, IFIP, Geneva, 1987.
22. P. Goel, "Test Generation Costs Analysis and Projections," *Proc. 17th Design Automation Conf.*, 1980, pp. 77-84.
23. R.E. Huston, "An Analysis of ATE Testing Costs," *Proc. IEEE Int'l Test Conf.*, 1983, pp. 396-411.
24. K. Cheng and V. Agrawal, "A Partial Scan Method for Sequential Circuits with Feedback," *IEEE Trans. Computers*, Vol. 39, No. 4, Apr. 1990, pp. 544-548.
25. I. Dear et al., "Hierarchical Testability Measurement and Design for Test Selection by Cost Prediction," *Proc. IEEE European Test Conf.*, 1989, pp. 50-57.
26. C. Dislis et al., "Cost Effective Test Strategy Selection," *Proc. IFIP Workshop on Knowledge Based Systems for Test and Diagnosis*, IFIP, Geneva, 1988.
27. M. Abadir, "TIGER: Testability Insertion Guidance Expert System," *Proc. IEEE Int'l Conf. Computer Aided Design*, IEEE CS Press, 1989, pp. 562-565.
28. R. Treuer, H. Fujiwara, and V. Agrawal, "Implementing a Built-In-Self-Test PLA Design," *IEEE Design & Test of Computers*, Vol. 2, No. 2, Apr. 1985, pp. 37-48.
29. R. Illman, "Design of a Self-Testing RAM," *Proc. Silicon Design Conf.*, Electronic Design Automation Pulications Ltd., London, 1986, pp. 439-446.
30. J. Miles, R. De Bondt, and L. Daeman, "A Test Economics Model and Cost Benefits Analysis of Boundary Scan," *Proc. Second European Test Conf.*, Vde-Verlag, Berlin, 1991, pp. 375-384.
31. M. Levitt and J. Abraham, "The Economics of Scan Design," *Proc. IEEE Int'l Test Conf.*, 1989, pp. 869-874.
32. D. Reinertsen, "Whodunit? The Search for the New-Product Killers," *Electronic Business*, Vol. 11, July 1983, pp. 106-109.
33. I. Dear, A. Ambler, and C. Maunder, "The Application of Analytical Cost Models for Optimisation of Field Service Strategies," *Proc. ACM Int'l Workshop Economics of Design and Test*, ACM, New York, 1991.
34. J. Dick, E. Trischler, and A. Ambler, "DOM: A Defect Occurrence Model for Evaluating the Life-Cycle Costs of Test Strategies," *Proc. ACM Int'l Workshop Economics of Design and Test*, ACM, 1991.

I.D. Dear is a lecturer in digital systems in the Department of Electrical and Electronic Engineering at Brunel University. His current research interests include design and test economics, test strategy planning, and CAD for VLSI devices. Dear holds a PhD from Brunel University in test strategy planning and economic modeling, a BSc in electronic engineering from the University of Sussex, and an MSc in digital systems from Brunel.

Chryssa Dislis is a research assistant in the same department at Brunel University. She currently works on an ESPRIT-funded project on cost-based test strategy planning for VLSI devices. Her interests include design for testability, CAD for VLSI devices, and the economics of test at chip and board levels. Dislis holds a BSc in electronic engineering from the University of Sussex. She is a member of the IEEE and an associate member of the Institute of Electrical Engineers in the UK.

A.P. Ambler holds the Racal Redac Chair in test technology at Brunel University. He also serves as European representative editor to *IEEE Design & Test of Computers* and recently joined the editorial board of JETTA. He has helped organize several conferences including ICCD, EDAC, the International Workshop on Expert Systems in Test and Diagnosis, and the First International Workshop on the Economics of Design and Test. His research interests include design for testability, the economics of test, CAD for VLSI devices, fault simulation, and hardware accelerators. Ambler holds a PhD in test and simulation of multiple-valued logic. He is a member of the IEEE Computer Society, the ACM, and the IEE. He is a chartered engineer.

Jochen Dick is a staff member of Siemens-Nixdorf's CAE software development laboratories. His work includes research in automatic test pattern generation and the development of a DFT rule checker. His recent research activities have focused on test economics and test strategy planning. He holds a MS in electrical engineering from the Technical University of Munich.

Direct questions about this article to A.P. Ambler, Brunel University, Department of Electrical Engineering & Electronics, Uxbridge, Middlesex, UB8 3PH, United Kingdom.

for Custom VLSI," *Proc. IEEE Int'l Test Conf.*, 1986, pp. 232-243.

