University of Southampton

LOW-COST TEST FOR CORE-BASED SYSTEM-ON-A-CHIP

by

Paul Theo Gonciari

A thesis submitted in partial
fulfilment of the requirements for the degree of
Doctor of Philosophy
in the
Faculty of Engineering and Applied Science
Department of Electronics and Computer Science

December 2002

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

LOW-COST TEST FOR CORE-BASED SYSTEM-ON-A-CHIP

by Paul Theo Gonciari

The availability of high level integration leads to building of millions of gates systems-on-a-chip (SOC). Due to the high complexity of SOCs, testing them is becoming increasingly difficult. In addition, if the current test practises are maintained, the high cost of test will lead to a considerable production cost increase. To alleviate the test cost problem, this research investigates methods which lead to *low-cost test of core-based systems-on-a-chip* based on test resource partitioning and without changing the embedded cores. Analysing the factors which drive the continuous increase in test cost, this thesis identifies a number of factors which need to be addressed in order to reduce the cost of test. These include volume of test data, number of pins for test, bandwidth requirements and the cost of test equipment. The approaches proposed to alleviate the cost of test problem have been validated using academic and industrial benchmark cores.

To reduce the volume of test data and the number of pins for test, the new Variable-length Input Huffman Coding (VIHC) test data compression method is proposed, which is capable of simultaneously reducing the volume of test data, the test application time and the on-chip area overhead, when compared to previously reported approaches. Due to the partitioning of resources among the SOC and the test equipment, various synchronisation issues arise. Synchronisation increases the cost of test equipment and hence limits the effectiveness of test resource partitioning schemes. Therefore, the synchronisation issues imposed by test data compression methods are analysed and an on-chip distribution architecture is proposed which in addition to accounting for the synchronisation issues also reduces the test application time.

The cost of test equipment is related to the amount of test memory, and therefore efficient exploitation of this resource is of great importance. Analysing the memory requirements for core based SOCs, useless test data is identified as one contributor to the total amount of allocated memory, leading to inefficient memory usage. To address this problem a complementary approach to test data compression is proposed to reduce the test memory requirements through elimination of useless test data.

Finally, a new test methodology is proposed which combines the approaches proposed in this thesis into an integrated solution for SOC test. The proposed solution leads to reduction in volume of test data, test pins, bandwidth requirements and cost of test equipment. Furthermore, the solution provides seamless integration with the design flow and refrains from changing the cores. Hence, it provides a low-cost test solution for core-based SOC using test resource partitioning.

Acknowledgements

I wish to thank my supervisor Dr. Bashir Al-Hashimi for his help and assistance, for his guidance and encouragement throughout this PhD. I am grateful for his invaluable help during the preparation of this thesis and several papers.

This PhD would have not been possible without the generous funding and computing facilities provided by the Department of Electronics and Computer Science, University of Southampton, for which I am grateful. Thanks are also due to the members of the Electronic Systems Design Group which have inspired me with their support. In particular I wish to acknowledge Dr. Neil J. Ross for serving on my nine-month and MPhil transfer, and for his constructive criticism.

I wish to thank my friend and colleague Dr. Nicola Nicolici for his guidance, constructive comments and supervision throughout the period of this PhD. I am grateful for the numerous late night technical discussions, especially before paper submission deadlines. I would also like to thank the Department of Electrical and Computer Engineering for providing the required computing facilities during the 3 months period whilst I was a visiting researcher in McMaster University.

I am thankful to my friends Marcus Schmitz, Mauricio Varea, Neil Henderson, Mircea Gândila, Reuben Wilcock and Manoj Gaur, to name a few, for their support and for providing feedback on several occasions. Special thanks are due to Paul Rosinger for intriguing discussions and valuable feedback throughout this PhD. I am also thankful to my landlord for putting up with me for the period of this PhD, and for introducing me to simplicity. Thanks are also due to Monika and Daniel Benta for their support whilst I was staying in Germany. Special thanks go to Ioana Oarga, for her love and for reminding me of the important things in life.

Last but certainly not least, this work could have not been carried out without the unshakable love, encouragement and tolerance of my mother Renate, father Matei, brother Demis, aunt Gerlinde and my grandmother Anna, to whom I am forever thankful.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ATE	Automatic Test Equipment
ATPG	Automatic Test Pattern Generation
BIST	Built-In Self-Test
CI	Code Identifier
CMOS	Complementary Metal Oxide Silicon
CTL	Core Test Language
CUT	Core Under Test
CWD	Core Wrapper Design
DFT	Design for Test
DIB	Device Interface Board
DRAM	Dynamic Random Access Memory
DUT	Device Under Test
EDA	Electronic Design Automation
FDR	Frequency Directed Run-length
FF	Flip Flop
HDL	Hardware Description Language
IC	Integrated Circuit
IP	Intellectual property
ITRS	International Roadmap for Semiconductors
MSC	Multiple Scan Chains
PCB	Printed Circuit Board
PG	Pattern Generator

RAM	Random Access Memory
RISC	Reduced Instruction Set Computer
RPCT	Reduce Pin Count Test
SC	Selective Coding
SECT	Standard for Embedded Core Test
SFF	Scan Flip Flop
SOC	System-on-a-Chip
SSC	Single Scan Chain
TAM	Test Access Mechanism
TAT	Test Application Time
TDC	Test Data Compression
TDCE	Test Data Compression Environment
UDL	User Defined Logic
UMA	Useless Memory Allocation
VIHC	Variable-length Input Huffman Coding
VTD	Volume of Test Data
WSC	Wrapper Scan Chain

Chapter 1

Introduction

Driven by the high level of integration the industry is capable of building an entire system into a single chip [1, 2, 3, 4]. Systems-on-a-chip (SOC) are composed from *embedded cores*, which make it easier to import existing technologies and shorten the time-to-market through design reuse [1]. The new market driven by the SOC revolution comprises *core vendors* as providers, and *core users* as customers. While the core vendor has full knowledge about the core's functionality and its internal structure, the core user is often knowledgeable only about the core's functionality. Therefore, besides the immediate advantages of core based SOCs, the SOC paradigm brings forth new problems from the design and test perspective [5, 1, 6, 2], which require new test strategies for testing the core itself as well as the entire system [7, 8, 9, 10, 11, 12].

Due to the continuous increase in chip complexity and transistor density, the cost of testing the chips will approach and even exceed the cost of manufacturing them [13]. Therefore, if the cost of test is not lowered, testing will have a negative impact on the cost of the design [14, 9, 15, 13] leading to an increase in the overall production cost of the chip. Addressing the cost of test problem in a core based SOC environment requires solutions which refrain from changing the cores. This is of great importance since if core redesign is required, then the very purpose of core based SOC, short time-to-market through core reuse, can be defeated. Considering this constraint imposes the usage of basic knowledge about the cores when addressing the cost of test problem. Therefore, the topic of this dissertation is to provide methods which contribute toward reduced cost of test providing *low-cost system-on-a-chip test* solutions assuming only basic knowledge about the cores, and hence leveraging the core based SOC environment.

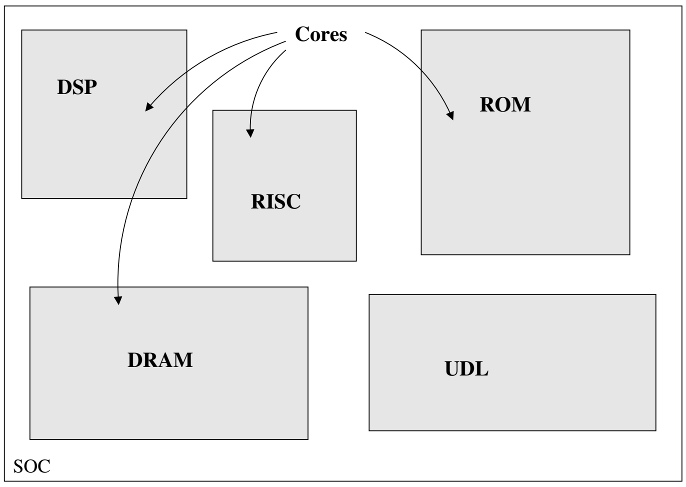


Figure 1.1. System-on-a-Chip, an example

This chapter is organised as follows. In Section 1.1 the differences between the traditional printed circuit board (PCB) and the new SOC design flows are illustrated, emphasising the implications on testing SOCs. Section 1.2, illustrates the SOC DFT challenges, and Section 1.3 investigates the main factors which drive the continuous increase in test cost, motivating the work of this thesis. Finally, Section 1.4 outlines the main contributions of this work and presents the organisation of the thesis.

1.1 Core based System-on-a-Chip design

To understand why testing in the new core based SOC paradigm becomes such a major problem, and why it is important to address it without imposing changes to the cores, one must understand the differences between the SOC and the traditional PCB design flows and how these differences influence the testing of the entire system. Therefore, in this section a generic integrated circuit (IC) design flow is described, and the various types of cores and their influences on the design flow are illustrated.

There are two main types of components is SOCs: the cores and the user defined logic (UDL). This is exemplified in Figure 1.1 where the cores (which in the figure are represented by memories (DRAM, ROM) and the RISC and DSP processors) and UDL are illustrated considering a fictive system. A *core* is a pre-designed, pre-verified silicon circuit block, usually containing at least 5,000 gates, that can be used in building a larger or more complex application on a semiconductor chip [1]. Similar to ICs on a

PCB, cores can perform a wide range of functions (e.g., digital signal processors, RISC processors, or DRAMs) and can be found in a number of technologies (e.g., complementary metal-oxide-silicon (CMOS) logic, DRAM and analog circuits). Furthermore, the more complex cores come in hierarchical compositions (i.e., complex cores comprise a number of simple cores). Often these cores are products of technology, software, and know-how that are subject to patents and copyrights. Hence, a core block represents intellectual property (IP) that the core builder licenses to the core user [1]. Therefore, the core user is not always entitled to do changes to the core and is forced to reuse it "as is" (as a black box), being knowledgeable only about the core's functionality, however, not about the implementation details. In addition, while ICs are delivered to the customer in a *manufactured and tested* form, cores are delivered in a range of hardware description levels (soft, firm, and hard). These two fundamental differences will influence not only the design of the SOCs (as shown in Section 1.1.2) but also the testing of the SOCs (as shown in Section 1.2).

UDL is added by the system integrator to the SOC to customise or to interface the different cores such that the desired functionality is obtained. While the cores are IPs delivered by third party vendors and are reused within the design, the UDL is custom logic specific to the SOC under development. Hence, the system integrator has full knowledge about its functionality and its implementation details. The UDL is a required investment from the design point of view, however, testing the UDL gives way to new issues in system test [16, 17, 18].

1.1.1 Design flow

In this section a generic IC design flow is illustrated. Starting from the initial requirements definitions, i.e., the specifications of the desired chip functionality described in a formal form, the design follows a number of phases which transform these specifications into the final chip in a chosen technology. This is illustrated with the left hand side of the generic core based SOC design flow in Figure 1.2. The term technology (or fabrication technology) refers to the semiconductor process used to produce the final chip, i.e., it denotes the type of semiconductor (e.g., silicon), the type of transistor (e.g., CMOS) or the details of a certain transistor technology (e.g., 0.18 micron CMOS) [19]. Each design phase corresponds to a level of abstraction, i.e., the level of dependence on the fabrication

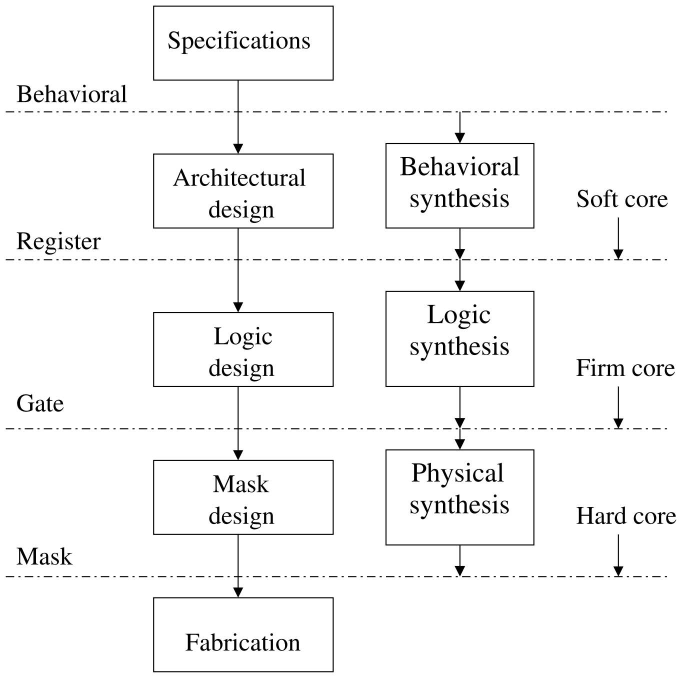


Figure 1.2. Core based SOC design

technology. There are four levels of abstraction: *behavioural level*, where the specifications are described using generic methods, no structural constraints are considered (e.g., a variable is of type integer; a conditional assignment is represented by an IF statement); *register transfer level* (RTL), where generic functions and variables are replaced by structural blocks (e.g., the variable is replaced by a register of a given size; an IF statement is represented by a multiplexer, the arithmetic operations are replaced with arithmetic logic units (ALUs)); *gate level*, where the structural components are mapped to netlists – interconnection of building cells, gates – in a specific cell library (e.g., a 2 : 1 multiplexer is represented by two AND gates and an OR gate); and *mask level* (or *physical level*), where each cell is represented by the layout of the transistor model counterpart. The specifications of the desired chip functionality are described at the various levels of abstraction using hardware description languages (HDL). For example, at the behavioural level of abstraction, the HDL can be VHDL [20, 21, 22, 23], Verilog [24], C and C++ [25, 26], and SystemC [27, 28, 29]; at the RTL level of abstraction, the HDL is usually VHDL and Verilog. From the initial specifications to the final chip, the design goes through a number of translation and verification steps. The translation of the descriptions from one level of abstraction into the other is referred to as *synthesis*. Verification is used to check whether the design in the current level of abstraction is correct, conforming with the specifications. Both steps are performed using computer aided design (CAD) tools [30].

Synthesis A synthesis process aims at finding the best equivalent representation in the next level of abstraction which guarantees the same functionality as the design in the current level of abstraction considering the constraints given by the designer (e.g., the area occupied by silicon should not exceed 300 sq-mils; the design should work for frequencies up to 100 MHz). Corresponding to the above four levels of abstractions, there are three synthesis steps: behavioural synthesis, logic synthesis and physical synthesis. These transform the initial specifications from behavioural to RTL, then from RTL to gate level, and finally from gate level to layout.

Verification Verification in hardware design mainly consists of validation and formal verification. Validation, verifies the design through simulation and testing. Formal verification, on the other hand, uses rigorous mathematical reasoning to show that a design meets all or parts of its specification [31, 32]. However, it does not necessarily guarantee correct operation with respect to timing, power, noise and routability [33]. At the different levels of abstraction, validation and formal verification take different shapes. For example, they are represented by property checking [31, 34] or RTL/cycle simulation [32] at the RTL level; and equivalence checking and physical verification at the gate and layout (or mask) level [35]. The verified design at mask level will then go through manufacturing.

Manufacturing The manufacturing of an IC implies wafer fabrication and packaging. Wafer fabrication involves printing of geometric shapes corresponding to the layout onto wafer layers [19]. As each wafer contains a number of chips, the wafer is cut and then packaged. Each of the above processes involved in wafer fabrication can introduce defects into the chip. For example, particles (material which is not removed in the areas exposed by the masking process) may cause bridges between two or more lines; incorrect spacing between connections may cause circuit shorts; holes (exposed areas that is unexpectedly etched) may lead to open interconnects. These manufacturing defects (or physical faults) affect the functionality of the circuit and hence lead to faulty chips. To avoid shipping bad chips to customers, and at the same time correct the manufacturing process which caused the defects, hence improving yield, testing of the manufactured circuit is required.

Testing After wafer manufacturing, testing is also referred to as manufacturing test. The wafer is first subjected to wafer sort, where most of the faulty chips are identified, and then the packaged chips are further tested in order to eliminate the defects which escaped wafer sort (due to electrical limitations), and in order to accelerate infant mortality through burn-in (which is a high-voltage, high-temperature stress of the device) [36, 37, 38]. Finally, the device is tested for conformance with the specifications. Because of the high complexity of the chips, testing is done using automated test equipment (ATE). The ATE is a precision measurement tool which provides the environment required to test the chip. There are two main types of testing: functional test and structural test. The former checks for the correct functionality of the manufactured chip. The latter, verifies that the chip was manufactured correctly [38]. If not specified differently, the term testing and manufacturing test will refer to structural testing as introduced above. Testing implies applying test stimuli (or test vectors), and inspecting the circuit responses (or test responses). If the circuit response is not correct then a defect was identified. While during verification, at the previous levels of abstraction, access to the entire model was available, during manufacturing test, the access to the chip's internals is limited by the inputs and outputs of the circuit. To increase the access to the chip internals and increase the effectiveness of testing, design for testability (DFT) has been introduced as a mandatory step in the design process [36, 38]. While DFT features are embedded in various design stages of the IC (e.g., at the RTL or the gate level of abstraction [39, 40, 41]), they are exploited after the manufacturing of the wafer and the packaging of the chip. In order to ensure the effectiveness of DFT, the manufacturing test problem has to be modelled at the higher levels of abstraction. Therefore, faults models were developed corresponding to the physical defects [36, 37, 38]. Based on the fault models, the problem of identifying a defect is equivalent to finding the circuit's input values (test stimuli) which will provoke a faulty and fault free output (test responses) depending on the existence of the fault. This process is performed by automatic test pattern generation (ATPG) [42, 43, 44] tools. ATPG tools can determine a list of faults, the test vectors which detect the faults, and the corresponding fault free responses. In addition, using fault simulation ATPG tools can estimate the fault coverage (how many from the total modelled faults can be detected), and they can also be used for fault diagnosis (which was the root cause of the defect). The test vectors generated by ATPG can be completely specified (e.g., "1100") or incompletely specified (e.g., "1x0x") in which case they are referred to as test cubes. For the different types of defects various fault models have been proposed (e.g., bridging fault models, open fault models, delay fault models, stuck-at fault models [36, 45, 37, 38, 46]). For example, the stuck-at fault model assumes that a given node has always a fixed value, either logic 0 or logic 1, i.e., the node is assumed to be either connected to the power line (stuck-at-1 fault) or connected to the ground line (stuck-at-0 fault).

1.1.2 Core based design

The hardware description levels in which core providers deliver their cores can be related to the levels of abstractions described in the previous subsection. Hence,

- *soft* cores consists of a synthesisable representation that can be re-targeted for different technologies and system constraints. As a HDL description (behavioural or RTL), these cores leave much of the implementation to the designer, however they are flexible and process-independent;
- *firm* cores usually consist of gate-level netlist representation ready for place and route. They give some flexibility to the designer, are accompanied by simulation models and fit into standard HDL design methodologies;
- *hard* cores include timing information and technology dependent layout, i.e., physical level description of the core. While they are optimised for predictable area and performance, they lack flexibility.

The three types of cores offer various trade-off opportunities, each of them having different modelling and test requirements [1], affecting the design at the corresponding level of abstraction. This is illustrated on the right hand side of Figure 1.2 (see Page 4), where the entrance of the different types of cores in the design flow is shown. Core based integration within the design flow is possible if electronic design automation (EDA) companies provide tools which are capable of embedding cores at their corresponding level of abstraction, considering not only their functionalities but also their DFT features [1]. As mentioned in the beginning of Section 1.1, there are two factors which fundamentally differentiate PCB design and core based SOC design: the IP factor and the manufacturing factor. The former will forbid the core user to do any changes to the core. This will impact the design cycle of the SOC, since the changes have to be done by the IP/core vendor [1], and the testing of the design, as the system integrator is forced to use the DFT embedded with the IP even if this does not suit the DFT methodologies adopted for the rest of the SOC. The latter will mainly influence the testing of the SOC as illustrated in the following section.

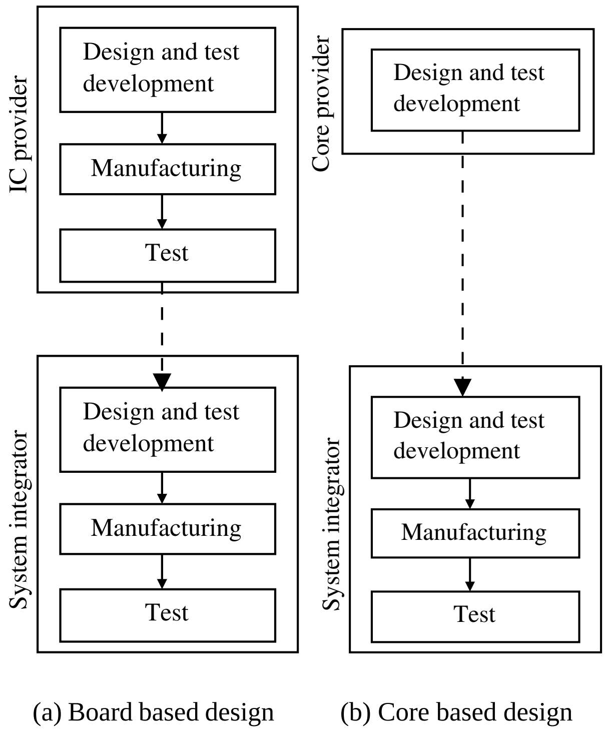


Figure 1.3. Board vs. SOC design [2]

1.2 Core based System-on-a-Chip test

Due to the high complexity of SOCs two main problems are identified with respect to test. Firstly, the test infrastructure for the SOCs, and secondly, the cost of test, which if current trends continue will have a negative impact on the total production cost of the chip. The former is illustrated in this section, while the latter is detailed in Section 1.3.

To illustrate the challenges in testing SOCs, firstly the difference between the PCB and SOC design flows is analysed. The two design flows are shown from the system integrator's perspective, illustrating only a high level view of the two design processes, in Figure 1.3. In the traditional PCB design (see Figure 1.3(a)) each IC has been designed and developed, manufactured and tested for manufacturing faults. The system integrator had merely to put the system together and test for interconnect faults between the ICs on the PCB. In the core base SOC design (see Figure 1.3(b)) the cores are only designed and developed, the manufacturing and testing of the cores is the job of the system integrator. The system integrator in this case has to manufacture the SOC (and not assemble the system as in PCB like systems), test each core for manufacturing faults, test the interconnects and the entire system. Hence, even though conceptually the difference between the two designs is small, the issues in testing core based designs differ considerably from testing PCBs. The problems related to core based SOC test [2] are detailed next.

Core level test deals with the testing of the core itself. Since the system integrator is usually forced to deal with the core as a black box, the core vendor has to provide the model, the DFT (e.g., scan based DFT or built-in self test (BIST) based DFT[1]) [36, 47, 38] structures and the corresponding test vectors.

Test access provides the electronic test infrastructure for accessing the cores. This is required since contrary to PCBs where the ICs primary inputs and outputs were accessible, and thus during test direct physical access to the IC is available, in SOCs the cores are embedded within the chip and thus, no access to the cores terminals is available. In addition to providing the means to access the cores, the test access needs to isolate the core under test (CUT) from its surroundings, facilitating inter-core test [48]. Furthermore, test scheduling is also required to run intra-core and inter-core tests such that initialisation and final content of the individual cores is not affected.

System level test addresses the test of the entire system. Compared to conventional PCB test, system level test is far more complex. The system chip test comprises: tests for the individual core, tests for the UDL and tests for interconnect logic and wiring. This composite system level test requires adequate test scheduling which must meet several chip-level constraints, such as, test application time (TAT), area overhead, and power dissipation [49].

Driven by the above challenges two conceptual architectures for SOC test have been proposed [2], illustrated in Figures 1.4 and 1.5. Characteristic to both architectures are the *source and sink*, the *test access mechanism* and the *core test wrapper*, described next:

- the *source and sink* are responsible for providing the test stimuli and analysing the test responses. Depending on the location of the source and the sink, an external-only (Figure 1.4) or internal-only (Figure 1.5) solution has been envisioned. The former is referred to as external deterministic test where the source and the sink are represented by the automatic test equipment (ATE), detailed in Section 1.2.2. For the

[1]For scan based and BIST based DFT the reader is referred to Appendix A

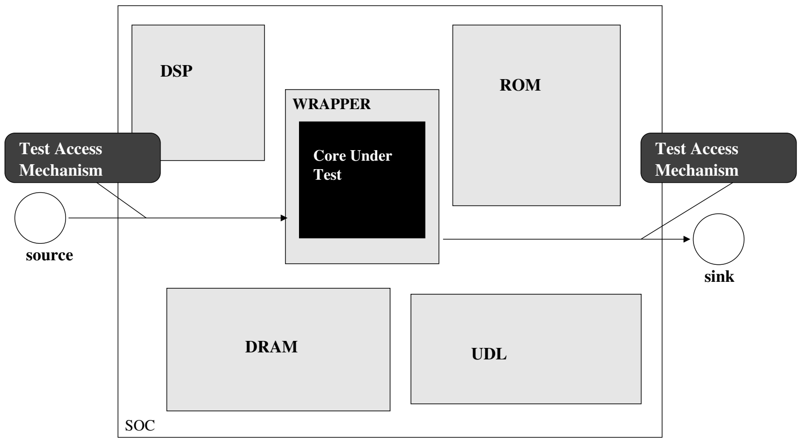


Figure 1.4. Conceptual architecture for external SOC test [2]

latter the internal source and sink are represented by BIST structures. The test in this case is initiated by an off-chip test equipment, which also receives the result of the test;

- *test access mechanism (TAM)* provides the mechanism to transport the test stimuli from the source to the CUT, and the test responses from the CUT to the sink. In general, having a wide TAM will reduce the test application time, however, it will also increase the wiring overhead. Hence, designing a TAM involves making trade-offs between the bandwidth and the test application cost. In addition, the type of test architecture (internal-only or external-only) must also be consider, since, when the external test is performed, the TAM width should not exceed the number of ATE I/O channels (see Section 1.2.2);

- *core test wrapper* provides an interface between the embedded core and its environment. The core test wrapper connects the core terminals to the rest of the chip and to the TAM. A core test wrapper, must support the following mandatory modes [49, 50, 51]: *Normal operation*: when the core test wrapper is transparent and the core is connected to the chip environment; *Core-internal test*: when the core test wrapper connects the TAM to the core such that the test patterns can be applied, and the test responses observed; *Core-external test*: when testing of the interconnect logic and wiring is allowed by the core test wrapper.

With respect to DFT, the above can be divided into *core level DFT*, which includes scan based and BIST based DFT; and *system level DFT*, which provides the SOC test

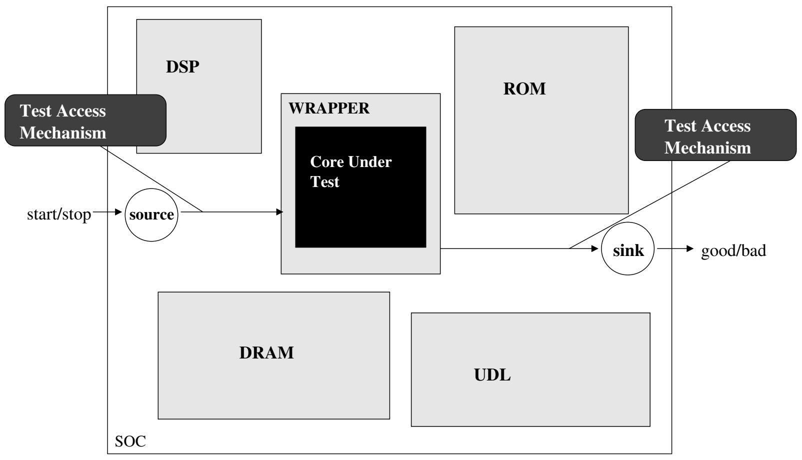


Figure 1.5. Conceptual architecture for internal SOC test [2]

infrastructures, i.e., the core test wrapper and the TAM. System level DFT comprising the core test wrapper design and TAM design are illustrated next.

1.2.1 System level DFT

It should be noted that the core wrapper design can be performed by both the core vendor and the system integrator. If not specified differently, it is considered in this dissertation that core wrapper design is the responsibility of the system integrator. This is motivated by the flexibility obtained in this case as detailed in the next section.

Core test wrapper design

It has been illustrated in the previous section that the core test wrapper provides an interface between the embedded core and its environment. Because the core user rarely participates in a core's architectural and functional development, the core appears as a black box with known functionality and I/Os [11]. To integrate the core into a design and to exploit the DFT features of the core, the core test access interface and the core test knowledge must be available to the core user. To provide a common basis for core test knowledge exchange and core test access design, the emerging IEEE P1500 [52] working group is focusing on standardising: *Core Test Language* (CTL), capable of representing all test related information [53, 50, 51, 54], and a scalable and configurable *core test wrapper* to allow easy test access to the core in a SOC [49, 53, 51].

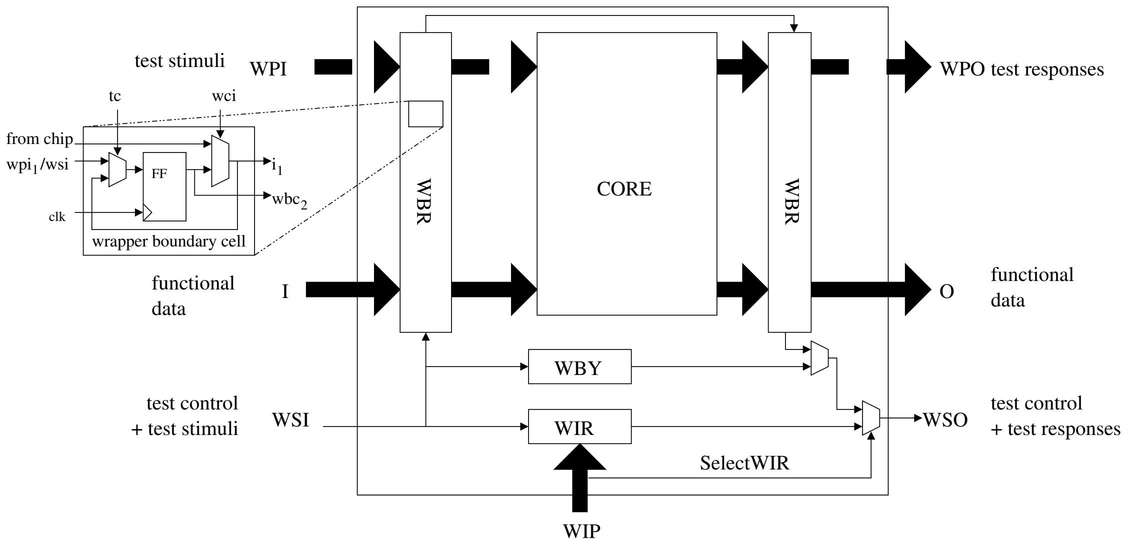


Figure 1.6. IEEE P1500 wrapped core [55]

The IEEE P1500 standard for embedded core test (SECT) does not cover the core's internal test methods or DFT, nor SOC test integration and optimisation. These are completely in the hands of the core provider or the core user respectively, and not suited for standardisation due to the fact that their requirements differ for different cores and SOCs [49, 53, 51]. Hence, in the emerging P1500 standard [52], it is assumed that the core provider performs DFT (scan design, BIST) [36, 47, 38] and supplies all the information required for the proper test of the core [49, 53, 51]. Currently the IEEE P1500 SECT is focusing on "non-merged"[2] digital logic and memory elements. In order to provide the flexibility required to test various types of cores, the IEEE P1500 SECT allows "two levels of compliance". (i) *IEEE 1500 Unwrapped Core*, this level will allow the core to be delivered without a complete IEEE 1500 wrapper however, it does have the IEEE 1500 CTL program. Using this program, the core user can develop its own IEEE 1500 wrapper. In this scenario, the CTL will describe the core test knowledge at the bare core terminals. (ii) *IEEE 1500 Wrapped Core*, this level requires the complete IEEE 1500 wrapper and the IEEE 1500 CTL to be delivered with the core. In this scenario, the CTL describes the core test knowledge including how to operate the core wrapper.

The P1500 core wrapper is a shell which allows various configurations at its inputs and outputs, i.e., various operating modes. A IEEE P1500 core wrapper is illustrated in Figure 1.6. The wrapper has as inputs the wrapper parallel input (WPI), the wrapper

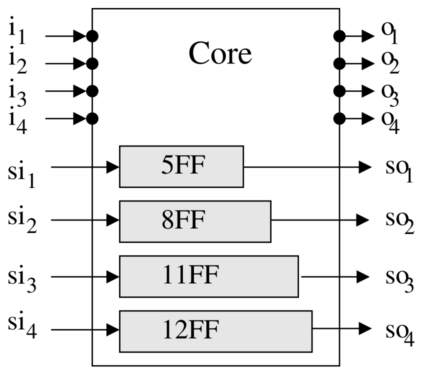
[2]"Non-merged" refers to cores that are tested as separate entities with test patterns developed for each core as stand alone [51]

serial input (WSI) and the functional inputs (I), and the wrapper interface port (WIP). The outputs are the wrapper parallel output (WPO), the wrapper serial output (WSO) and the functional outputs (O). The core wrapper comprises two wrapper boundary registers (WBR), the wrapper bypass register (WBY) and the wrapper instruction register (WIR).

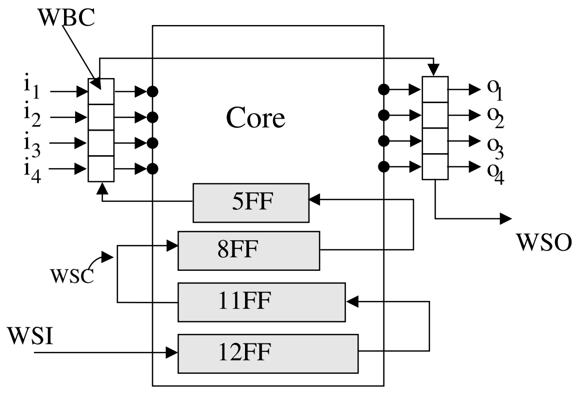
The WBRs provide the interface between the test environment and the core, facilitating the test modes noted in Section 1.2: normal operation, core-internal test and core-external test. The WBR is a register formed out of wrapper boundary scan cells. The wrapper cell assigned to the inputs are referred to as wrapper boundary input cells, while those assigned to the outputs are referred to as wrapper boundary output cells. The cell illustrated in the top-left part of Figure 1.6 is a wrapper boundary input cell. The functionality of the cell is similar to the scan cell (see Appendix A). One difference is the output multiplexer controlled by the wrapper control input (wci) which differentiate between the test mode and the normal functional mode.

The WIP facilitates the load of instructions corresponding to the above modes into the WIR. For each of the above modes there can be different configurations. For example, there can be a serial internal test, when the test stimuli are provided through the WSI (serial access), and the test responses are observed at the WSO; or a parallel internal test when the test stimuli are provided through the WPI (parallel access), and the test responses are observed at the WPO. It should be noted that this dissertation targets core internal tests, for both the serial and the parallel configurations. However, the methodologies presented are also applicable to core external tests. While the behaviour of the core wrapper and its interface are standardised, the design of the core wrapper itself can be adapted for the system integrator's necessities. The usage of the serial and parallel test access are illustrated next for a fictive core.

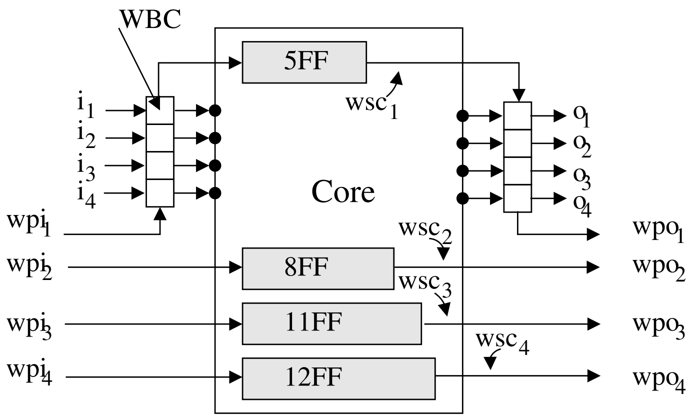
Core wrapper design implies the construction of wrapper scan chains (WSCs). A WSC comprises a number of wrapper boundary input cells, scan chains and wrapper boundary output cells. Corresponding to the two types of access (serial or parallel) the WSC construction is either a simple concatenation of scan chains, or an optimisation problem with respect to minimum test application time. This is illustrated in Figure 1.7 where for the core in Figure 1.7(a), a core wrapper design for a serial test access is given in Figure 1.7(b), and a possible WSC configuration for a parallel test access is given Figure 1.7(c). The core illustrated in Figure 1.7(a) has four inputs (i_1, i_2, i_3 and i_4), four outputs (o_1, o_2, o_3 and o_4) and four scan chains (s_1, s_2, s_3 and s_4) of length $5, 8, 11$ and 12



(a) Core



(b) Core wrapper with serial access



(c) Core wrapper with parallel access

Figure 1.7. Core and core wrapper design

FFs respectively. In Figure 1.7(b), the internal scan chains and the WBCs are connected together into a long scan chain, while in Figure 1.7(c) a four bit wide WPI port is assumed, and the four WSCs have been constructed accordingly. For example, in the case of WSC_1, four wrapper boundary input cells are chained together with s_1, and with four wrapper boundary output cells. Similar to multiple scan chain designs, using a greater number of WSCs can lead to a smaller test application time (TAT). It can be observed in Figure 1.7(b) that the length of the WSC is considerable greater than the length of the WSCs in Figure 1.7(c). Hence, the advantage of reducing TAT when parallel test access is available. While apparently simple, the core wrapper design problem for parallel test access has been shown to be an $NP-hard$ problem [56, 57] (an optimum solution is found in at least exponential computational time [58, 59]). The maximum number of WSCs is dictated by the width of the parallel test access mechanism, as it will be seen next.

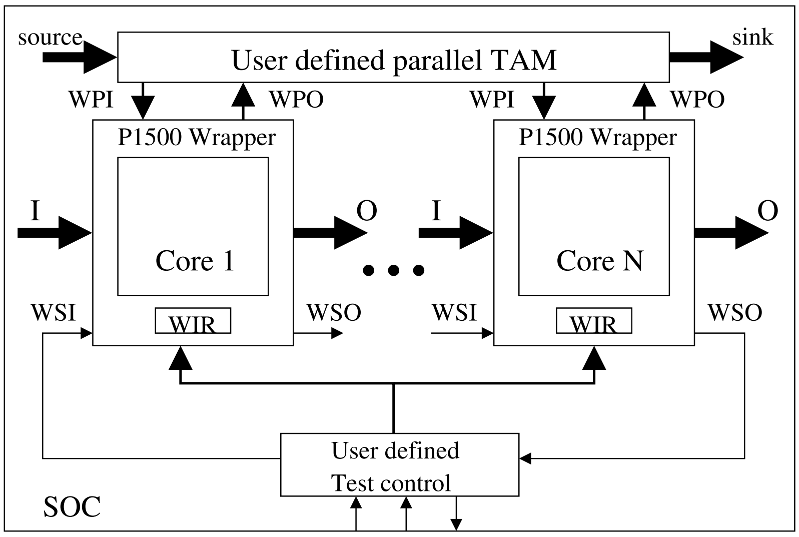


Figure 1.8. SOC with P1500 based cores wrappers [55]

Test access mechanism (TAM) design

It was illustrated previously that the core wrapper can be connected to either a serial test TAM, or a parallel one. In both cases, the system integrator has to ensure proper access to the core. Based on the IEEE P1500 core wrapper, in Figure 1.8 a SOC test scenario is illustrated when both, a serial and a parallel TAM are defined. The serial TAM is a one bit bus connected to the WSIs and WSOs of each core. The parallel TAM on the other hand is optional, its width is the system integrator's choice. The parallel TAM is in effect a bus of a given width connected to the WPIs and WPOs of the cores. The figure also shows the user defined test control which based on a known test scheduling scheme starts the test sessions corresponding to the various cores. The source and sink can be on-chip divided across multiple TAMs, each of them having different widths. The width of the TAM and the number of TAM lines used to connect the various cores may differ. This is illustrated in Figure 1.9, for a SOC comprising four cores connected to two TAMs. The test control and the WSIs/WSOs have been omitted for clarity. As it can be seen in Figure 1.9, the 40 bit TAM originating from the source is divided into two TAMs of 16 and 24 bits respectively. Cores 1 and 3 are using the maximum available TAM width – 8 TAM lines for WPI and WPO in the case of $Core1$, and 12 TAM line for WPI and WPO in the case of $Core3$. Cores 2 and 4, on the other hand are using only 6 and 10 TAM lines respectively. Determining the widths of the TAMs and the assignment of cores to TAMs depending on the system integrator's constraints, e.g., the maximum TAT, the maximum TAM width, and routing constraints, is known as TAM design. The TAM design problem

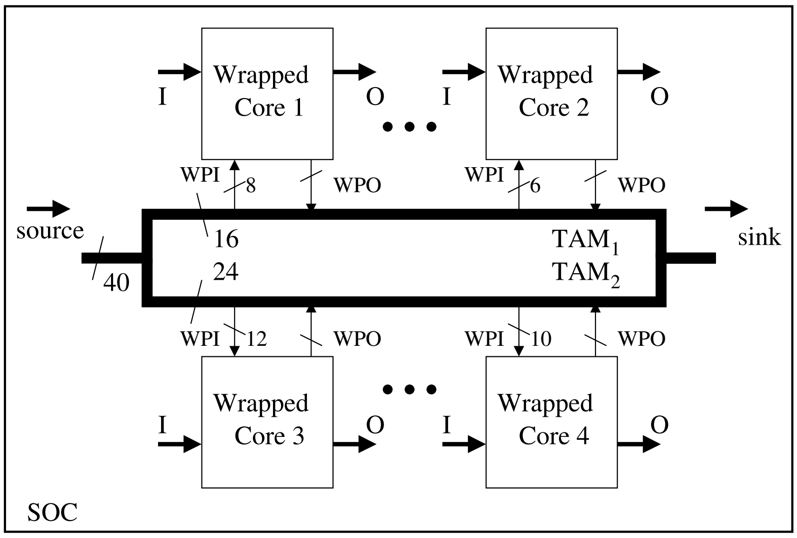


Figure 1.9. SOC with multiple TAMs

has been also shown to be $NP-hard$ [60, 61]. As illustrated in both Figures 1.8 and 1.9, the TAM is connected to the source and sink. These can be on-chip (BIST source and sink) and off-chip (ATEs source and sink). The ATE is illustrated in the following section.