16. J. Miles, A. Ambler, and K. Totton, "Area and Performance Estimation of Design for Test Methods," *Proc. IEEE Int'l Conf. Computer Design*, 1988.
17. J. Miles, *Cost Modelling for VLSI Circuit Conversion to Aid Testability*, doctoral dissertation, Brunel Univ., Uxbridge, UK, 1988.
18. C. Fey, "Custom LSI/VLSI Chip Design Productivity," *IEEE J. Solid State Circuits*, Vol. SC-20, No. 2, Apr. 1985, pp. 555-561.
19. C. Fey and D. Paraskevopoulos, "A Techno-Economic Assessment of Application-Specific Integrated Circuits Current and Future Trends," *Proc. IEEE*, Vol. 75, No. 6, June 1987, pp. 829-841.
20. J. Di Giacomo, ed., *VLSI Handbook*, McGraw-Hill, 1988.
21. I. Dear and A. Ambler, "Predicting Cost and Quality Improvements as a Result of Test Strategy Planning," *Proc. IFIP Workshop on Fast Prototyping*, IFIP, Geneva, 1987.
22. P. Goel, "Test Generation Costs Analysis and Projections," *Proc. 17th Design Automation Conf.*, 1980, pp. 77-84.
23. R.E. Huston, "An Analysis of ATE Testing Costs," *Proc. IEEE Int'l Test Conf.*, 1983, pp. 396-411.
24. K. Cheng and V. Agrawal, "A Partial Scan Method for Sequential Circuits with Feedback," *IEEE Trans. Computers*, Vol. 39, No. 4, Apr. 1990, pp. 544-548.
25. I. Dear et al., "Hierarchical Testability Measurement and Design for Test Selection by Cost Prediction," *Proc. IEEE European Test Conf.*, 1989, pp. 50-57.
26. C. Dislis et al., "Cost Effective Test Strategy Selection," *Proc. IFIP Workshop on Knowledge Based Systems for Test and Diagnosis*, IFIP, Geneva, 1988.
27. M. Abadir, "TIGER: Testability Insertion Guidance Expert System," *Proc. IEEE Int'l Conf. Computer Aided Design*, IEEE CS Press, 1989, pp. 562-565.
28. R. Treuer, H. Fujiwara, and V. Agrawal, "Implementing a Built-In-Self-Test PLA Design," *IEEE Design & Test of Computers*, Vol. 2, No. 2, Apr. 1985, pp. 37-48.
29. R. Illman, "Design of a Self-Testing RAM," *Proc. Silicon Design Conf.*, Electronic Design Automation Pulications Ltd., London, 1986, pp. 439-446.
30. J. Miles, R. De Bondt, and L. Daeman, "A Test Economics Model and Cost Benefits Analysis of Boundary Scan," *Proc. Second European Test Conf.*, Vde-Verlag, Berlin, 1991, pp. 375-384.
31. M. Levitt and J. Abraham, "The Economics of Scan Design," *Proc. IEEE Int'l Test Conf.*, 1989, pp. 869-874.
32. D. Reinertsen, "Whodunit? The Search for the New-Product Killers," *Electronic Business*, Vol. 11, July 1983, pp. 106-109.
33. I. Dear, A. Ambler, and C. Maunder, "The Application of Analytical Cost Models for Optimisation of Field Service Strategies," *Proc. ACM Int'l Workshop Economics of Design and Test*, ACM, New York, 1991.
34. J. Dick, E. Trischler, and A. Ambler, "DOM: A Defect Occurrence Model for Evaluating the Life-Cycle Costs of Test Strategies," *Proc. ACM Int'l Workshop Economics of Design and Test*, ACM, 1991.

I.D. Dear is a lecturer in digital systems in the Department of Electrical and Electronic Engineering at Brunel University. His current research interests include design and test economics, test strategy planning, and CAD for VLSI devices. Dear holds a PhD from Brunel University in test strategy planning and economic modeling, a BSc in electronic engineering from the University of Sussex, and an MSc in digital systems from Brunel.

Chryssa Dislis is a research assistant in the same department at Brunel University. She currently works on an ESPRIT-funded project on cost-based test strategy planning for VLSI devices. Her interests include design for testability, CAD for VLSI devices, and the economics of test at chip and board levels. Dislis holds a BSc in electronic engineering from the University of Sussex. She is a member of the IEEE and an associate member of the Institute of Electrical Engineers in the UK.

A.P. Ambler holds the Racal Redac Chair in test technology at Brunel University. He also serves as European representative editor to *IEEE Design & Test of Computers* and recently joined the editorial board of JETTA. He has helped organize several conferences including ICCD, EDAC, the International Workshop on Expert Systems in Test and Diagnosis, and the First International Workshop on the Economics of Design and Test. His research interests include design for testability, the economics of test, CAD for VLSI devices, fault simulation, and hardware accelerators. Ambler holds a PhD in test and simulation of multiple-valued logic. He is a member of the IEEE Computer Society, the ACM, and the IEE. He is a chartered engineer.

Jochen Dick is a staff member of Siemens-Nixdorf's CAE software development laboratories. His work includes research in automatic test pattern generation and the development of a DFT rule checker. His recent research activities have focused on test economics and test strategy planning. He holds a MS in electrical engineering from the Technical University of Munich.

Direct questions about this article to A.P. Ambler, Brunel University, Department of Electrical Engineering & Electronics, Uxbridge, Middlesex, UB8 3PH, United Kingdom.