1.2.2 Automated test equipment (ATE)

As emphasised in Section 1.1.1 and illustrated with the conceptual architectures in the beginning of Section 1.2, regardless of the embedded DFT method, each chip after manufacturing has to be tested using an automated test equipment (ATE). Testing implies applying a set of test stimuli to the device under test (DUT) and verifying the correctness of the test responses. The ATE is a precision measurement tool which is responsible for providing the required test patterns to the input of the DUT and for comparing the DUT responses to the fault free test responses [19]. A generic ATE – DUT interface is illustrated in Figure 1.10. The ATE is connected to a device interface board (DIB) which is connected to a wafer prober for testing of uncut and unpackaged wafers, or to a socket for testing packaged chips [38]. Due to the high precision electronics on the ATE the cost of these machines is considerable. It is estimated in the international roadmap for semiconductors (ITRS) [13] that the cost of an ATE can exceed \$20M. In addition, as it will be seen later in Section 1.3, the cost of the DIB can also considerably influence the overall cost of the ATE, and hence of the design. Thus, it is desirable to reduce the inter-

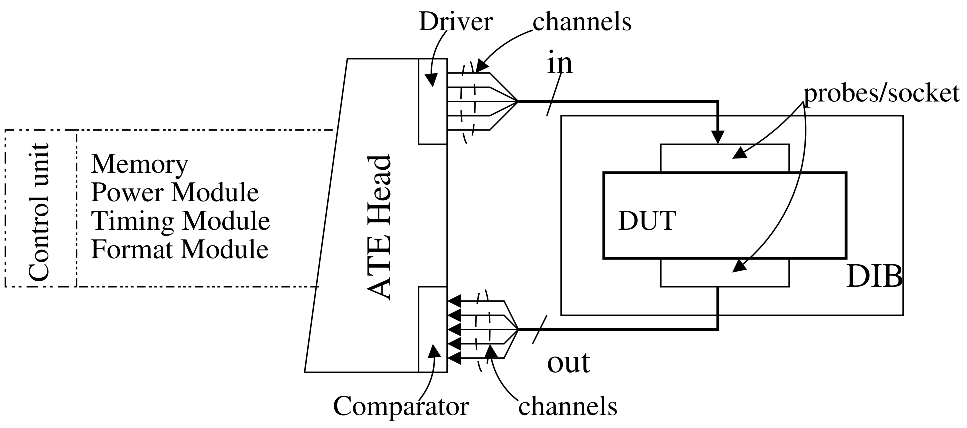


Figure 1.10. ATE – DUT interface

action and the changes required to the DIB, through elimination of this component from the cost of test. The test stimuli are applied to the DUT using the driver, and the DUT responses are compared to the fault free responses using the comparator as illustrated in Figure 1.10. The channels are the connection between the ATE and the DIB. The generic ATE illustrated in the figure comprises: memory – to store the test stimuli and required timing information; power module – to generate the required high and low voltages and precisely measuring the circuit outputs; format module – to prepare the data, to be sent to the DUT, considering the test stimuli and the timing information; timing module – to define the clock edges for each pin; and control unit – a computer which monitors the entire process, and stores the test result: good/bad chip. There are a number of parameters which characterise the ATE. These include: the number of *ATE channels* (or ATE test pins) – the number of chip pins which can be connected to the ATE; the *ATE frequency* – the frequency with which the ATE is capable of supplying data on the ATE channels; the *ATE accuracy* – the accuracy of the device to provide test data with respect to the ATE clock, e.g., for an ATE frequency of $1GHz$ the accuracy is of $\pm 200ps$ – the data will be accurate within $\pm 200ps$ from the rising edge of the clock.

The amount of time required to complete one test set is referred to as test application time (TAT). In this dissertation it is considered that the TAT is given in ATE clock cycles. As it will be seen in the following section, where the cost of test is analysed, the ATE has a substantial contribution to the overall cost of test. The main reason for this is that regardless of the type of methodology embedded within the chip, the chip has to go through the manufacturing testing process. And hence, it has to be tested using an ATE. In addition, due to the fact that ATEs are built using previously available technology, they are limited in providing the performances required by cutting edge chips.

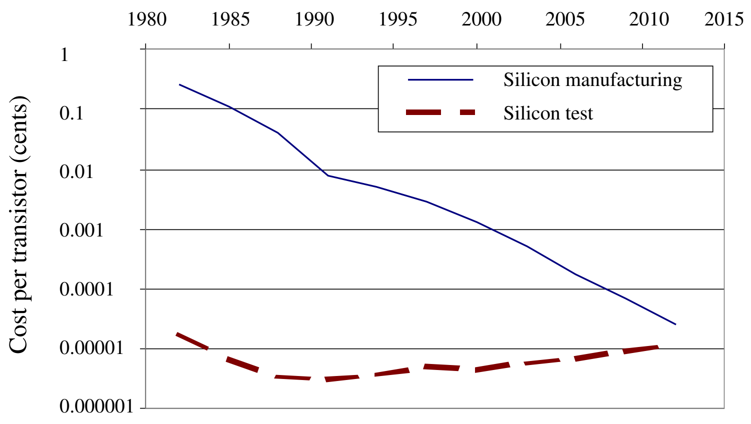


Figure 1.11. Test and manufacturing cost per transistor [13]

1.3 Cost of test

Testing SOCs requires the insertion of SOC DFT which facilitates the test of the embedded cores and the entire system. In addition to the SOC DFT problems illustrated in the previous section, another important issue is the cost of test. It is anticipated in the ITRS that, if the current trends are maintained, the cost of testing a transistor will approach and may even exceed the cost of manufacturing it [13]. This is illustrated in Figure 1.11, where the test curve will intersect the manufacturing curve by 2015. To motivate the work described in this thesis and at the same time highlight the importance of considering low-cost SOC test, in this section the cost of test and the main parameters which drive its continuous increase are analysed.

Conceptually the cost of test (C_{test}) [62] can be computed as given below:

$$C_{test} = C_{prep} + C_{exec} + C_{silicon} + C_{quality} \tag{1.1}$$

where:

- C_{prep} - the fixed cost of test preparation. It captures fixed costs of test generation, tester program creation, and any design efforts for incorporating any test related features, i.e., it includes all non-recurring costs, including software systems;
- C_{exec} - the cost of test execution. Consists of test related hardware and it includes: tester setup time, tester execution time, capital cost of the tester and capital equipment depreciation rate;
- $C_{silicon}$ - is the cost required to incorporate DFT features;

- $C_{quality}$ - is the cost of imperfect test quality. That is, the profit loss from performance degradation caused by the added DFT circuitry, the test of cost escape (the relation between fault coverage and fault occurrence - or simply yield), and cost of good dies being deemed faulty by imperfect test;

With SOCs comprising cores at different levels of abstractions provided by third party vendors, none of the above components are directly under the control of the system integrator. That is, both the core user and the system integrator influence them. For example, the system integrator ensures the proper TAM and on-chip test pattern source and sink through SOC DFT, however, the core vendor incorporates the core level DFT to ensure the required fault coverage (see Section 1.2). These two are related since the core level DFT influences the system level DFT requirements. In addition, the test quality ($C_{quality}$) depends on the fault coverage and the types of faults targeted by the core level DFT [37], hence these must be in concordance with system integrators' necessities [48], i.e., manufacturing process. Therefore, the more information the core vendor has about the final manufacturing process used by the system integrator the greater the chance the delivered core is suitable for the system integrator's needs. Driven by the need for high quality test and the assurance that the core performance is in concordance with the specifications, the system integrator prefers hard cores [1]. As noted in Section 1.1.2, these come with predictable area and performance, however, they lack flexibility. This lack of flexibility restricts the system integrator from performing fault simulation and ATPG, hence it is constraint to accepting the core's DFT and test information delivered with the core "as is". As it will be seen in Section 1.3.1, the cost of test execution (C_{exec}) is also dependent on the test time of the cores which is influenced by the core DFT methodology and the SOC DFT. Hence, this cost is also, to some extend, dependent on both the core vendor and the system integrator.

While in order to reduce the cost of test, all the four parameters are sought small, C_{prep} is a fixed cost per design, $C_{quality}$ is directly dependent on the core DFT methodology, and $C_{silicon}$ is a requirement for providing core level DFT. Hence, cost reduction in manufacturing test is possible if the C_{exec} is reduced. This is because each chip, regardless of the area overhead or the fault coverage obtained by using DFT, has to be physically tested. Therefore, while all the parameters are sought small, this work will address the issue of reducing the overall cost of test with special emphasis on the *cost of test execution*, detailed in the next section.

1.3.1 Cost of test execution

As shown in Equation (1.1), the cost of test execution is part of the cost of test and it includes: tester setup time, tester execution time, capital cost of the tester and capital equipment depreciation rate. The cost of test execution can be approximated using the following equation:

$$C_{exec} \approx C_{hw} + T_{tester} \times C_{tester} \tag{1.2}$$

where C_{hw} represents the cost of necessary hardware to perform the test (excepting the cost of the tester), C_{tester} represents the cost of the tester, and T_{tester} represents the time spent on the tester. These three components are analysed next.

Cost of extra hardware C_{hw} represents a recurring cost per design, and comprises the cost of the probes and sockets, and the cost of extending the device interface board (DIB) to accommodate for different DFT features. The cost of the sockets and probes is bounded by the physical handling of the DUT. The DIB, due to the mix of high and low frequency signals, requires special design techniques in order to provide the DUT with unaltered signals [63, 64]. In addition, the various extension required by various DFT techniques – e.g., external clocks, serialisation units, are also implemented on the DIB [65]. Therefore, the DIB can considerably influence the cost of the design, especially since the DIB is custom built for the exact pin specification of the DUT [13]. While the cost of the DIB cannot be eliminated completely from the cost equation, it is desirable to reduce the interaction and the changes required to the DIB, hence reducing the cost of C_{hw}.

Test time The semiconductor industry is building 300nm fabs capable of producing up to 2000 wafers per week. With the average test time of a wafer of up to 24 hours [9], the test execution time clearly becomes an important factor in the overall test cost. To illustrate the importance of T_{tester}, Figure 1.12 shows the increase in test time relative to 1999. It becomes clear from the figure that the increase in test time is considerable, and the trend is an ascending one. T_{tester} from the Equation (1.2) can be further detailed as illustrated below:

$$T_{tester} \approx T_{inactive} + T_{active} \tag{1.3}$$

where, the time spent on tester T_{tester} is divided into the time when the tester is just providing a socket for the DUT, since it is inactive ($T_{inactive}$), i.e., like in the case of internal-only

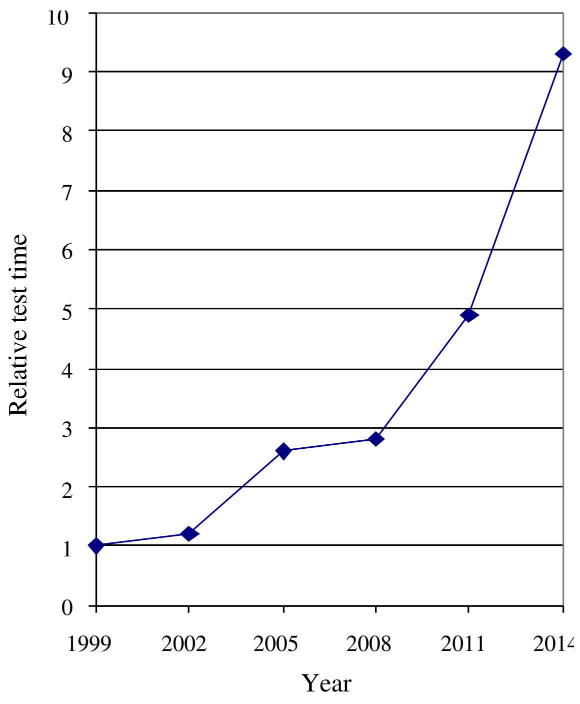


Figure 1.12. Test time increase relative to 1999 [15]

testing where the interaction between the tester and the DUT is reduced to a minimum; and the time when the tester is actively testing the DUT (T_{active}). $T_{inactive}$ is mainly due to inefficient tester usage, for example, when a BIST session is started the LFSR will generate patterns for a given number of clock cycles. In this time, the tester is not used, leading to inefficient tester usage. Therefore, $T_{inactive}$ main increase the cost since the DUT has to physically remain connected to the tester until the entire test process is finished.

T_{active}, on the other hand, is determined by the volume of test data (VTD), needed to test the DUT, and the ATE parameters, as illustrated below:

$$T_{active} = \frac{VTD}{n_{ate} \times f_{ate}} \tag{1.4}$$

where, n_{ate} represent the number of ATE channels used for testing the DUT, and f_{ate} represents the ATE operating frequency. The product $n_{ate} \times f_{ate}$ is also referred to as the bandwidth. The test time increase illustrated in Figure 1.12 is due to the continuous increase in VTD and the insufficient ATE bandwidth when compared to the requirements. These are illustrated next.

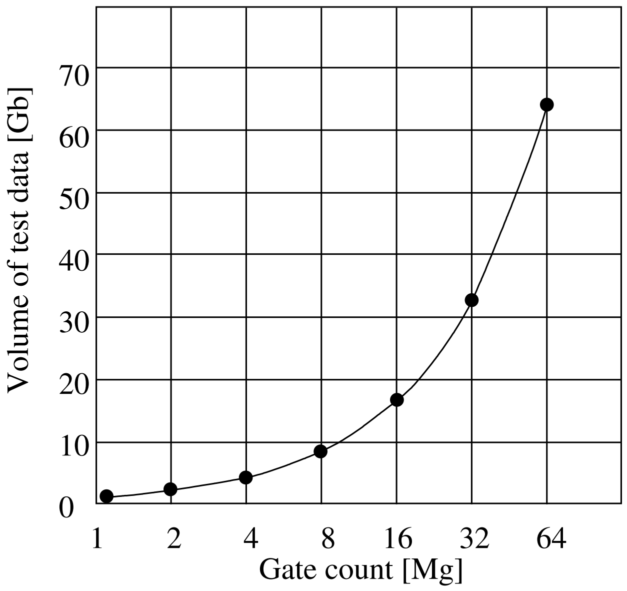


Figure 1.13. Gate count vs. volume of test data trend [66]

The VTD for single scan chain designs can be approximated using equation (1.5):

$$VTD \approx scan\_chain\_length \times no\_patterns \tag{1.5}$$

i.e., the VTD is given by the length of the scan chain multiplied with the number of scan patterns [9]. Since, the number of scan cells is directly related to the number of gates (typically, 1 scan cell to every 20 gates) [9], the number of scan cells will increase directly proportional to the number of gates. The number of gates, according to Moore's "law", doubles every 18 months, and hence, the number of scan cells will double every 18 months as well. Since, the VTD is proportional with the number of scan cells, it will also double every 18 month, and therefore, the ATE memory requirements will also **double** every 18 month. This is illustrated in Figure 1.13, where the increase in VTD (given in giga bytes) with the increase in design size (given in million of gates) follows the above rule. In addition to the above, further increase in VTD is expected due to DFT methods' dependencies on scan to deliver their test vectors. For example, delay and open faults require pairs of scan test vectors when tested. In order for the ATE to account for the continuous increase in VTD, up to 60% of the invested capital for ATE upgrade is needed to meet the new memory capacity requirements.

From the above discussion a potential solution to reduce test time, and hence reduce one of the factors that contribute to C_{exec}, is to reduce the VTD and inefficient ATE usage.

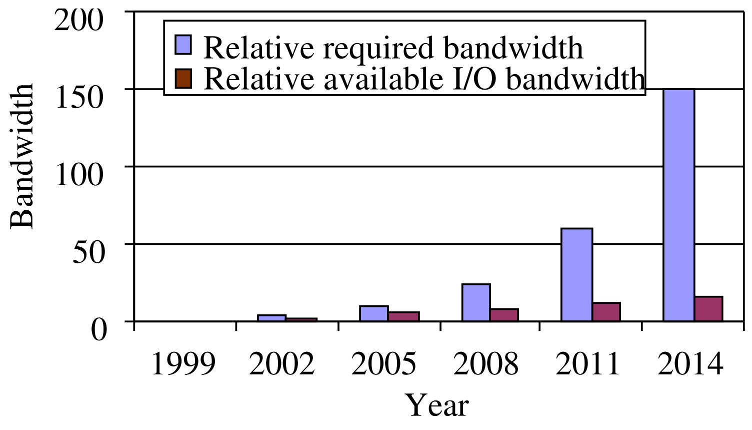


Figure 1.14. Growth in complexity and bandwidth relative to 1999 [15]

The alternative solution, of increasing the bandwidth, is analysed next.

It is immediately derived from Equation (1.4) that the test time can be reduce if the bandwidth is increased. This however, encounters the following limitations. Firstly, the available ATE bandwidth does not increases at the same rate as the IC bandwidth [14]. And secondly, not all the chip pins can be used for testing, and therefore, there is a gap between the required bandwidth and the available bandwidth for test [15] (see Figure 1.14). Finally, increasing the available ATE bandwidth is followed by an increase in ATE cost, hence, reducing test time at the expense of an increase in C_{tester}. The cost of the tester is analysed next.

Cost of tester The cost of the tester (C_{tester}) from Equation (1.2), can be further divided into the cost of the test equipment itself (C_{ate}), and the various cost of handling the test equipment ($C_{handling}$)

$$C_{tester} \approx C_{ate} + C_{handling} \tag{1.6}$$

The increase in pin count and the mixture of various technologies in system chips, causes ATE costs - for running the tests as well as the equipment itself - to increase significantly according to [13]. Furthermore, ATEs with higher accuracy are much more expensive than lower accuracy ATEs [11]. Thus, increasing the ATE operating frequency and the number of ATE pins, hence the bandwidth, does not provide viable solutions for low-cost system test, as they increase the cost of the ATE itself. Therefore, in order to keep the cost of ATE low, *reduce pin count test (RPCT)* methodologies, and methods which *reduce the bandwidth requirements* are needed. It should be noted that there is a clear distinction between RPCT and reduced bandwidth requirements. The former refers to reducing the number of pins for test, while the latter refers to reducing the amount of time needed to transport data from the ATE to the CUT using the available ATE channels and fully exploiting the features of the ATE. With constant changes in the VTD and required accuracy, the ATE lifespan is considerably reduced. Therefore, avoiding ATE upgrades and reusing existing ATE infrastructure will also considerably contribute to cost reduction [67].

From the issues outlined above, a methodology which has the potential to providing a low-cost solution for SOC test, will have: low volume of test data, reduced pin count test (RPCT), reduced bandwidth requirements and will efficiently use the ATE resources without requiring any DIB extensions. In addition, when addressing the above factors care must be taken to ensure core reuse by leveraging the IP based SOC design, hence refraining from changing the core. Therefore, an additional condition in a low-cost methodology for testing core-based SOC is seamless design integration and core reuse.

1.4 Thesis contributions and organisation

Providing a low-cost test solution for core based SOC does not only require the understanding of the factors which drive the cost, but it also imposes the need to comprehend the implications and limitations of previous approaches which addressed these factors. Therefore, a comprehensive analysis of previous work is given in Chapter 2, which motivates the usage of test resource partitioning [68], i.e., the partitioning of some test resources on-chip and some off-chip alleviating the burden on the ATE, with main focus on test data compression. Based on this analysis, and with the prime focus on reducing the cost of test the next four chapter represent the contributions of this dissertation.

Chapter 3 presents a detailed characterisation of test data compression methods with respect to test application time, volume of test data and area overhead and proposes a new test data compression method. The proposed compression method is based on a new coding scheme, named Variable-length Input Huffman Coding (VIHC), and a novel on-chip decoder. The proposed VIHC compression/decompression method leads to simultaneous improvement in application time, volume of test data and area overhead when compared to previous work. With respect to the cost factors detailed in the previous section, this chapter addressed the issue of volume of test data, bandwidth requirements and on-chip overhead.

Due to the resource partitioning, various synchronisation issues arise. These can increase the cost of the device interface board (DIB) and they can lead to a recurring cost per design. This issue is addressed in Chapter 4, and a generic on-chip test solution for single scan chain based cores is proposed. The proposed solution leads to complete elimination of DIB extensions, hence eliminating this component from the cost of the design. Therefore, when combined with the solution proposed in Chapter 3, a low-cost test solution for single scan chain core based SOCs is proposed.

While Chapters 3 and 4 focus on test data compression, Chapter 5 provides a complementary solution to reducing the volume of test data in core based SOCs without the need for on-chip area overhead through elimination of useless test data. Useless test data is identified as one of the contributors to the total amount of test data, comprising the padding bits necessary to compensate for the difference between the lengths of different chains in multiple scan chains designs. Therefore, Chapter 5 investigates the sources of useless test data and proposes a new test methodology which combines careful core wrapper design and ATE memory management capabilities to reduce the memory requirements for multiple scan chains designs.

Chapter 6, provides an integrated solution for deterministic SOC testing. The proposed solution is based on an on-chip test data decompression architecture which exploits the advantages of the distribution architecture proposed in Chapter 4, leading to elimination of the synchronisation overhead. It reduces the volume of test data, test application and the power dissipation during scan by exploiting the features of the core wrapper design algorithm proposed in Chapter 5. The proposed solution is scalable and programmable and, since it can be considered as an *add-on* to a test access mechanism of a given width, it provides seamless integration with any design flow. Thus, the proposed integrated solution leads to an efficient low-cost test methodology for core based SOC addressing all the factors outlined in the previous section.

Finally, Chapter 7 summarises the presented work and concludes this thesis. The contributions outlined in Chapters 3, 4, 5 and 6 resulted in original work published, or submitted for publication, in [69, 70, 71, 72, 73, 74, 75, 76] and itemised in Appendix G.

Chapter 2

Previous work

As illustrated in the previous chapter, the problems brought forth by the high level of integration are, on the one hand, the SOC test challenges (also introduced in the previous chapter as SOC DFT issues), and on the other hand, the high cost of test. To provide an overview of previous work which aimed at mitigating these two problems, in Section 2.1 a comprehensive review of SOC DFT approaches is provided, and in Section 2.2 approaches which aim at reducing the cost of test with the main focus on volume of test data reduction are detailed. Finally, Section 2.3 provides concluding remarks for this chapter.

2.1 SOC DFT

Testing complex SOCs requires new design and test techniques. These range from simply providing a direct access to the core terminals [77], to complete test access mechanism design under various system constraints [61]. To provide a comprehensive overview of work undertaken in this area, previous approaches are categorised in: (i) **system test driven**, which exploit various particularities of the cores interfaces, and (ii) **core wrapper driven**, which require a core wrapper and a test access mechanism (TAM).

System test driven approaches The simplest approach to test cores is to simply multiplex the core terminals to the existing SOC pins [77]. While this apparently solves the TAM problem, due to a large number of cores and high number of core terminals, it incurs large routing and area overhead. Furthermore, with the increase in the ratio of core terminals to chip pins, the approach is reaching its limits.

Another approach is to exploit isolation rings. An isolation ring is a boundary scan chain which provide test isolation for the core. Since the isolation rings add logic on the functional interconnect between cores they may affect critical timing paths between cores. Exploiting these isolation rings, Touba and Pouya [16, 18] provided a method for designing partial isolation rings which avoid adding logic on critical timing paths without affecting the fault coverage. As illustrated in the previous chapter, user defined logic (UDL) is an unavoidable component of the SOCs. Exploiting the UDL another solution for testing embedded cores has been provided [17], where the UDL is modified to allow easy access of test data to the CUT. In [78], the inputs and the outputs of the cores are isolated using "test collar" cells. These cells can then be connected to variable-width buses for test data and control. A disadvantage of this approach is that if the test buses are widened, then the routing overhead can adversely influence the routability of the ICs.

A different system test strategy has been proposed in [79], which assumes the existence of core transparency for test data transport. While, core transparency will reuse most of the existing test infrastructure, the cores need to be made transparent by the core provider in order for this approach to be effective. If this is not the case, the system integrator requires knowledge of the core internals and will resort to core redesign. Hence, in core based SOC environments the applicability of this approach is somewhat limited. A similar approach has been proposed in [80], where the cores are equipped with a bypass mode. This mode will facilitates data transfer from the core's inputs to its outputs similar to core transparency. While the approach eases the problem of finding paths from the SOC inputs to the CUT, it adds multiplexers and serialisation/de-serialisation units to the cores. These, incur extra area and routing overhead.

Since cores are based on ICs, which in order to be tested on a board required an IEEE 1149.1 [36] compliant test port, in [81] a test access architecture has been proposed which exploits the IEEE 1149.1 test port. In [8] the addressable test port has been proposed. The addressable test ports provide the capability of addressing the cores to be tested and, once addressed, effectively test the addressed core [8].

Assuming a given access mechanism to the cores, the order of testing the cores, also referred to as test scheduling, is required to ensure minimum TAT. Therefore, approaches which address test scheduling considering cores with multiple test sessions have been proposed, e.g., one core can have one (or more) BIST session(s) and one (or more) external test session(s). In [82] a test scheduling algorithm has been proposed where each BIST-ed core has been assumed with its own BIST controller, and the external test was limited to only one core at a time. The above restrictions have been removed in [83], where a mixed integer linear programming model has been provided. The proposed solution can handle cores which shared BIST logic as well as cores that have multiple test sets.

The above presented approaches allow testing of embedded cores by exploiting their various features, e.g., the UDL, the isolation rings and transparency. However, to ease the interoperability between the core provider and the core user, standard and scalable mechanisms are required. Core wrapper driven approaches are detailed next. Note that some of the above approaches can be also applied in a core wrapper driven environment, e.g., [8].

Core wrapper driven approaches The core wrapper design approaches begun with the effort to develop a standard for easing the interoperability between the core provider and the core user. The "TestShell" proposed in [84] represents the basis for the P1500 core wrapper. The TestShell is scalable and supports the operating modes required by the IEEE P1500. Since the initial approach [84] presents the disadvantage of unbalanced wrapper scan chains (i.e., the scan chains formed by the internal scan chains and the core's inputs/outputs, tend to have unequal lengths), heuristics have been proposed, in [56] and [57], to balance the wrapper scan chains with focus on TAT.

Imposing a core test wrapper splits the SOC test issues from a system integrator problem into a joint core provider – core user problem. While the former has to provide the core test wrapper or at least the CTL description necessary to construct it [53, 50, 54], the latter has to design the test access mechanism. In [85] three test architectures for testing multiple scan chain cores have been proposed: *multiplexing architecture* - where one test bus is multiplexed between multiple cores; *daisy chain architecture* - where multiple cores are daisy chained to one test bus; and, *distribution architecture* - where the test bus can be divided (distributed) among different cores. With respect to the above, two main research directions have evolved. The first one is based on using "TestRail" [84], which combines the daisy chain and a distribution architecture; while the second one is using test buses as TAM, hence combining the multiplexing and the distribution architecture. Using these two types of buses a number of approaches have emerged targeting TAM design when different design constraints are targeted, such as minimising TAT, routing overhead, power dissipation, and memory requirements.

With respect to TestRail, the work in [61] addresses the TAM design problem in conjunction with minimising TAT considering problem formulations ranging from assuming fixed width test buses to the design of the entire TAM under given maximum test bus width constraint, using the core wrapper design from [84]. In [60], place and route and the power dissipation constraints have been also considered in the TAM design problem with primary focus on TAT minimisation. The approach in [86] provides a TAM design algorithm which restricts the TAM width to be greater or equal to the number of cores in the system. This limitation has been removed in [87], where the *TR-ARCHITECT* algorithm has been proposed.

With respect to TAM design based on test buses, in [88], by managing the number of bridges (e.g., MUXes, controllable buffers and bypass routes) between the cores, the TAM is designed for minimum TAT and bridge area overhead. In [57] the approach in [56] has been generalised such that the wrapper design, TAM and TAT minimisation have been combined into a unified problem formulation. Various heuristics with different computational complexity, and considering various constraints, have been proposed in [89, 90, 91, 92, 93]. In [89], rectangle packing has been used to model the TAM design problem and heuristics have been provided to solve the defined problem. In [90], the volume of test data (VTD) has been added as a new parameter in the cost function for TAM design, while in [91] the ATE memory depth has been considered as a constraint for multi-site testing. 3D-rectangle packing has been proposed in [92], where the approach in [89] has been extended with power dissipation as an additional constraint in the TAM design problem. Since, rectangle packing requires complex algorithms when the dimensions of the packing problem increase, in [93], $k-tuples$ have been introduced to efficiently model the various constraints for TAM design.

As noted above, the problem of VTD reduction has been addressed in conjunction with TAM design [87, 91, 90]. In these approaches, the VTD has been taken into account in the cost function which drives the TAM design heuristic. However, there is an inherent trade-off, in core wrapper design, between the volume of test data and TAM width, and consequently between the volume of test data and the TAT. Hence, targeting TAT and test data volume minimisation will produce a TAM design which provides a trade-off between the two as illustrated in [90]. This trade-off leads to inefficient ATE memory utilisation, and it is due to *useless* test data, which can adversely influence the memory requirements for a given test bus width as it will be seen in Chapter 5.

Designing the core test wrapper and the TAM is performed once for a SOC, and hence it is related to the cost of preparation (C_{prep}). However, it also influences the cost of execution (C_{exec}) due to the memory requirements and the TAT. In the following section test resource partitioning is overviewed.

2.2 Test resource partitioning

With respect to the two conceptual SOC test architectures illustrated in Figures 1.4 and 1.5 (see Section 1.2 on Page 10), two types of approaches have emerged: (a) external-only approaches (no on-chip overhead is required) and (b) internal-only approaches (no off-chip interaction is required). The external-only approaches (a) target volume of test data (VTD) reduction. These include test data compaction [42, 94, 95] and methods which reduce the amount of data transfered from the workstation to the ATE [96, 97]. While these approaches effectively reduce the VTD, they do not reduce the ATE bandwidth requirements. The internal-only approaches (b) are based on built-in self-test (BIST), i.e., pseudo-random and/or deterministic BIST. The interaction between these and the ATE is kept to a minimum, and therefore, BIST is one of the advocated solutions for SOC test [13]. However, to increase the fault coverage of pseudo-random BIST intrusive techniques such as test point insertion are required [98]. Such techniques are not acceptable in a core-based SOC environment since they lead to a major core redesign effort, defeating the very purpose of core based SOC design: short time-to-market through core reuse. In addition, when intellectual property (IP) protected cores are targeted, intrusive techniques cannot be applied. To alleviate the problems related to pseudo-random BIST, deterministic BIST [99, 100, 101, 102, 103] approaches have emerged. These, however, trade-off test quality and area overhead [104], and as also illustrated in [15] the test time can be considerable.

As the above approaches have their disadvantages, in this dissertation test resource partitioning [4] is investigated. Test resource partitioning implies the allocation of some test resources on-chip, and some test resources off-chip, on the ATE. This is illustrated in Figure 2.1. The ATE memory will hold test data (for both test stimuli and test responses) and test control information. The SOC will have an on-chip decoder to decode the data from the ATE, and an on-chip compactor, which will encode the test responses sent to the ATE. The amount of test data sent from the ATE to the on-chip unit is desired much smaller than the amount of test data sent from the on-chip unit to the CUT; this is en-

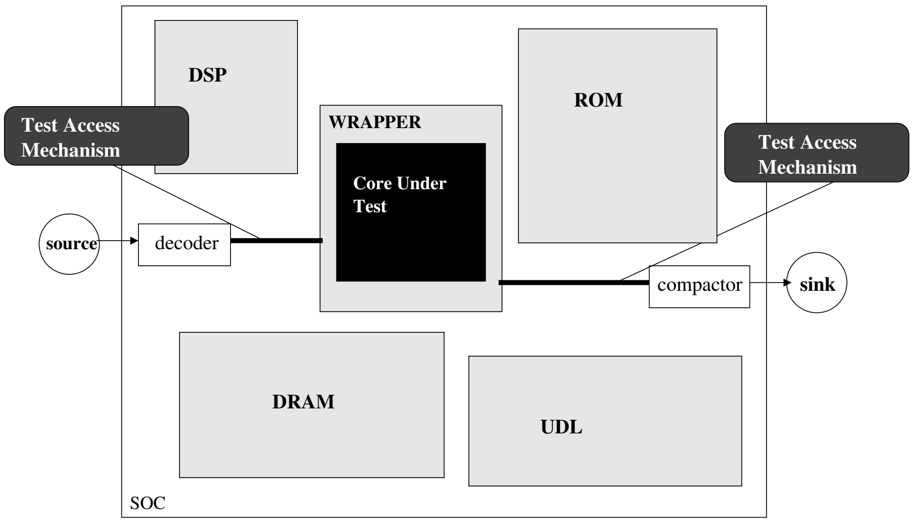


Figure 2.1. System-on-a-Chip and test resource partitioning

sured through test data compression. Similarly, the amount of data going into the on-chip compactor is desired much larger than the amount of data sent to the ATE; this is ensured through test response compaction. Test response compaction and test data compression are detailed in the following sections.

2.2.1 Test response compaction

Test response compaction, is the process which reduces the volume of test responses sent from the CUT to the ATE. This process is usually performed using signature analysers [38]. Due to the fact that signature analysers may suffer from aliasing (i.e., the case when the information comprised in the signature is not enough to differentiate between a fault and a fault free response), special schemes may be required. One common solution to the aliasing problem is to have the entire (or part of the) signature analyser's content regularly sent to the ATE. This will increase the amount of information known about the circuit responses, and therefore decrease the aliasing probability [38]. In addition to the above, alternative methods have been proposed for memories [105, 106] and for logic circuits [107, 108]. The approaches in [105, 106] compress the responses obtained when memory testing is performed, while the methods in [107, 108] proposed output compactor methods which are capable of generating viable responses even in the presents of undefined values in the circuit responses. These undefined values are due to uninitialised flip-flops and due to multi-clock domain crossing.

2.2.2 Test data compression

Test data compression (TDC) is a test resource partitioning scheme whose main target is the reduction of volume of test data sent from the ATE to the CUT. TDC implies the usage of a compression scheme during test set preparation, and an on-chip decompression unit during test set application. By introducing on-chip decompression hardware it reduces the load on the ATE, and therefore it simplifies the ATE channel capacity and speed requirements. A variety of TDC methods have been proposed. These can be classified into approaches (i) which extend deterministic BIST, (ii) which exploit the spareness of care bits in the test set, and (iii) which exploit the regularities within the test set.

(i) **Extending deterministic BIST** The first category [109, 110, 111, 112, 113] (i) enhances deterministic BIST approaches, thus alleviating some of their drawbacks (e.g., area overhead, test application time). In [109], an approach which combines external and BIST generated patterns is proposed, alleviating the need for long pseudo-random sessions. In [110] by using variable length re-seeding, a solution is proposed which has less area overhead and shorter test application time than convectional re-seeding techniques. In [111], weighted pattern BIST is combined with external test sources to account for the large area overheads incurred by the conventional weighted pattern BIST. While these approaches [109, 110, 111] effectively reduce the volume of test data, they have two main disadvantages when viewed from the system integrator's perspective. Both disadvantages, are because these approaches rely on pseudo-random BIST to cover the easy to detect faults. Therefore, they require fault simulation in order to ensure the test quality, and, they incur a long ATE inactivity period which leads to inefficient ATE usage. To alleviate the fault simulation problem, the approaches in [112, 113] considered a different approach: instead of letting the pseudo-random session run freely, the on-chip LFSR is controlled such that the generated test set covers a predetermined deterministic test set. While shown to be effective, these approaches can also lead to long ATE inactivity periods. Long ATE inactivity leads to inefficient ATE usage and it may also adversely influence the cost of test [114].

(ii) **Exploiting care bit spareness** The second category [115, 116, 114, 107, 117] (ii), by exploiting care bit spareness, can reduce the volume of test data when the percentage of "don't care" bits is high. The underlining idea behind these approaches is that using a simple structure, which can be easily mapped into linear equations, the input values of the structure are determined such that the generated outputs can be mapped over a deterministic test vector. To implement this idea, the approach proposed in [115] uses the scan chains of a providing core; and [116] uses a network of XORs driven by multiple ATE channels. In [114] an LFSR and an XOR network are used to implement the approach, while [107] uses a ring generator and a phase-shifter, and [117] uses multiple LFSR structures. While, these architectures can efficiently reduce the VTD and test application time (TAT), they suffer from lock-out situations [99], i.e., the inability of the on-chip decoder to generate a given test pattern. These lock-out situations can degrade test quality, increase test control complexity, as well as negatively influence TAT. When the percentage of "don't care" bits decreases, these lock-out situations can increase significantly. While the approaches in [114, 107, 117] can handle the increase in care bit density at the expense of extra on-chip circuitry and less volume of test data and TAT reduction, the approaches in [115, 116] are embedded into the ATPG procedures. The usage of ATPG may be prohibited in the core based design flow where the system integrator cannot access the structural information of the embedded cores, and hence, the applicability of these approaches from the system integrators perspective is limited.

(iii) **Exploiting test set regularities** The third category [118, 119, 120, 121, 122, 123, 124, 125, 126] (iii), exploits the regularities within the test set and does not require any ATPG re-runs. Since, ATPG procedures require fault simulation, which may be prohibited in IP environments, and long ATE inactivity periods can increase the cost (see (i) and (ii) above), this thesis investigates test data compression methods with prime focus on the third category. Methods which use test set regularities exploit different features of the test set, such as runs of '0's, and/or small differences between consecutive test vectors. These methods are based on data compression coding schemes (e.g., run-length, Huffman, Golomb [127]) [118, 119, 121, 122, 123, 126] and custom coding schemes [120, 124, 125]. These coding schemes are implemented using on-chip hardware decoders or on-chip software decoders, as detailed next.

With respect to on-chip hardware decoders, Iyengar et al. [118] proposed a new built-in scheme for testing sequential circuits based on statistical coding. Results indicate that the method is suitable for circuits with a relatively small number of primary inputs. To overcome this disadvantage a selective coding (SC) algorithm has been proposed [121].

The method splits the test vectors into *fixed-length* block size patterns[1] and applies Huffman coding to a carefully selected subset. In [119], test vector compression is performed using run-length coding. The method relies on the fact that successive test vectors in a test sequence often differ in only a small number of bits. Hence, the initial test set T_D is transformed into T_{diff}, a difference vector sequence, in which each vector is computed as the difference between two consecutive vectors in the initial test set. Despite its performance, the method is based on variable-to-fixed-length codes and, as demonstrated in [122], these are less effective than variable-to-variable-length codes. Chandra and Chakrabarty [122] introduced a compression method based on Golomb codes. Both methods, [122] and [119], use the difference vector sequence, hence, in order to generate the initial test set on-chip, a cyclical scan register (CSR) [119, 122] architecture is required (see Figure 2.2). In [123], a new method based on frequency-directed run-length (FDR) codes is proposed. The coding scheme exploits a particular pattern distribution, and by means of experimental results it is shown that the compression ratio can be improved when compared to [122]. In [125], a method which exploits the correlation between two consecutive parallel loads into multiple scan chains has been proposed. Since the method uses multiple scan chains it reduces the testing time, however, the required control signals may negatively influence the compression ratio and control complexity. Recently, in [126] a method which employs the LZ77 compression algorithm has been proposed, which by using the boundary scan of the chip and adding on-chip decompression hardware reduces the volume of test data.

In addition to the above approaches, which require an *on-chip hardware decoder*, methods which assume the existence of an embedded processor [120, 124], hence an *on-chip software decoder*, were also proposed. The method in [120] is based on storing the differing bits between two test vectors, while the method in [124] uses regular geometric shapes formed only from '0's or '1's to compress the test data. Regardless of their benefits, in systems where embedded processors are not available or where embedded processors have access only to a small number of embedded cores, these on-chip software decoder based methods are not applicable.

A particular advantage of methods which use data compression coding schemes is that they are capable of exploiting the ever increasing gap between the on-chip test frequency and the ATE operating frequency. While in the past this gap has been exploited at the

[1]It should be noted that throughout the thesis, a *pattern* refers to an *input pattern* used in the coding scheme and not to a *test pattern*. A coding scheme uses *fixed-length* patterns if all the input patterns have the same length, otherwise, it uses *variable-length* patterns.

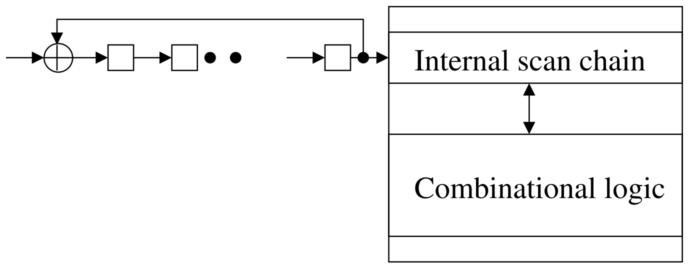


Figure 2.2. Cyclical scan register architecture

cost of *multiple ATE channels* [128, 65], hence increasing the bandwidth requirements, approaches which use data compression coding schemes can leverage the frequency ratio without the penalty of extra ATE channels. This leads to feeding data to the cores using the on-chip test frequency instead of using the external ATE operating frequency as in conventional approaches.

2.3 Concluding remarks

SOC DFT has been addressed in previous work as illustrated in Section 2.1. However, as noted in [90] minimum test application time and volume of test data reduction are conflicting optimisation objectives in test access mechanism (TAM) design. This leads to the investigation of memory requirements for core based SOCs in Chapter 5, which also provides a novel test methodology to address this problem.

Test data compression is a test resource partitioning scheme which provides a possible solution for cost of test since it reduces the volume of test data and the burden on the ATE. However, previous approaches have focused only on some of the factors which drive the cost of test (e.g., volume of test data), while others have been omitted (e.g., additional test hardware due to synchronisation requirements). As will be seen in this dissertation all the factors which drive the cost of test need to be accounted for, otherwise a partial low-cost test solution is developed. Therefore, Chapter 3 simultaneously reduces the volume of test data, test application time and area overhead when compared to previous work; Chapter 4 eliminates the additional test hardware imposed by synchronisation requirements; and Chapter 6 provides an integrated solution for low-cost core based SOC test.

Chapter 3

Volume of test data reduction through compression

Test data compression is a test resource partitioning scheme whose main target is the reduction of volume of test data. By introducing on-chip decompression hardware it reduces the load on the ATE, and therefore it simplifies the ATE channel capacity and speed requirements. As illustrated in Chapter 2, a number of approaches for test data compression have been proposed (e.g., [121, 123, 122]). Therefore, two aspects are targeted in this chapter. Firstly, in order to understand the factors which influence the performances of the previously proposed compression methods, an analysis of compression methods based on data compression coding schemes is given. The test data compression environment (TDCE) is introduced and previous work is characterised with respect to the three TDCE parameters: compression ratio, area overhead and test application time. Secondly, a new coding scheme named Variable-length Input Huffman Coding (VIHC), and its corresponding parallel on-chip decoder are proposed. The two combined attain simultaneous improvement in all the three TDCE parameters when compared to previous work.

The remainder of this chapter is organised as follows. Section 3.1 provides the taxonomy on test data compression and the previous work analysis with respect to the presented taxonomy. Sections 3.2 and 3.3 give the proposed compression algorithm and the associated decompression architecture. Section 3.4 provides extensive experimental results and comparisons with previous approaches. Finally, Section 3.5 concludes this chapter.

3.1 Test data compression environment

To better understand how the volume of test data, the area overhead and the testing time can be simultaneously reduced, this section introduces the test data compression environment (TDCE) and characterises the TDCE with respect to the factors which influence it. Three previous approaches are also analysed from the TDCE parameter's standpoint. Testing in TDCE implies sending the compressed test data from the ATE to the on-chip decoder, decompressing the test data on-chip and sending the decompressed test data to the core under test (CUT). There are two main components in TDCE: the **compression method**, used to compress the test set off-chip, and the associated **decompression method**, based on an **on-chip decoder**, used to restore the initial test set on-chip. The on-chip decoder comprises two units: a unit to identify a compressed code and a unit to decompress it. If the two units can work independently (i.e., decompressing the current code and identifying a new code can be done simultaneously), then the decoder is called **parallel**. Otherwise, the decoder is referred to as **serial** (i.e., when, while decompressing the current code, no new code can be identified)[1].

3.1.1 TDCE characterisation

Testing in TDCE is characterised by the following three parameters: **(a) compression ratio** which identifies the performance of the compression method, and the memory and channel capacity requirements of the ATE; **(b) area overhead** imposed by the on-chip decoder (dedicated hardware or on-chip processor); and **(c) test application time** given by the time needed to transport and decode the compressed test set.

There are a number of factors which influence the above parameters: the *mapping and reordering algorithm*, which prepares the test set for compression by mapping the "don't cares" in the test set to '0's or '1's, and by reordering the test set; the *compression algorithm*, which based on a coding scheme compresses the initial test set; the *type of input patterns* used as input by the coding scheme, which can be of fixed or variable lengths; the *length of the pattern*, which is the maximum allowed length of the input pattern for the coding scheme; and the *type of the on-chip decoder*, i.e., serial or parallel.

The relationship between these factors and the two components of the TDCE is illus-

[1]The reader is referred to Chapter 4 Section 4.1.1 for more detail about the two types of decoders

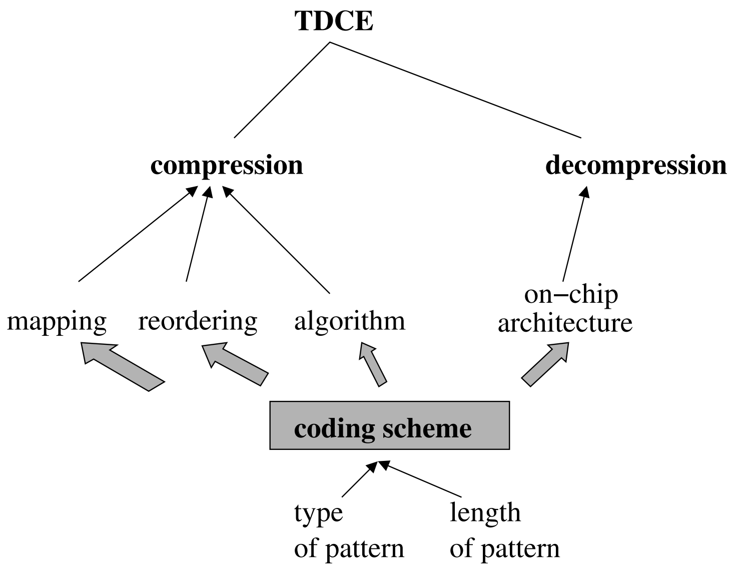


Figure 3.1. TDCE dependency map

trated in Figure 3.1. As it can be seen in the figure, the *coding scheme* not only determines the compression factors (i.e., mapping/reordering and compression algorithm), but it also influences the decompression factor (i.e., on-chip architecture). Hence, the selection of the coding scheme is a *key* element that determines the factors that influence the three TDCE parameters, which are discussed next:

(a) Compression ratio Using patterns of various types and of variable lengths, the compression algorithms exploit different features of the test set. Mapping and reordering the initial test set emphasises these features. Therefore, the compression ratio is influenced firstly by the *mapping and reordering algorithm*, and then by the *type of input patterns* and the *length of the pattern*, and finally by the *compression algorithm*.

(b) Area overhead Area overhead is influenced firstly by the *nature of the decoder*, and then by the *type of the input pattern* and the *length of the pattern*. If the decoder is serial then the communication between the two units (code identification and decompression) is already at hand. However, if the decoder is parallel, then communication between the two units must be explicitly provided, which can lead to increased control complexity and consequently to higher area overhead. Depending on the *type of the input pattern* different types of logic are required to generate the pattern on-chip. For example, if the coding scheme uses fixed-length input patterns, then a shift register is required to generate the patterns, however, if variable-length input patterns (runs of ’0’s for example) are used, then counters can be employed to generate the patterns. Since the *length of the pattern* impacts the size of the decoding logic, it also influences the area overhead.

(c) Test application time (TAT) TAT is firstly influenced by the **compression ratio**, and then by the *type of the on-chip decoder*, and the *length of the pattern*. To illustrate the factors that influence TAT, consider the ATE operating frequency as the reference frequency. Minimum TAT (min_{TAT}) is given by the size of the compressed test set in ATE clock cycles. However, this TAT can be obtained only when the on-chip decoder can **always** process the currently compressed bit before the next one is sent by the ATE. In order to do so, the relation between the frequency at which the on-chip decoder works and the ATE operating frequency must meet certain conditions. The *frequency ratio* is the ratio between the on-chip test frequency (f_{chip}) and the ATE operating frequency (f_{ate}). Consider that the *optimum frequency ratio* is the frequency ratio for which min_{TAT} is obtained. Since the min_{TAT} is given by the size of the compressed test set, increasing the compression ratio would imply further reduction in TAT. However, this reduction happens *only* if the optimum frequency condition is met. Since real environments cannot always satisfy the optimum frequency ratio condition, then a natural question is what happens if this condition is not met? TAT in these cases is dependent on the *type of on-chip decoder*. If the on-chip decoder has a serial nature, then the TAT is heavily influenced by changes in the frequency ratio. However, if the decoder has a parallel nature, the influences are minor. The *length of the pattern* determines the pattern distribution and the number of clock cycles the on-chip decoder requires to generate a pattern. Therefore, the *length of the pattern* determines the optimum frequency ratio and, hence, it also influences the TAT.

3.1.2 Analysis of previous approaches in TDCE

In this section three representative previous approaches [121, 122, 123] are analysed from the TDCE parameter's standpoint and it is shown that they have improved some of the parameters at the expense of the others.

(i) Selective coding (SC) [121] This method splits the test vectors into fixed-length input patterns of size *b* (block size), and applies Huffman coding to a carefully selected number of patterns while the rest of the patterns are prefixed. The SC decoder has a parallel nature. Although, due to this parallelism, the on-chip decoder yields good TAT, the use of *fixed-length* input patterns and prefixed codes requires shift registers of length b, which lead to a large area overhead. In addition, *fixed-length* input patterns restrain the method from exploiting '0'-mapped test sets. Hence, special pre-processing algorithms [129, 130] were proposed to increase the compression attainable with SC. However, these algorithms, which target the SC fixed-length input pattern principle, further increase the computational complexity of this method. It should be noted that using SC with T_{diff} allows good compression when the block size is increased. Hence, *due to a fixed-length input pattern* the method is restrained from exploiting '0'-mapped test sets, and the main drawback of this approach is its large area overhead.

(ii) Golomb coding [122] This method assigns a variable-length Golomb code, of group size m_g, to a run of '0's. Golomb codes are composed from two parts, a prefix and a tail, and the tail has the length given by $\log_2 m_g$. Using runs of '0's as input patterns, the method has two advantages: firstly the decoder uses counters of length $\log_2 m_g$ instead of shift registers, thus leading to low area overhead; and secondly it exploits the '0'-mapped test sets improving the compression ratio. Golomb codes are optimum only for a particular pattern distribution, hence the coding scheme yields the best compression only in particular cases. Furthermore, since the method in [122] uses a serial on-chip decoder, the TAT is heavily influenced by changes in the frequency ratio. Therefore, the method's main drawback is its large TAT.

(iii) Frequency-directed run-length (FDR) coding [123] The FDR code is composed of two parts, a prefix and a tail, where both parts have the same length. In order to decompress a FDR code, the on-chip decoder has to identify the two parts. Because the code is not dependent on a group size, unlike the Golomb code, the decoder has to detect the length of the prefix in order to decode the tail. In order to do so, the FDR code requires a more complicated decoder with fixed area overhead. The code being developed to exploit particular pattern distributions common to most test sets yields good compression ratios. However, despite the good compression ratios, having a serial on-chip decoder a large TAT is the main disadvantage of the FDR method.

As illustrated above, current approaches for test data compression, efficiently address some of the test parameters at the expense of the others. Therefore, this chapter proposes a new coding scheme and a new on-chip decoder, which when combined allow simultaneous improvement in the three TDCE parameters: compression ratio, area overhead and test application time, when compared to previous work.

3.2 Compression

As illustrated in Section 3.1.1, the coding scheme is the key component in the TDCE. This section introduces a new Variable-length Input Huffman Coding scheme (Section 3.2.1) and a new compression algorithm which combines the new coding scheme with a mapping and reordering algorithm (Section 3.2.2).

3.2.1 New Variable-length Input Huffman Coding (VIHC) scheme

The proposed coding scheme is based on Huffman coding. Huffman coding is a statistical data-coding method that reduces the average code length used to represent the unique patterns of a set [127]. It is important to note that the previous approach [121] which uses Huffman coding in test data compression employs only patterns of *fixed-length* as input to the Huffman coding algorithm. As outlined in the previous section *fixed-length* input patterns restrict exploitation of the test set features for compression. This problem is overcome by the coding scheme proposed in this section which uses patterns of *variable-length* as input to the Huffman algorithm, allowing an efficient exploitation of test sets which exhibit long runs of ’0’s. This *distinction* influences not only the compression, but it also provides the justification for employing a *parallel decoder* using *counters* (see Section 3.3.1). This will lead not only to significantly lower area overhead, but it will also facilitate TAT reduction when the compression ratio and frequency ratio are increased (see Sections 3.3.2 and 3.4). The following two examples illustrate the proposed coding scheme and highlight some of its properties.

Example 3.1 Figure 3.2 illustrates the proposed coding scheme for a group size of 4 ($m_h = 4$). The group size represents the maximum acceptable number of ’0’s contained in a run of ’0’s. Using the group size, the initial test vector (t_{init}) is divided into runs of ’0’s of length smaller than, or equal to, 4, which are referred to as patterns (Figure 3.2(a)). These patterns are used as input to the Huffman coding scheme, where for each pattern

$m_h = 4$ $\quad |t_{init}| = 26$ bits $\quad |t_{cmp}| = 16$ bits

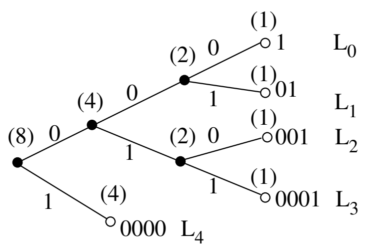
$$t_{init} \quad \underbrace{1}_{000}\ \underbrace{01}_{001}\ \underbrace{0000}_{1}\ \underbrace{0000}_{1}\ \underbrace{0000}_{1}\ \underbrace{0001}_{011}\ \underbrace{0000}_{1}\ \underbrace{001}_{010}$$

t_{cmp} 000 001 1 1 1 011 1 010

(a) Initial test vector

Pattern	Occurrence	Code
L_0 = 1	1	000
L_1 = 01	1	001
L_2 = 001	1	010
L_3 = 0001	1	011
L_4 = 0000	4	1

(b) Dictionary



(c) Huffman tree

Run of 0s	Golomb code	VIHC code
1	000	000
01	001	001
0000 0000 0000 0001	1 1 1 011	1 1 1 011
0000 001	1 010	1 010

(d) Comparison with Golomb for $m_h = 4$

Figure 3.2. VIHC for $m_h = 4$

the number of occurrences is determined. Having obtained the patterns and the number of occurrences, the first two columns of the dictionary are filled (see Figure 3.2(b)). Using these two columns, the Huffman tree[2] is built as explained next (see Figure 3.2(c)). First, the patterns are ordered in the descending order of their occurrences. Then, the two patterns with the lowest number of occurrences (L_0 and L_1) are combined into a new node (L_0, L_1) with the number of occurrences given by the sum of the two patterns' occurrences. This step is repeated until all the nodes are merged into one single node. Initially, patterns L_2 and L_3 are merged into (L_2, L_3), then nodes (L_0, L_1) and (L_2, L_3) are merged together, and finally, node L_4 is merged with node $((L_0, L_1), (L_2, L_3))$. This process is illustrated in Figure 3.2(c), where the occurrences of the patterns are also given between brackets. After the tree is created, the Huffman code is obtained by assigning binary '0's and '1's to each segment starting from the nodes of the Huffman tree (Figure 3.2(c)). The codes are illustrated in the third column in Figure 3.2(b). Using the obtained Huffman code, the initial test vector t_{init} is compressed into t_{cmp}. It should be noted that ordering

[2]For a detailed description of the Huffman algorithm, the reader is referred to [127]

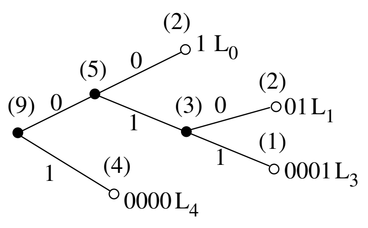
$m_h = 4$ $|t_{init}| = 26$ bits $|t^V_{cmp}| = 17$ bits $|t^G_{cmp}| = 19$ bits

t_{init}	1	01	0000	0000	0000	0001	0000	1	01
t^V_{cmp}	00	011	1	1	1	010	1	00	011
t^G_{cmp}	000	001	1	1	1	011	1	000	001

(a) Test vectors

Pattern	Occurrence	Code
L_0 = 1	2	00
L_1 = 01	2	011
L_3 = 0001	1	010
L_4 = 0000	4	1

(b) Dictionary



(c) Huffman tree

Run of 0s	Golomb code	VIHC code
1	000	00
01	001	011
0000 0000 0000 0001	1 1 1 011	1 1 1 010
0000 1	1 000	1 00

(d) Comparison with Golomb for $m_h = 4$

Figure 3.3. VIHC, the general case

the patterns in the ascending or descending order of their occurrences does not affect the Huffman tree building algorithm. It is important, however, to always combine the patterns with the smallest number of occurrences.

One interesting property, which can be easily observed when the proposed VIHC scheme and the Golomb coding scheme [122] are compared, is that Golomb codes represent a particular case of VIHCs. This is illustrated in Figure 3.2(d), where the codes obtained for the runs of ’0’s from t_{init} in Figure 3.2(a), are the same for both coding schemes. Because of this particularity, the compression ratios of the Golomb coding scheme will be upper bounded by the compression ratios of the VIHC coding scheme for a given group size. This is illustrated in the following example:

Example 3.2 Consider the scenario illustrated in Figure 3.3, where a simple change in the t_{init} from Figure 3.2(a) is performed: the 24th bit is set from 0 to 1. The dictionary and the Huffman tree for this new scenario are shown in Figure 3.3(b) and 3.3(c) respectively. The Golomb and the VIHC codes for this case are given in Figure 3.3(d). When t_{init} is

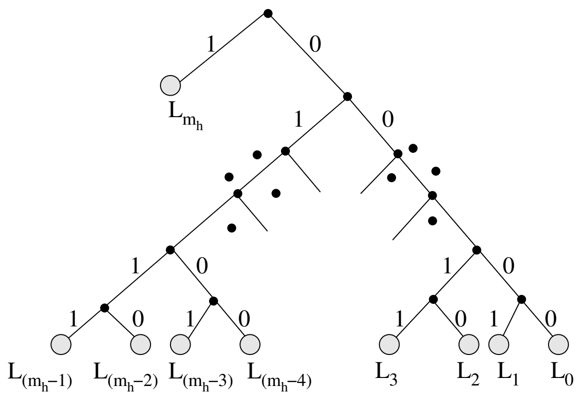


Figure 3.4. Huffman tree when the Golomb code and the VIHC are equal

encoded using Golomb coding, the encoded test vector is 19 bits long (t^G_{cmp}). If the test vector is encoded using the proposed VIHC scheme, then the encoded test set is 17 bits long (t^V_{cmp}), which shows the reduction in volume of test data.

In the following the proposed coding scheme and the properties illustrated using the previous examples are formalised. Let m_h be the group size, and $L = \{L_0, \ldots, L_{m_h}\}$ be a set of unique patterns obtained after the test set was divided into runs of '0's of maximum length m_h. For the computed set of patterns L, the set of number of occurrences $P = \{n_0, \ldots, n_{m_h}\}$ is determined. $n = \sum_{i=0}^{m_h} |L_i| * n_i$ is the size of the test set, where $|L_i|$ denotes the length of pattern L_i. The Huffman codes are obtained using L and P. For a pattern L_i the Huffman code is denoted by c_i, and the length of the code is given by w_i. The minimum codeword length $min\{w_i\}$ is referred to as w_{min}. The size of the compressed test set using a group size of m_h is denoted by $H(m_h) = \sum_{i=0}^{m_h} n_i * w_i$. It should be noted that the terms "Huffman code" and "codeword" will be used interchangeably and the term "group size" is preferred to "block size". It is interesting to note that there are at most $m_h + 1$ patterns and thus $m_h + 1$ leafs in the Huffman tree. There are m_h patterns formed from runs of '0's ending in 1 ($L_0 \ldots L_{m_h-1}$) and one pattern formed from '0's only (L_{m_h}) as illustrated in Figure 3.2(b) for $m_h = 4$.

Example 3.1 illustrated that Golomb coding is a particular case of VIHC. Hence, for a given group size and for a particular pattern distribution the VIHC will reduce to the Golomb code. Consider the following pattern distribution: $(a)\ n_{m_h} \geq n_{i_0} + n_{i_1} + \ldots + n_{i_{\frac{m_h}{2}-1}}, \forall\ distinct\ i_0 \ldots i_{\frac{m_h}{2}-1} < m_h$ and $(b)\ n_k \geq n_i + n_j, \forall i, j, k < m_h, i \neq j \neq k$, where m_h represents the group size and n_i is the number of occurrences of pattern L_i. Based on the above conditions, the Huffman tree illustrated in Figure 3.4 is constructed. Condition

(a) guarantees that the pattern L_{m_h} will be the left-most leaf in the tree (as shown in Figure 3.4). Condition (b) ensures that the Huffman tree construction algorithm will merge all the simple patterns first, while the merged patterns are processed afterwards, i.e., it will ensure that the simple patterns $(L_0 \dots L_{m_h-1})$ are on the last level of the tree as illustrated in Figure 3.4. It should be noted that even if the patterns L_i, with $i < m_h$, appear in a different order than the one illustrated in the figure, then as long as the above conditions are met, swapping and rearranging the nodes in the Huffman tree will lead to the tree presented in Figure 3.4 with no penalties in compression ratio[3] [127]. Using the convention that the left leaf of a node is assigned a '1' and the right one a '0', it is found that $c_{m_h} = 1$ and $c_i = 0$ followed by i represented on $log_2 m_h$ bits. It is clear that if a run of '0's of length lower than m_h is compressed, then the code is the same as Golomb's. If the length of run of '0's (l) is greater than m_h, then the assigned Golomb code is given by $\underbrace{1\dots1}_{\lfloor l/m_h \rfloor}0tail$, where $tail = l - \left\lfloor \frac{l}{m_h} \right\rfloor$ represented on $log_2 m_h$ bits. The same representation is obtained if the code is derived using the proposed VIHC method. Since the proposed method can produce other codes as well, depending on the pattern distribution for a given group size, the Golomb code is a particular case of the proposed coding method.

It is interesting to note that the above observation ensures that the VIHC decoder proposed in Section 3.3.1 can be used to decompress a Golomb compressed test set. Furthermore, if the VIHC coding obtains the same code as Golomb, the minimum code length is 1, or $w_{min} = 1$. This derives from the previous observation and can be easily seen in Figures 3.2(c) and 3.4. Moreover, as illustrated in Figure 3.3, Golomb coding leads to smaller compression ratios in comparison to the proposed VIHC scheme. This is formalised in the following theorem.

Theorem 3.1 *For a given group size, the compression ration obtained by the Golomb coding is lower than, or equal to, the compression ration obtained by VIHC.*

Proof: The above can be easily shown by using the fact that the Huffman code is an optimal code ([127, Theorem 5.8.1]), i.e., any other code will have an expected length greater than the expected length of the Huffman code. Since the Golomb code is a particular case of VIHC (the VIHC in this particular case is referred to as $VIHC_G$), it is optimal

[3] It should be noted that when $n_k = n_i + n_j$ in condition (b), the Huffman algorithm may construct a tree different to the one shown in Figure 3.4. However, by rearranging the leafs, the tree in Figure 3.4 can be obtained without any penalties in compression ratio [127]

only for a particular pattern distribution. For other pattern distributions the $VIHC_G$ code is not optimal, thus the expected length of the $VIHC_G$ code is greater. Based on this fact, and using the same reasoning as in [127, Theorem 5.8.1] the proof of the theorem is immediate.

3.2.2 Compression algorithm based on VIHC

This section presents the compression algorithm which employs the VIHC scheme for compressing the test set. In this work, both the compression of the difference vector sequence (T_{diff}) and the compression of the initial test sequence (T_D) is taken into account. The initial test set (T_D) is partially specified and the test vectors can be reordered. This is because automated test pattern generator (ATPG) tools specify only a small number of bits in every test vector, thus allowing for greater flexibility during *mapping*. In addition, since full scan circuits are targeted, *reordering* of test vectors is also allowed. The mapping and reordering occurs in a pre-processing step, which prepares the test set for compression by mapping the “don’t cares” to ’0’s or ’1’s and reordering the test set such that longer runs of ’0’s are generated. This is, then, exploited by coding schemes based on runs of ’0’s leading to increased compression ratio (as illustrated in Section 3.4). The compression algorithm has three main procedures: $i)$ prepare initial test set, $ii)$ compute Huffman code and $iii)$ generate decoder information. The first two procedures are used for compression only and the last one determines the decoding architecture described in Section 3.3. The compression algorithm is illustrated in Algorithm 3.1. The inputs to the algorithm are the initial test set (T_D) and the type of performed compression (*compression_type*), which can be either *diff*, if the reordered difference vector sequence is compressed, or *var*, if the reordered initial test set is compressed. It is important to note that the MinTest dynamic compacted test sets [131] used in this thesis have on average 27% specified bits, with up to 100% specified bits per test vector.

$i)$ **Prepare initial test set** As illustrated in the previous section the VIHC method uses runs of ’0’s as input patterns. The length of these patterns can be increased using this pre-processing procedure. The procedure consists of two steps. In the first step, for the compression of T_{diff} the “don’t cares” are mapped to the value of the previous vector on the same position; for the compression of T_D, the “don’t cares” are mapped to ’0’s (*Set-*

Algorithm 3.1 VIHC compression

```
CompressTestSet($T_D$, compression_type)
begin
i). Prepare initial test set
1.    Mapping
          SetDefaultValues($T_D$,compression_type)
2.    Reordering
          $T_D^R$={$t_{min}$}, Remove($t_{min}$,$T_D$), previous_solution=$t_{min}$
          while not $T_D$ empty
            $t_{min}$ = DetermineNextVector($T_D$,compression_type)
            if compression_type <> diff then $T_D^R$=$T_D^R \cup t_{min}$
            else $T_D^R$=$T_D^R \cup xor(t_{min}, previous\_solution)$
            Remove($t_{min}$,$T_D$), previous_solution=$t_{min}$
ii). Huffman code computation
      $(L,P)$=ConstructDictionary($T_D^R$)
      $H_C$=GenHuffmanCode($L$,$P$)
iii). Generate decoder information
      $B_C$=AssociateBinaryCode($H_C$,$m_h$)
end
```

DefaultValues in Algorithm 3.1). In the second step the test set is reordered and denoted by T_D^R in the algorithm. The reordering process is illustrated in the algorithm in step *2*. Starting with the test vector which has the minimum number of ’1’s, the next test vector t_{min} is determined (procedure *DetermineNextVector* in the algorithm) such that the following conditions are met. For T_{diff}, the next test vector is selected such that the number of ’1’s in the difference between the test vector and the *previous_solution* is minimum, and the length of the run of ’0’s obtained by concatenating the two vectors is maximum (i.e., if there are more test vectors which lead to the same number of ’1’s in the difference with *previous_solution*, then the one which has the longer run of ’0’s, when the two vectors are concatenated, is selected). For T_D, the next test vector is selected such that the length of the run of ’0’s obtained by concatenating the two vectors is maximum. Further on, if the *compression_type* is *diff*, the difference between t_{min} and the previous solution (*previous_solution*) is added to the reordered test set (T_D^R), otherwise, the selected test vector (t_{min}) is added to T_D^R. The reordering algorithm has a complexity of $O(|T_D|^2)$, where $|T_D|$ represents the number of test vectors in the test set. The mapping and reordering performed in this procedure will be exploited by the *variable-length* input patterns used for Huffman code computation in the following procedure.

m_h	H_C	B_C
4	000	(001,0)
	001	(010,0)
	010	(011,0)
	011	(100,0)
	1	(100,1)

Table 3.1. Core user/vendor information exchange

ii) **Huffman code computation** Based on the chosen group size (m_h) the dictionary of variable-length input patterns (L) and the number of occurrences (P) are determined from the reordered test set (*ConstructDictionary* in Algorithm 3.1). This is a mandatory step for VIHC because, in contrast to Golomb which has the codewords precomputed for a given group size, the codewords are determined based on the group size and the pattern distribution. Using L and P the Huffman codes (H_C) are computed using the Huffman algorithm [127] (*GenHuffmanCode* in the algorithm). Having determined the Huffman codes, the last procedure generates the information required to construct the decoder.

iii) **Generate decoder information** This procedure is performed by *AssociateBinaryCode* in Algorithm 3.1. For each Huffman code c_i a binary code b_i is assigned. The binary code is composed from the *length* of the initial pattern on $\lceil log_2(m_h+1)\rceil$ bits and a *special* bit which differentiates the cases when the initial pattern is composed of '0's only, or it is a run of '0's ending in 1. Thus, $b_i = (|L_i|, 0)$ for $i < m_h$, and $b_{m_h} = (|L_{m_h}|, 1)$ ($|L_i|$ denotes the length of pattern L_i, see Section 3.2.1). It should be noted that if the core provider supplies a test set compressed with the VIHC method, then the group size, the Huffman codes and the corresponding binary codes are the only information required to generate the on-chip decoder. For the example in Figure 3.2, Table 3.1 illustrates the required information. The following section shows how the binary code is used by the proposed *VIHC decoder*.

3.3 Decompression

This section introduces the new on-chip VIHC decoder (Section 3.3.1) for the VIHC scheme and provides the TAT analysis for the proposed decoder (Section 3.3.2). For the remainder of this chapter a generic decompression architecture as shown in Figure 3.5 is

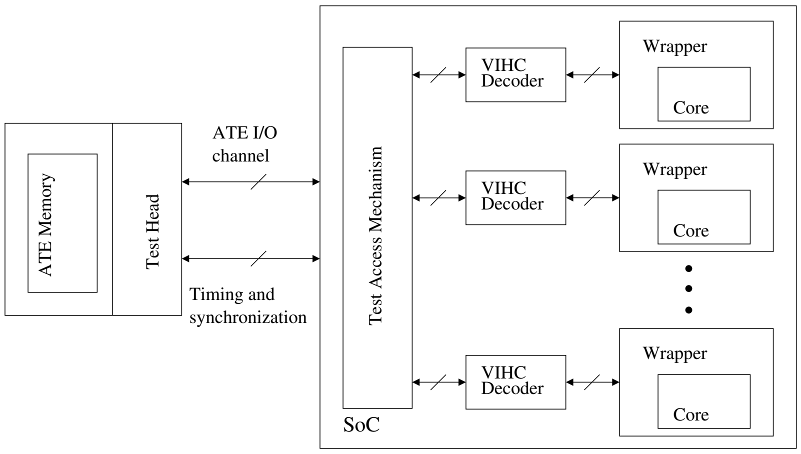


Figure 3.5. VIHC generic decompression architecture based on [122]

assumed. The VIHC decoder, proposed in Section 3.3.1, uses a novel parallel approach contrary to the Golomb [122] serial decoder. It should be noted that for the decompression of T_{diff}, a CSR architecture [119, 122] is used after the VIHC decoder in Figure 3.5. This work assumes that the ATE is capable of external clock synchronisation [65].

3.3.1 VIHC decoder

A block diagram of the VIHC decoder is given in Figure 3.6. The decoder comprises a *Huffman decoder* [118] (Huff-decoder) and a *Control and Generator Unit* (CGU). The Huff-decoder is a finite state machine (FSM) which detects a Huffman code and outputs the corresponding binary code. The CGU is responsible for controlling the data transfer between the ATE and the Huff-decoder, generates the initial pattern and controls the scan clock for the CUT. The *data in* line is the input from the ATE synchronous with the external clock (*ATE clock*). When the Huff-decoder detects a codeword, the *code* line is high and the binary code is output on the *data* lines. The *special* input to the CGU is used to differentiate between the two types of patterns, composed out of ’0’s only (L_{m_h}) and runs of ’0’s ending in 1 ($L_0 \dots L_{m_h-1}$). After loading the code, the CGU generates the pattern (*data out*) and the internal *scan clock* for the CUT. If the decoding unit generates a new code while the CGU is busy processing the current one, the *ATE sync* line is low notifying the ATE to stop sending data and the *sync FSM clk* is disabled forcing the Huff-decoder to maintain its current state. Dividing the VIHC decoder in Huff-decoder and CGU, allows the Huff-decoder to continue loading the next codeword while the CGU generates the current pattern. Thus, the Huff-decoder is interrupted *only if necessary*,

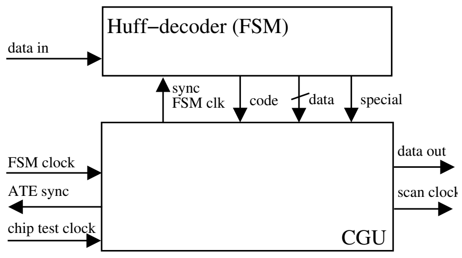


Figure 3.6. VIHC decoder

which is in contrast to the Golomb [122] and the FDR [123] serial decoders. This leads to a low TAT in comparison to the Golomb and the FDR methods, as will be shown in Section 3.4. It should be noted that if the ATE responds to the stop signal with a given latency (i.e., it requires a number of clock cycles before the data stream is stopped or started), the device interface board between the ATE and the system will have to account for this latency using a first in first out (FIFO) - like structure. This issue will be detailed in Chapter 4. The *FSM clock* signal ensures that the FSM will reach a stable state after one internal clock cycle, therefore it is generated as an one internal clock cycle for each external clock cycle period. This can be achieved by using the same technique as in [65] for the *serial scan enable* signal. An additional advantage of the proposed on-chip parallel decoder is having the two components working in two different clock domains (the Huff-decoder changes state with the ATE operating frequencies, and the CGU generates data at the on-chip test frequency). This will leverage the frequency ratio between the on-chip test frequency and the ATE operating frequency, hence, facilitating data delivery to the CUT at the on-chip test frequency using only *one* ATE channel.

The FSM for the Huffman decoder corresponding to the example from Figure 3.2 is illustrated in Figure 3.7. Starting from state S_1, depending on the value of *data in* the Huff-decoder changes its state. It is important to note the following:

- after the Huff-decoder detects a codeword it goes back to state $S1$; for example, if the *data in* stream is “001”, the Huff-decoder first changes its state from $S1$ to $S2$, then from $S2$ to $S3$ after which back to $S1$, and sets *data* and *special* to the corresponding binary code ($(010,0)$ in this case), and the *code* line high;
- the number of ATE clock cycles needed to detect a code is equal to the length of the code; for example, if the *data in* stream is “001”, the Huff-decoder needs three ATE clock cycles to identify the code;
- the Huff-decoder has a maximum of m_h states for a group size of m_h; the number of states in a Huff-decoder is given by the number of leafs in the Huffman tree minus

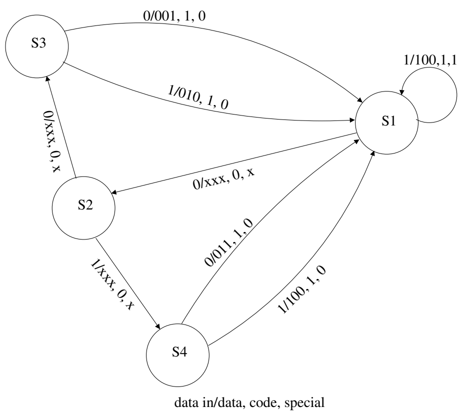
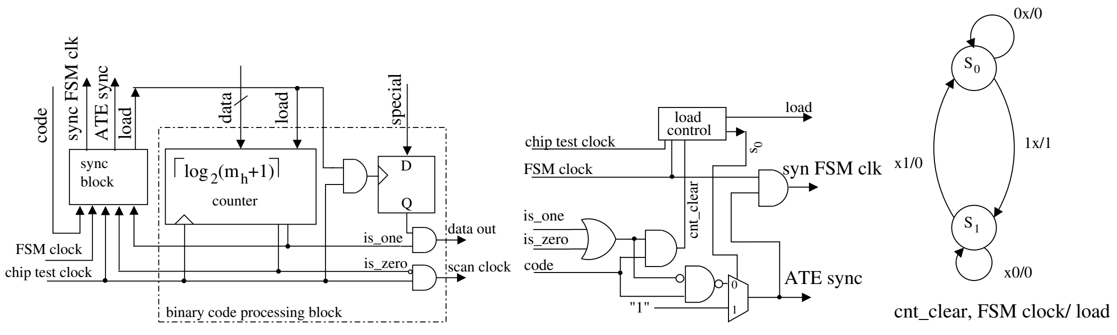


Figure 3.7. FSM for example in Figure 3.2



(a) CGU (b) Synchronisation block (c) Load control FSM

Figure 3.8. CGU for VIHC decoder

one; since there are a maximum of m_h+1 leafs for a group size of m_h, the maximum number of states is m_h.

A high level view of the CGU is given in Figure 3.8(a). There are two main components of the CGU: the binary code processing block and the synchronisation block. The binary code processing block comprises a counter which will hold the *length* of the initial pattern, and a flip flop which holds the *special* bit (see the binary code representation in Section 3.2.2). Since it is a parallel decoder there are two issues to be considered. Firstly, the binary code processing block should be loaded only when a new code is identified; and secondly, the load of new data is done only when the pattern corresponding to the already loaded binary code was fully processed. These tasks are controlled by the synchronisation block (*sync*) illustrated in Figure 3.8(b). As noted earlier, when the Huff-decoder identifies a code, it will output the associated binary code and set the *code* line high. When the *code* line is high, the *sync* block will determine the values of *sync FSM clk*, *load* and

ATE sync. The key element in the synchronisation block is the *load control* FSM, the state diagram of which is detailed in Figure 3.8(c). The inputs into the load control FSM are the *cnt_clear* and the *FSM clock*. The *cnt_clear* will notify the FSM when the current code has been processed (i.e., the counter's value is either zero or one) and there is a new code available (i.e., the code line is high). The *FSM clock* is used to synchronise the load control FSM with the Huff-decoder FSM. When *cnt_clear* is 1, the FSM will change its state from S_0 to S_1 and set *load* to 1. After one clock cycle, the FSM will set *load* to 0. If *FSM clock* is 1, the FSM will return to state S_0, otherwise it will remain in state S_1. This last condition was added since a new code can only occur after the *FSM clock* was 1 (this is how the Huff-decoder is controlled). When the FSM is in state S_0, the S_0 line in Figure 3.8(b) together with the logic driving the multiplexer will ensure that the Huff-decoder continues the load of data from the ATE once the data has been loaded into the binary code processing block. Hence, the stream of data from the ATE to the Huff-decoder is stopped only when necessary. The load control FSM can be implemented using one FF and additional logic. A detailed implementation of the proposed decoder is given in Appendix B.

When compared to the SC decoder [121], which has an internal buffer and a shift register of size b (the group size), and a Huffman FSM of at least b states, the proposed decoder has only a $\lceil log_2(m_h+1)\rceil$ counter, two extra latches and a Huffman FSM of at most m_h states. Thus, for the same group size ($m_h = b$) significant reduction in area overhead is obtained, as will be seen in Section 3.4. Similar to SC [121], the synchronisation channel to the ATE can be eliminated if the VIHC codes are manipulated at the expense of reduced compression ratio. In addition, both VIHC and SC can eliminate the synchronisation channel if the frequency ratio is greater or equal to their corresponding optimum frequency ratio. These issues are detailed in Chapter 4. For VIHC this is illustrated in the following section.

3.3.2 Test application time analysis

Because the on-chip decoder has two components working in two different clock domains (i.e., the Huff-decoder is receiving data with ATE operating frequencies and the CGU generating data at the on-chip test frequencies), the TAT is influenced by the ratio between the two frequencies: the on-chip test frequencies and the ATE operating frequencies. To

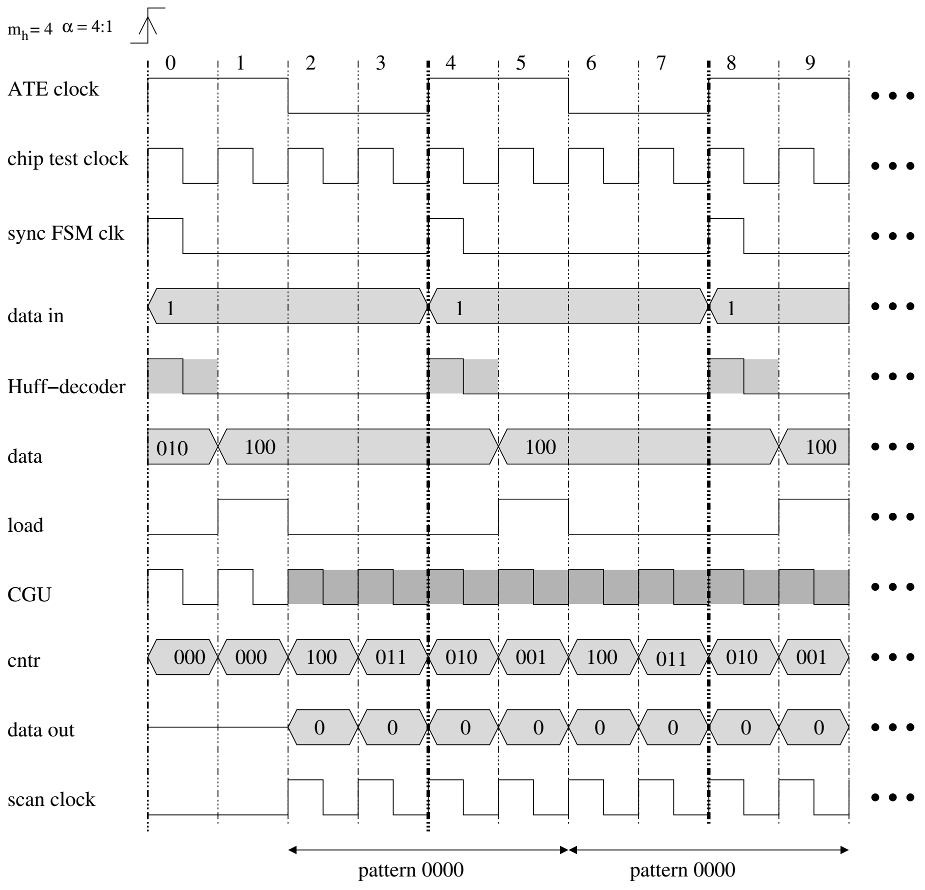


Figure 3.9. VIHC decoder timing diagram

understand the effects of the frequency ratio on the TAT, this section provides a TAT analysis with respect to the frequency ratio. It is considered that the data is fed into the Huff-decoder FSM with the ATE operating frequency and that the FSM reaches a stable state after one on-chip test clock cycle. Analysing the FSM for the Huffman decoder it can be observed that the number of ATE clocks needed to identify a codeword c_i is equal to the size of the codeword w_i (see Section 3.2.1). On the other hand, the number of internal clock cycles needed to generate a pattern is equal to the size of the pattern.

In order to illustrate the functionality of the CGU unit and to provide an example for the TAT analysis described next, Figure 3.9 shows the timing diagram for a frequency ratio of $\alpha = 4$, considering $m_h = 4$. The diagram corresponds to the generation process of the first two “0000” patterns in Figure 3.2 (see Example 3.1 in Section 3.2.1). This case was chosen to illustrate the parallelism of the proposed VIHC decoder and the overlap between generating the last bit of the pattern and loading the next binary code. A gate level timing simulation for Example 3.1 is given in Appendix B. The *ATE clock*, the *chip test clock* at a ratio of 4 : 1 with respect to the *ATE clock*, and the *sync FSM clk* required to drive the *Huff-decoder* are shown in the upper part of Figure 3.9. The *data in* row illustrates the data send from the ATE to the VIHC decoder. As detailed in the previous section, the Huff-decoder reaches a stable state after one internal clock cycle. Hence, the *data* signals which are loaded into the CGU unit are valid after one internal clock cycle. The time intervals in which the Huff-decoder identifies the last bit of a codeword and outputs the binary code (see Section 3.3.1) are highlighted in the *Huff-decoder* row with dashed areas. With the *data* signals valid, the CGU sets the *load* signal high and loads *100* into the counter (*cntr* row in Figure 3.9 clock cycle 2). For the next four *chip test clock* cycles, the CGU is busy generating pattern "0000" as illustrated in the figure with the dashed areas in row *CGU*. The *cntr* is decremented and the *data-out* outputs the patterns' bits. While *cntr* is not zero, the *scan clk* is generated. At clock cycle 4, the *Huff-decoder* will identify the next codeword. Again, the *load* signal is high and the *data* signals are loaded into the counter. It is important to note that this occurs simultaneously with the output of the last bit from the previous pattern. Hence, between two consecutive counter loads, for $\alpha = 4$, there are effectively 4 clock cycles in which data can be generated.

Formally, if $\alpha = \frac{f_{chip}}{f_{ate}}$ is the ratio between the on-chip test frequency (f_{chip}) and the ATE operating frequency (f_{ate}), then after a Huffman code is identified, $\alpha - 1$ internal clock cycles from the current ATE cycle can be used to generate a pattern. Thus, in order for the Huff-decoder to run without being stopped by the CGU, the CGU must be able to generate the pattern L_i in the number of internal clock cycles remaining from the ATE clock in which it was started plus the number of internal clock cycles needed for the Huff-decoder to identify the next codeword. Or, $|L_i| < (\alpha - 1) + 1 + \alpha * (w_j - 1)$, where w_j is the length of the next codeword in bits. With $max(|L_i|) = m_h$ and $min\{w_i\} = w_{min}$, the frequency ratio to obtain the smallest TAT is given by $\alpha_{max} = \frac{m_h}{w_{min}}$ (*optimum frequency ratio*). The lowest TAT is given by $H(m_h) + \delta$, where $H(m_h)$ is the number of bits in the compressed test set in ATE clock cycles and δ is the number of extra ATE clocks needed to decode the last codeword. When the $\alpha \geq \alpha_{max}$ an increase in compression ratio will lead to a reduction in TAT.

To compute the TAT for the compressed test set for frequency ratios smaller than the optimum frequency ratio, an approximation function with respect to $\alpha = \frac{f_{chip}}{f_{ate}}$ is given next. The function to compute the TAT for the VIHC decoder in Section 3.3.1 is given by

$$\tau(\alpha) = H(m_h) + \delta + \sum_{i=\lceil w_{min}*\alpha \rceil}^{m_h} n_i * \left\lceil \frac{|L_i| - w_{min} * \alpha}{\alpha} \right\rceil \qquad (3.1)$$

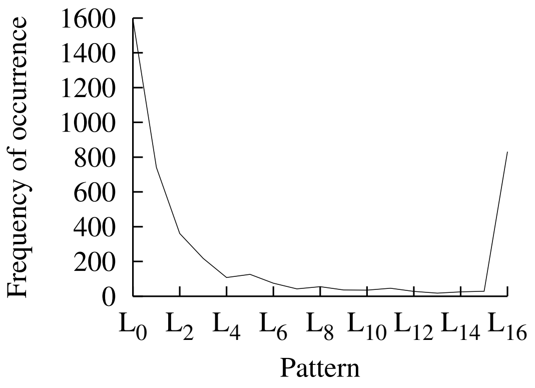


Figure 3.10. Pattern distribution for s5378 with $m_h = 16$

when each pattern is followed by the codeword with the smallest decoding time (w_{min} in ATE clock cycles). In order to give an approximation to the above function the pattern distribution is analysed. For example, for the full scan version of the s5378 ISCAS89 benchmark circuit, Figure 3.10 illustrates the pattern distribution for $m_h = 16$ for the MinTest [131] test set. It can be observed that the patterns with length smaller than 4 and greater than $(m_h - 1)$ are the most frequent. Therefore the function can be approximated with

$$\tau(\alpha) \approx H(m_h) + \delta + n_o * \left\lceil \frac{m_h - w_{min} * \alpha}{\alpha} \right\rceil \tag{3.2}$$

where n_o is the number of patterns with length m_h (the patterns L_{m_h-1} and L_{m_h}). It should be noted that δ will be ignored since $\delta \ll H(m_h)$. In the worst case scenario, for $n_o = \left\lceil \frac{n}{m_h+1} \right\rceil$ (each run of length m_h is assumed to be followed by a run of length 1)

$$\tau(\alpha) \approx H(m_h) + \frac{n}{m_h + 1} * \frac{m_h - w_{min} * \alpha}{\alpha} \tag{3.3}$$

To show that formula (3.3) provides an approximation for the TAT of the decoder, in Figure 3.11 a comparison between the TAT obtained using the above formula and the TAT obtained by simulating the compressed test set for circuit s5378 using a group size of 16 is illustrated. The TAT reported in the figure is in ATE clock cycles. As it can be observed, the difference between the two curves is small. Note that the TAT is minimum for a frequency ratio of 8 since the minimum code length is 2 ($w_{min} = 2$) in this case. In addition, for frequency ratios greater than α_{max} the TAT equals $\tau(\alpha_{max})$ and the synchronisation channel which notifies the ATE to start/stop the data stream is no longer necessary.

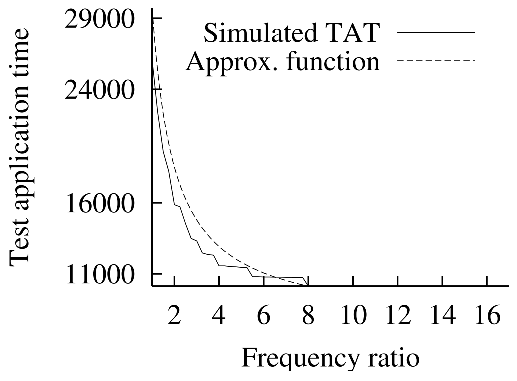


Figure 3.11. TAT for s5378 with $m_h = 16$

3.4 Experimental results

To validate the efficiency of the proposed method, experiments have been performed on the full-scan version of the largest ISCAS89 benchmark circuits [132]. The test sets used in these experiments have been obtained using MinTest [131] dynamic compaction (also used in [122]). The experiments have been performed on a Pentium III 500MHz workstation with 128 MB DRAM. Details about the tools and the benchmark circuits used in the experiments are given in Appendix F. SC [121], Golomb [122], FDR [123] and the proposed VIHC method have been implemented in C++. Firstly, the proposed compression method is analysed from the compression ratio's point of view. Secondly, an area overhead comparison between the four on-chip decoders (SC [121], Golomb [122], FDR [123] and the proposed VIHC method) is given. Finally, the TATs obtained by simulating the four methods' on-chip decoders are compared. The experiments have been performed for both the difference vector sequence (T_{diff}) and the initial test set (T_D). Since SC [121] did not report compression ratios for the test sets used in this chapter the results reported in this section have been computed by implementing the SC method and applying it to the MinTest [131] test sets. It should be noted that SC was applied on the same test set as VIHC after the mapping and reordering algorithm proposed in this chapter (see Section 3.2.2). In addition, the number of patterns considered for selective coding is given by the group size plus one, i.e., for a group size of $m_h = 4$, 5 patterns have been considered. This is motivated by the pattern distribution obtained with the considered mapping and reordering algorithm, which for $m_h = 4$ usually leads to the patterns: “0000”, “0001”, “0010”, “0100” and “1000”, as being the most frequent.

Circuit	Compression ratio							Size of $H(Max)$	Size of T_{diff}	Size of MinTest [94]
	Group size									
	4	6	8	12	14	16	Max			
s5378	51.52	54.46	55.23	55.85	56.66	*57.33*	**60.73**	9328	23754	20758
s9234	54.84	58.12	58.75	58.45	58.76	*59.02*	**60.96**	15332	39273	25935
s13207	69.02	75.90	79.07	82.03	82.69	*83.21*	**86.83**	21758	165200	163100
s15850	60.69	65.67	67.48	68.65	68.86	*68.99*	**72.34**	21291	76986	57434
s35932	40.35	49.92	56.97	61.08	62.54	*66.47*	**71.91**	7924	28208	19393
s38417	54.51	58.57	59.96	60.86	61.22	*61.98*	**66.38**	55387	164736	113152
s38584	56.97	61.21	62.50	*63.01*	63.00	62.97	**66.29**	67114	199104	104111

Table 3.2. VIHC for T_{diff}

Circuit	Compression ratio							Size of $H(Max)$	Size of T_D	Size of MinTest [94]
	Group size									
	4	6	8	12	14	16	Max			
s5378	41.13	42.15	42.85	44.85	45.73	*46.94*	**51.78**	11453	23754	20758
s9234	45.27	*46.14*	45.27	46.14	46.02	46.14	**47.25**	20716	39273	25935
s13207	67.60	74.12	76.92	79.49	80.03	*80.36*	**83.51**	27248	165200	163100
s15850	58.01	62.43	63.73	*64.42*	64.28	64.06	**67.94**	24683	76986	57434
s35932	29.76	38.73	44.24	47.07	47.89	*51.84*	**56.08**	12390	28208	19393
s38417	37.19	40.45	43.78	46.97	47.34	*47.79*	**53.36**	76832	164736	113152
s38584	54.43	57.89	58.81	58.90	59.17	*59.62*	**62.28**	75096	199104	104111

Table 3.3. VIHC for T_D

Compression ratio Prior to providing a comparison with previous approaches the performances of the proposed method are analysed. The experiments performed using the T_{diff} and T_D test sets are summarised in Tables 3.2 and 3.3 respectively. For each ISCAS89 benchmark circuit, the tables list the compression ratio obtained for different group sizes (columns 2 to 8), the maximum obtained compression (*Max*), the size of the maximum compressed test set ($H(Max)$), the size of the initial test set in bits, and the size of the fully compacted MinTest [94] test set. It should be noted that since T_{diff} is derived from T_D by making the difference between consecutive test vectors, it has the same size. However, as will be seen next, since the content of the two test sets differ, the compression ratio differs as well. As it can be observed in the tables, the maximum compression ratio is up to *86.83* for circuit s13207 in Table 3.2, and up to *83.51* for circuit s13207 in Table 3.3. Analysing the compression ratios vs. group sizes in the tables, it can be observed that the compression ratio tends to increase with the increase in group size. This is illustrated in Figure 3.12 for circuit s5378. However, after a group size of 16, the increase in compression ratio up to the maximum compression ratio is less than 3%. It is interesting to note that this is contrary to the Golomb [122] compression ratio vs. group size trend that tends to have a good compression ratio for one group size, after which by

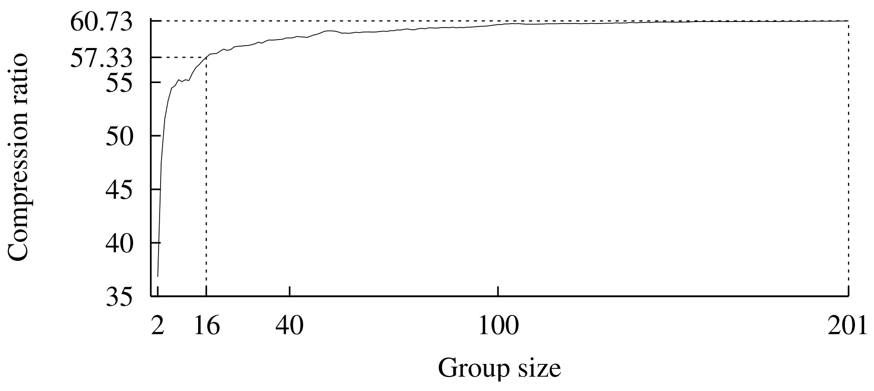


Figure 3.12. Compression ratio vs. group size for s5378 with T_{diff}

Circuit	SC [121]	Golomb [122]	Golomb T_D^R	Golomb Δ%	FDR [123]	FDR T_D^R	FDR Δ%	VIHC
s5378	52.33	40.70	*51.00*	10.30	48.19	*59.00*	10.81	**60.73**
s9234	52.63	43.34	*58.08*	14.74	44.88	*58.85*	13.97	**60.96**
s13207	77.73	74.78	*82.34*	7.56	78.67	*85.50*	6.83	**86.83**
s15850	63.49	47.11	*66.58*	19.47	52.87	*71.02*	18.15	**72.34**
s35932	65.72	N/A	*23.28*	23.28	10.19	*49.78*	29.59	**71.91**
s38417	57.26	44.12	*56.72*	12.60	54.53	*64.32*	9.79	**66.38**
s38584	58.74	47.71	*61.20*	13.49	52.85	*65.27*	12.42	**66.29**

Table 3.4. Best compression ratio comparison for T_{diff}

increasing the group size, the compression ratio decreases. This can be explained by the fact that the particular pattern distribution for which Golomb leads to an optimum code, and hence to good compression, is generally not respected (see Section 3.2.1).

A comparison between the four compression methods (SC [121], Golomb [122], FDR [123] and the proposed VIHC method) is given next. The comparison is summarised in Tables 3.4 and 3.5 for the T_{diff} and T_D test sets respectively. It can be observed that except for circuit s35932 in the case of T_D (see Table 3.5 on the following page), the proposed method obtains constant improvement in compression ratio. For example, for circuit s38417, in the case of T_{diff} (see Table 3.4), the proposed method obtains an increase in compression ratio of 9%, 22% and 13% over SC [121], Golomb [122] and FDR [123], respectively. Similarly, for circuit s38417 in the case of T_D, the increase in compression ratio is 8%, 15% and 10% over SC [121], Golomb [122] and FDR [123], respectively. It should be noted that in the case of circuit s35932, for the T_D test set, the SC [121] obtains better compression due to the regular nature of the test set, however, this is an isolated case as it can be seen in the table. To illustrate the improvement in compression ratio when using the proposed reordering algorithm over the one proposed in [122], the table

Circuit	SC	Golomb			FDR			VIHC
	[121]	[122]	T_D^R	Δ%	[123]	T_D^R	Δ%	
s5378	43.64	37.11	37.13	0.02	48.02	48.03	0.01	**51.78**
s9234	40.04	45.25	45.27	0.00	43.59	43.53	-0.06	**47.25**
s13207	74.43	79.74	79.75	0.01	81.30	81.30	0.00	**83.51**
s15850	58.84	62.82	62.83	0.01	66.22	66.23	0.01	**67.94**
s35932	**64.64**	N/A	N/A	N/A	19.37	19.36	-0.01	56.08
s38417	45.15	28.37	28.38	0.01	43.26	43.26	0.00	**53.36**
s38584	55.24	57.17	57.17	0.00	60.91	60.92	0.01	**62.28**

Table 3.5. Best compression ratio comparison for T_D

also illustrates the compression ratios obtained when applying Golomb and FDR on the test sets mapped and reordered with the algorithm given in Section 3.2.2. These results are given in the table under the T_D^R heading for Golomb and FDR in the case of T_{diff} (see Table 3.4) and T_D (see Table 3.5) respectively. As it can be seen, in the case of T_{diff} the increase in compression ratio when compared to the results reported in [122] and [123] for the two methods is considerable (see column Δ% in Table 3.4). For example, for circuit s35932, 23.28% and 39.59% improvement in compression ratio is obtained. In the case of T_D, the increase in compression ratio is small or negligible (see column Δ% in Table 3.5). This is because, on average, the distribution of the runs of '0's in the case of T_D is dependent mainly on the mapping, while the reordering has smaller influence. In order to provide a fair comparison, the results obtained using the proposed mapping and reordering algorithm will be used for the remainder of this chapter. Since, as illustrated in Figure 3.12, the increase in compression ratio for group sizes larger than 16 is small (less than 3% on average), the maximum group size for all the remaining comparisons is considered 16.

To show that the proposed method exploits better the test set for a given group size, a comparison between SC, Golomb and VIHC for the group sizes for which SC and Golomb obtain the best compression ratio is given next. The maximum group size for all comparisons is 16. The comparison between SC and VIHC is given in Table 3.6 on the next page and the comparison between Golomb and VIHC is given in Table 3.7 on the following page. Both tables list the group size and the compression ratio for which the corresponding method obtained best compression for the T_{diff} and T_D test sets. Analysing Table 3.6, it can be found that, for the T_{diff} test set, the proposed compression method (VIHC) always obtains better compression ratios than SC. For the T_D test set, with the exception of the s35932 circuit, the VIHC compression method always obtains better compression ratios. A similar behaviour can be observed in Table 3.7, where the pro-

Circuit	T_{diff} Group size	T_{diff} SC [121]	T_{diff} VIHC	T_D Group size	T_D SC [121]	T_D VIHC
s5378	12	52.33	**55.85**	12	43.64	**44.85**
s9234	10	52.63	**58.79**	8	40.04	**45.27**
s13207	16	77.73	**83.21**	16	74.43	**80.36**
s15850	16	63.49	**68.99**	16	58.84	**64.04**
s35932	16	65.72	**66.47**	16	**64.64**	51.84
s38417	12	57.26	**60.86**	10	45.15	**45.25**
s38584	14	58.74	**63.00**	12	55.24	**58.90**

Table 3.6. Comparison between SC and VIHC

Circuit	T_{diff} Group size	T_{diff} Golomb T_D^R	T_{diff} VIHC	T_D Group size	T_D Golomb T_D^R	T_D VIHC
s5378	8	51.00	**55.23**	4	37.13	**41.13**
s9234	8	58.08	**58.75**	4	45.27	**45.27**
s13207	16	82.34	**83.21**	16	79.75	**80.36**
s15850	8	66.58	**67.48**	8	62.83	**63.73**
s35932	4	23.28	**40.35**	4	N/A	**29.76**
s38417	8	56.72	**59.96**	4	28.38	**37.19**
s38584	8	61.20	**62.50**	8	57.17	**58.81**

Table 3.7. Comparison between Golomb and VIHC

posed VIHC compression method always obtains better compression ratios than Golomb [122]. The improvements are up to 20% for T_D in the case of s35932, and up to 29% for T_{diff} for the same circuit. These comparisons have illustrated that for a given group size the proposed compression method yields better compression ratios, and hence it better exploits the test set than the previous approaches. It should be noted that in the case of SC, better results might be obtained at the cost of extra computational complexity if other mapping algorithms are used. An area overhead comparison between the proposed on-chip decoder and the three previously proposed on-chip decoders (SC [121], Golomb [122], FDR [123]) is given next.

Area overhead The four on-chip decoders have been synthesised using the Synopsys Design Compiler [133]. The results are summarised in Table 3.8. For all methods, the entire decoder including buffers, shift registers and counters was synthesised. For SC, VIHC and Golomb, the area overhead was computed for group sizes of 4, 8 and 16. Without loss of generality the decoders for s5378 have been synthesised. As shown in Table 3.8, the area overhead for SC is **up to 3 times greater** than the area overhead for VIHC. The area overhead for Golomb is almost equal to the one of VIHC, and the area

Compression Method	Area overhead in tu*		
	Group size		
	4	8	16
SC [121]	349	587	900
Golomb [122]	**125**	227	307
FDR [123]	320		
VIHC	136	**201**	**296**

* technology units for the lsi10k library (Synopsys Design Compiler)
1tu is equivalent to the area of an inverter in lsi10k

Table 3.8. Area overhead comparison for s5378

overhead for FDR is constantly greater than the area overhead for VIHC. It is important to note that even though the Golomb decoder [122] has lower area overhead for a group size of 4, when compared to the proposed decoder, by increasing the group size the area overhead of the Golomb's decoder is greater than that of the proposed decoder. This is because, in order to decode the tail part of the code, the Golomb decoder implements the behaviour of a counter within the Golomb FSM. It should be noted that for the case when the T_{diff} test set is used, a CSR architecture is required. Since the CSR must be of the length of the scan chain fed by the decoder, the CSR overhead is dominated by the length of the internal scan chain of the targeted core (e.g., for core s5378 the CSR comprises 179 scan cells), and it is independent of the chosen compression method. Hence, a CSR is needed by all four approaches and therefore not considered in the area overhead comparison given in Table 3.8. In addition, it has been shown in [119] how the internal scan chains of neighbouring cores in a SOC can be used to eliminate the need for the CSR. The comparisons provided until now show that the proposed method improves on the first two TDCE parameters: compression ratio and area overhead. The last parameter, the TAT, is compared with previous work next.

Test application time To provide an accurate TAT comparison between the four methods the following experimental setup has been considered: the maximum frequency ratio is $\alpha = 8$ and the maximum acceptable group size is 16. To compute the TAT, a simulator was implemented based on the TAT analysis for SC [121], Golomb [122], FDR [123] and the VIHC decoder (Section 3.3.1). For all decoders it was assumed that the data is fed into the decoder at ATE operating frequency and the internal FSM reaches a stable state after one internal clock cycle. In order to provide a fair comparison, firstly the test sets have been mapped and reordered, then the compression methods have been applied. Using these compressed test sets the simulations for each method were performed. The TATs resulting after the simulation are reported in Table 3.9. Analysing columns 3 to 6 and 7 to

Circuit	Comp. Method	TAT (ATE clock cycles) for T_{diff}				TAT (ATE clock cycles) for T_D			
		$\alpha = 2$	$\alpha = 4$	$\alpha = 6$	$\alpha = 8$	$\alpha = 2$	$\alpha = 4$	$\alpha = 6$	$\alpha = 8$
s5378	SC [121]	**15835**	12412	11323	11323	**17259**	14323	13387	13387
	Golomb [122]	20649	16020	13782	13782	23529	19232	14935	14935
	FDR [123]	22263	14678	12968	11679	24933	16803	15259	14039
	VIHC	15868	**11569**	**10777**	**10137**	17668	**13740**	**12914**	**12782**
s9234	SC [121]	**24815**	20675	18605	18605	27459	**23547**	23547	23547
	Golomb [122]	31186	23555	19894	19894	35580	28537	21494	21494
	FDR [123]	36135	24128	21381	19001	42066	29206	26675	24086
	VIHC	24895	**17994**	**16905**	**16095**	**27235**	23860	**21154**	**21154**
s13207	SC [121]	**89368**	53490	45137	36784	91369	57455	49847	42239
	Golomb [122]	104440	66381	47783	47481	106169	69190	51272	50847
	FDR [123]	107059	63011	49858	41989	116101	70361	57089	48538
	VIHC	89865	**52769**	**44180**	**36065**	**90920**	**55229**	**47319**	**40048**
s15850	SC [121]	**47256**	33844	30975	28106	**47186**	35432	31687	31687
	Golomb [122]	57860	41452	33442	33442	64730	48528	32326	32326
	FDR [123]	62419	39628	33767	29488	65020	42270	36732	32362
	VIHC	47366	**32513**	**29437**	**26692**	48169	**34735**	**31055**	**30871**
s35932	SC [121]	**16870**	**11749**	**10710**	9671	**16879**	**11932**	**10780**	**9974**
	Golomb [122]	31758	26699	21640	21640	42267	38341	34415	34415
	FDR [123]	32509	20605	18438	17045	42159	27468	25893	24821
	VIHC	17584	12076	10857	**9645**	19908	15641	14670	13736
s38417	SC [121]	105076	78851	70403	70403	**116282**	98849	90354	90354
	Golomb [122]	136554	103140	86974	86974	177074	147530	117986	11798
	FDR [123]	144811	93450	81578	73182	186261	123700	113451	10521
	VIHC	**104642**	**74293**	**67940**	**62625**	118989	**92777**	**87914**	**87002**
s38584	SC [121]	**125141**	97892	90020	**82148**	**126309**	98019	89124	89124
	Golomb [122]	156238	115731	96073	96073	161236	122107	103200	10320
	FDR [123]	170143	110982	96677	85687	179530	118628	104630	93260
	VIHC	126969	**92632**	**83130**	82582	127849	**92377**	**85783**	**80392**

Table 3.9. TAT comparison

10, it can be observed that in general for small frequency ratios the SC has slightly better TAT than the proposed method (e.g., s5378, s15850 and s38584 with $\alpha = 2$ for both T_{diff} and T_D). However, generally the TATs of the proposed method are better than the ones of the previous methods (SC [121], Golomb [122], FDR [123]). The reduction in TAT when compared to previous methods is given for the same experimental setup in Table 3.10, which also reports the TAT reduction when compared to the TAT obtained for the fully compacted MinTest test set [94]. It should be noted that the TAT for the MinTest test set was considered to be the time needed to transfer the data to the core under test, hence the size of the test set in bits. It can easily observed that overall when compared to SC, TAT reduction of up to 12% is obtained for T_{diff} (e.g., in the case of circuit s9234 for $\alpha = 4$). Similarly for T_D, the TAT is reduced by up to 10% when compared to SC (e.g., in the case of circuit s9234 for $\alpha = 6$). When compared to Golomb and FDR, for the T_{diff} test set, TAT reductions up to 54% and 45% are obtained in the case of circuit s35932. For T_D, the

Circuit	Method	TAT reduction in % for T_{diff}				TAT reduction in % for T_D			
		$\alpha = 2$	$\alpha = 4$	$\alpha = 6$	$\alpha = 8$	$\alpha = 2$	$\alpha = 4$	$\alpha = 6$	$\alpha = 8$
s5378	MinTest [94]	23	44	48	51	14	33	37	38
	SC [121]	0	6	4	1	-2	4	3	4
	Golomb [122]	23	27	21	26	24	28	13	14
	FDR [123]	28	21	16	13	29	18	15	8
s9234	MinTest[94]	4	30	34	34	-5	8	18	18
	SC [121]	0	12	9	1	0	-1	10	10
	Golomb [122]	20	23	15	19	23	16	1	1
	FDR [123]	31	25	20	15	35	18	20	12
s13207	MinTest[94]	44	67	72	77	44	66	70	75
	SC [121]	0	1	2	1	0	3	5	5
	Golomb [122]	13	20	7	24	14	20	7	21
	FDR [123]	16	16	11	14	21	21	17	17
s15850	MinTest[94]	17	43	48	53	16	39	45	46
	SC [121]	0	3	4	5	-2	1	1	2
	Golomb [122]	18	21	11	20	25	28	3	4
	FDR [123]	24	17	12	9	25	17	15	4
s35932	MinTest[94]	9	37	43	50	-2	19	24	29
	SC [121]	-4	-2	-1	0	-17	-31	-36	-37
	Golomb [122]	44	54	49	55	52	59	57	60
	FDR [123]	45	41	41	43	52	43	43	44
s38417	MinTest[94]	7	34	39	44	-5	18	22	23
	SC [121]	0	5	3	11	-2	6	2	3
	Golomb [122]	23	27	21	27	32	37	25	26
	FDR [123]	27	20	16	14	36	24	22	17
s38584	MinTest[94]	21	42	48	48	20	42	46	50
	SC [121]	-1	5	7	0	-1	5	3	9
	Golomb [122]	18	19	13	14	20	24	16	22
	FDR [123]	25	16	14	3	28	22	18	13

Table 3.10. TAT reduction obtained with VIHC over previous methods

TAT reduction is as high as 59% and 43% for Golomb and FDR respectively. For the rest of the circuits, in the case of T_{diff}, the TAT ranges from similar values (when frequency ratio increases) to reduction of up to 27% in comparison to Golomb (s38417), and reduction of up to 31% in comparison to FDR (s9234). The same applies for T_D, where the TAT reduction is up to 26% (s38417) in comparison to Golomb, and up to 36% (s38417) in comparison to FDR. Comparing the TAT with the one obtained by MinTest [94] gives an idea on how the TAT behaves for different frequency ratios when compared to fully compacted test sets. As it can be easily seen in the table, when compared to MinTest, the TAT is reduced as much as 77% for the circuit s13207 in the case of T_{diff} for a frequency ratio of $\alpha = 8$. Similarly for T_D in the case of circuit s13207, TAT reduction up to 77% is obtained for $\alpha = 8$. It should be noted that MinTest should improve its TAT if serialisation buffers are introduced between the ATE and the SOC. However this implies the use of *multiple ATE channels* for one scan chain, which is avoided by the proposed approach.

	SC[121]	Golomb[122]	FDR[123]	VIHC
Compression	X	X	$\surd$	$\surd$
Area overhead	X	$\surd$	X	$\surd$
Test application time	$\surd$	X	X	$\surd$

Table 3.11. Previous approaches compared to VIHC

Finally the comparisons conducted in this section between the previous compression methods [121, 122, 123] and the proposed VIHC are summarised in Table 3.11. While FDR [123] gives compression ratios comparable to VIHC, it leads to both higher TAT and greater area overhead. On the other hand, Golomb [122] has similar area overhead when compared to VIHC at the expense of lower compression ratio and higher TAT. SC [121], on the other hand, has overall comparable TAT when compared to VIHC. However, this is achieved at a high penalty in area overhead which is the main shortcoming of the parallel decoder based on *fixed-length* Huffman coding.

3.5 Concluding remarks

This chapter has presented a new compression method called Variable-length Input Huffman Coding (VIHC) and provided a taxonomy on the factors which influence the three TDCE parameters: compression ratio, area overhead and test application time. Unlike previous approaches [121, 122, 123] which reduce some test parameters at the expense of the others, the proposed compression method is capable of improving all the test data compression parameters simultaneously. This is achieved by accounting for multiple interrelated factors that influence the results, such as pre-processing the test set, the size and the type of the input patterns to the coding algorithm, and the type of the decoder. The results in Section 3.4 show that the proposed method obtains constantly better compression ratios than [121, 122, 123]. Furthermore, employing the parallel decoder leads to savings in TAT when compared to serial decoders [122, 123]. Moreover, by exploiting the variable-length input approach, great savings in area overhead are achieved (up to threefold reduction when compared to fixed-length approach [121]). Thus, it was shown in this chapter that the proposed method reduces the ATE memory and channel capacity requirements by obtaining good **compression ratios**, and reduces **TAT** through its parallel on-chip decoder with small **area overhead**. With small volume of test data and small TAT, the proposed solution reduces two of the three low cost test parameters: volume of test data and bandwidth requirements, and hence, it contributes toward low cost SOC test.

Chapter 4

Synchronisation overhead in test data compression environments

Chapter 3 proposed the Variable-length Input Huffman Coding (VIHC) compression method to reduce the volume of test data required for testing. Testing in test data compression environments (TDCE) implies sending the compressed test data from the ATE to the on-chip decoder, decompressing the test data on-chip and sending the decompressed test data to the core under test (CUT). Due to resource partitioning among the ATE and the on-chip decoder, synchronisation between the two is required, as also illustrated in Figure 3.5 (see Section 3.3 on Page 49). Hence, a synchronisation overhead is incurred which limits the effectiveness of test data compression methods. Synchronisation overhead includes one or more of the following: the use of multiple ATE channels, adapting ATEs to start and stop the data stream in real time, on-chip serialisation units and the extensions to the device interface boards (DIB). This chapter provides an analysis of synchronisation overhead and a new solution to reduce it. The proposed solution targets parallel on-chip decoders, it does not impose changes to the existing decoders and it provides easy design flow integration. The effectiveness of the proposed approach is illustrated using the VIHC compression method described in Chapter 3.

The remainder of this chapter is organised as follows. The next section outlines the importance of synchronisation in TDCE and analyses previous approaches to reduce the synchronisation overhead. Section 4.2 proposes the new method for synchronisation overhead reduction. Section 4.3 and 4.4 give experimental results and conclusions respectively.

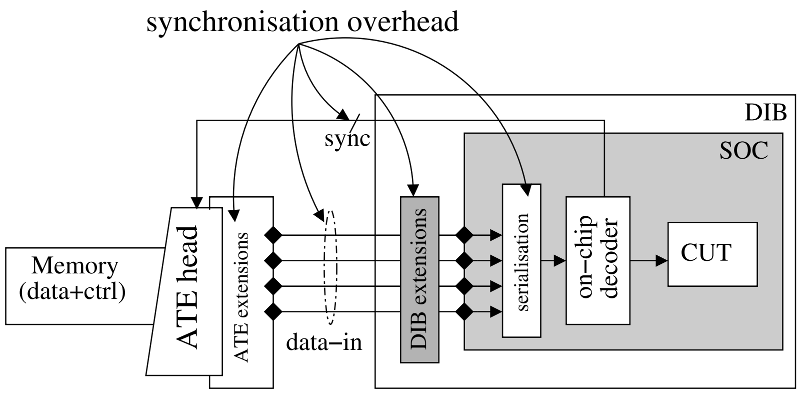


Figure 4.1. Test data compression environment with synchronisation overhead

4.1 The importance of synchronisation in TDCE

As it will be seen in this chapter, synchronisation overhead includes one or more of the following: (*i*) the use of multiple ATE channels; (*ii*) adapting ATEs to start and stop the data stream in real time; (*iii*) on-chip serialisation units; and (*iv*) extensions to the DIB. Each of the above influences the cost of test and diminishes the possibility of a low-cost methodology using test data compression (TDC), as explained next:

- the use of multiple ATE channels (*i*), diminishes the effectiveness of reduce pin count test (RPCT) and lead to inefficient ATE resource usage;
- the adaption of ATEs to start and stop the data stream in real time (*ii*) requires ATE extensions, such as source synchronous buses, which can increase cost;
- the usage of on-chip serialisation units (*iii*) increases the area overhead on-chip;
- the extensions to the DIB (*iv*) may lead to prohibitive high DIB costs due to the complex designs of DIB when high frequency SOCs are tested [63, 64, 13].

When all of the above are included, the ATE – DUT interface is illustrated in Figure 4.1. As noted above, all the extensions will lead to increased test cost, hence, synchronisation is an important issue which requires cost effective solutions in order to take full advantage of the benefits of TDC. In this section the above issues will be detailed and an analysis of previous approaches which have targeted synchronisation issues provided. It should be noted that the synchronisation overhead is addressed with respect to single scan chain designs.

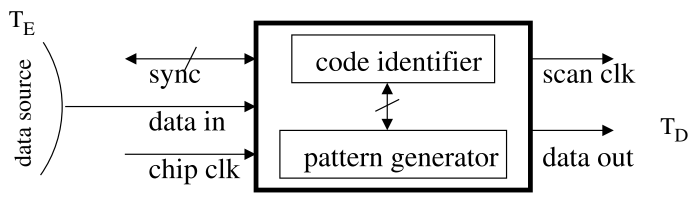


Figure 4.2. Generic on-chip decoder

4.1.1 Synchronisation overhead

To understand why synchronisation overhead is implied in TDC, in this section a generic decoder is illustrated and, depending on the type of the decoder, the synchronisation requirements are detailed.

An on-chip decoder can be conceptually seen as comprising a *code identifier* and a *pattern generator* unit (Figure 4.2). Based on the dependencies between these two units, on-chip decoders can be characterised as *serial*, i.e., during the decompression of the current code, no new code can be identified, or *parallel*, i.e., the decompression of the current code and the identification of a new one can be done simultaneously. The “data source”, in the figure, provides the compressed test set (T_E) to the decoder’s “data in” and can be either on-chip or off-chip. While the primary data source is on the ATE, some decoders require on-chip serialisation units which will therefore become their “data source”. The “sync” lines ensure the synchronisation between the on-chip decoder and the “data source”, and can be composed of ATE clock signals, start/stop signals from the decoder to the ATE or to the on-chip source, and special clock signals. The output of the decoder (“data out”) drives the internal scan chain synchronously with the “scan clk”. The compressed data is sent from the ATE to the DIB at the ATE operating frequency (f_{ate}), and the decompressed data is sent from the decoder to the scan chain with the chip test frequency (f_{chip}).

In order to reduce TAT, when feeding single scan chain designs, the pattern generator needs to operate at a frequency greater than f_{ate}. This is explained as follows: if the pattern generator is operating at f_{ate}, then the time required to generate the initial test set (T_D) will be the same as if T_D is directly transported from the ATE to the CUT. In addition, the code identifier unit adds its latency, which is the time required to identify a code. Thus, with the pattern generator running at f_{ate} the TAT is **not** reduced, rather it is actually increased! Therefore, to reduce TAT the ratio between the on-chip test frequency and the ATE operating frequency must be exploited ($\alpha = \frac{f_{chip}}{f_{ate}}$). It should be noted that when the

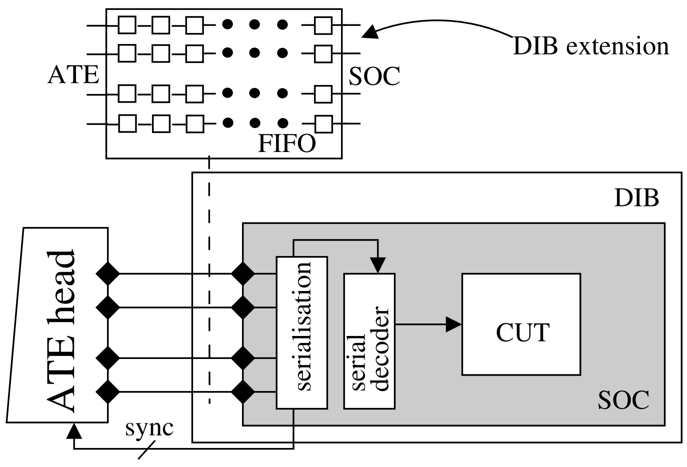


Figure 4.3. Serial decoder's synchronisation overhead

code identifier can identify any code in one ATE clock cycle and the pattern generator can generate any pattern in one ATE clock cycle, then there is no need to exploit the frequency ratio, since no delay is introduced by the on-chip decoder. However, in order for this approach to be efficient multiple data inputs are required, and the output of the decoder must feed multiple scan chains. For single scan chain designs, the synchronisation requirements needed to exploit the frequency ratio are decoder dependent as shown next.

(i) When a **serial decoder** is employed, in order to exploit α the entire decoder operates at f_{chip}. Hence, the "data source" is constrained to providing data at f_{chip} using techniques like the ones described in [128] and [65]. This implies the usage of *multiple ATE channels* and a serialisation unit on-chip or on the DIB. This is exemplified in Figure 4.3, where for $\alpha = 4$, the ATE channels and the serialisation unit are illustrated. In addition, when the current pattern is generated no more data can be processed by the code identifier, hence, the "sync" lines need to notify the ATE to stop sending data. Since, some ATEs are not capable of stopping the data stream immediately and require a number of clock cycles to change the state of the channel, the DIB must account for this latency. Therefore, the interface between the ATE and the chip requires a first-in first-out (FIFO)-like structure to solve this problem. The number of elements in the FIFO is given by the maximum number of clock cycles needed to start/stop the data stream on the channels. The FIFO-like structure is marked in the figure as DIB extension. Hence, a serial decoder will not only require α input ATE channels, but also an α bit wide FIFO-like structure.

(ii) When a **parallel decoder** is employed, the code identifier can receive data at f_{ate}, and the pattern generator can generate data at f_{chip}. Hence, the decoder implicitly

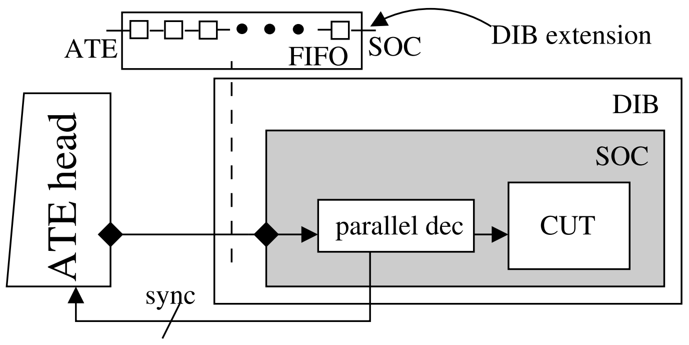
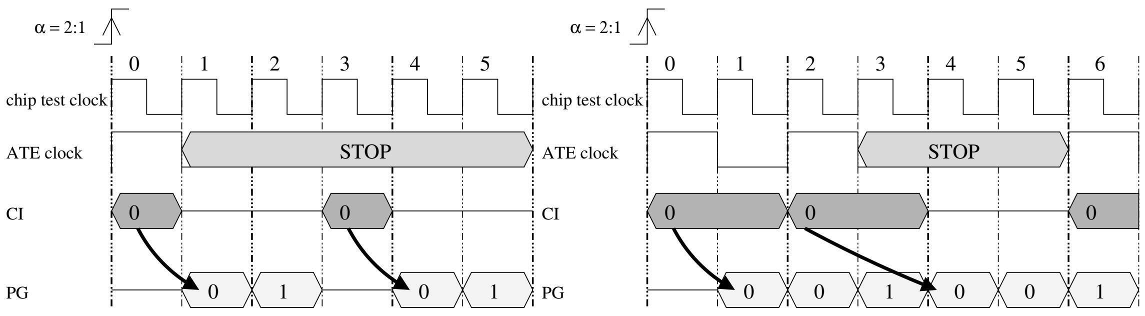


Figure 4.4. Parallel decoder's synchronisation overhead

exploits the frequency ratio, and therefore only one ATE channel is required for feeding data to the decoder as shown in Figure 4.4. Despite this advantage, if the frequency ratio is less than an *optimum frequency ratio*, i.e., α_{max} is the frequency ratio at which the pattern generator will generate any pattern in the number of ATE clock cycles needed by the decoder to identify any codeword, the on-chip decoder will notify the ATE to stop sending data. As also noted earlier, the ATE has a certain latency to starting/stopping the data stream on a channel and therefore a FIFO-like structure is required, which is marked in the figure as DIB extension. Hence, when compared to a serial decoder, the parallel decoder has less synchronisation overhead with respect to the interface between the ATE and the DUT. However, since the two units (the code identifier and the pattern generator) can work independently explicit communication between two units is required.

To illustrate the fundamental distinction between serial and parallel decoders, and at the same time to illustrate the main source of synchronisation overhead in Figure 4.5 the functionality of the two types of decoders is exemplified considering a frequency ratio of $\alpha = 2$. The serial decoder receives data at the f_{chip} frequency and it generates data at the f_{chip} frequency, while the parallel decoder receives data at the f_{ate} frequency and generates data at the f_{chip} frequency. The stopping of the data stream is illustrated for a serial decoder in Figure 4.5(a). It can be seen in the figure that the code identifier (CI in the figure) works at the on-chip test frequency and that while the pattern generator (PG in the figure) generates data, the code identifier will not process any new data. This will lead to stopping the ATE data stream as shown in the figure. The stopping of the ATE data stream is illustrated for a parallel decoder in Figure 4.5(b). It can be clearly seen in the figure that the code identifier can work independently from the pattern generator, i.e., the CI can receive data while the PG generates data, and that the ATE data stream is stopped only when required.



(a) Serial decoder (b) Parallel decoder

Figure 4.5. The stopping of the ATE data stream

With the advent of new generation ATEs, one may consider exploiting the source synchronous buses feature to eliminate the DIB extensions introduced by the starting/stopping of the data stream. However, in addition to being an expensive feature which requires new investment, the starting/stopping of the data deployment on a channel must be programmed. Hence, the DIB extensions can be eliminated at the expense of programming the starting/stopping of the data stream within the ATE memory. In addition, as it will be seen in Section 4.2.1, halting the ATE data stream is not periodical, i.e., it does not happen once in 3 ATE clock cycles, and therefore, the ATE must know exactly when the data stream must be started, and stopped. This can lead to prohibitively larger ATE programs, hence, larger memory requirements, and it will not reduce the TAT.

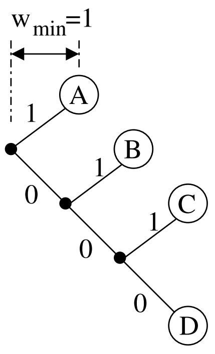
Summarising, the *synchronisation overhead* of a decoder accounts for the number of ATE channels, ATE adaption to starting/stopping the test data stream in real time, DIB extensions and on-chip requirements for data transfer and synchronisation. As detailed in Figures 4.3 and 4.4 the *synchronisation overhead* makes TDC a less appealing solution for SOC testing, hence, reducing this overhead is an **important** issue which necessitates cost-efficient solutions in order to take full advantage of the benefits offered by TDCE.

4.1.2 Previous approaches to reduce synchronisation overhead

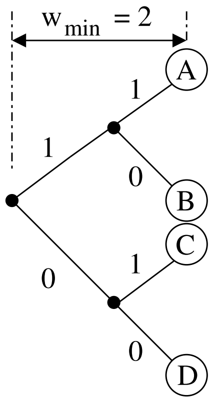
As illustrated in the previous section, pending the type of decoder, different synchronisation schemes are required. In this section the approaches proposed in [121] and [122] to reduce the synchronisation overhead are analysed. It should be noted that these approaches are not generally applicable.

pattern	occ.	code	code–2
A	10	1	11
B	3	01	10
C	1	001	01
D	2	000	00
VTD (bits)		25	32

(a) Dictionary



(b) Initial Huffman tree



(c) Tailoring for $w_{min} = 2$

Figure 4.6. Tailoring the compression method

Selective coding (SC) [121] employs a parallel on-chip decoder to decompress the test set on-chip. The SC decoder, uses a modified Huffman finite state machine (FSM) as the code identifier, and a serialiser as the pattern generator. The code identifier is receiving data at the ATE operating frequency, and the pattern generator, generates data at the on-chip test frequency. Hence, synchronisation with the ATE is required if $\alpha < \alpha_{max}$. To solve this problem, [121] tailors the compression method to suite α. With $\alpha_{max} = \frac{b}{w_{min}}$ [121], where b is the block size and w_{min} is the minimum length codeword, the synchronisation overhead is reduced if $w_{min} \geq \frac{b}{\alpha}$. Hence, to reduce the synchronisation overhead for a given block size, w_{min} has to be increased. This can be done in two ways as follows. (a) If the number of patterns chosen for selective coding do not yield the required w_{min}, the number of patterns can be increased. Since w_{min} is the code length for the pattern with the greatest number of occurrences, the number of patterns required to increase w_{min} can be very large. While this could increase compression ratio, it also increases the on-chip decoder area considerably [121]. (b) A second approach, assuming a fixed number of patterns, is to increase w_{min} by manipulating the Huffman tree [127] used to obtain the selective code, and appending bits to the codewords which are shorter than required. This, however, has a direct impact on the compression ratio since increasing w_{min} increases the average codeword length, which reduces compression ratio leading to greater volume of test data (VTD), as illustrated next.

Consider the four patterns (A,B,C and D) and their number of occurrences given in Figure 4.6(a) in the first two columns. As detailed in Section 3.2.1 (see Page 41) based on these two columns the Huffman tree can be built. The corresponding Huffman tree is illustrated in Figure 4.6(b). Based on the Huffman tree, the codes are derived and shown in the third column in Figure 4.6(a). It can be observed that $w_{min} = 1$. If for example, $b = 4$ and $\alpha = 2$, then based on the formula shown previously $w_{min} \geq \frac{4}{2} = 2$, in order to eliminate the synchronisation overhead. Tailoring the compression method to obtain $w_{min} = 2$ is illustrated with Figure 4.6(c), and the codes obtained based on this new tree are shown Figure 4.6(a) in the fourth column. Summing up, the VTD for the two codes is 25 and 32 bits respectively. Hence, it becomes clear that tailoring the compression method can lead to an increase in VTD. It should be noted that if $b = 6$, a w_{min} of 3 can only be obtained if the codes are padded.

Thus, the synchronisation overhead is reduced at the expense of increased VTD and a frequency ratio dependent on-chip decoder. Hence, if testing is sought at different frequencies, the ATE is constrained to operate at a new frequency such that the frequency ratio, between the CUT and the ATE, remains the same. With already limited ATEs, the on-chip decoders should fully exploit the ATE capabilities and not further constrain them. Therefore, Section 4.2.1 proposes to *tailor the compressed test set*, and not the compression method, for a given frequency ratio. In addition to obtaining similar and smaller test sets than the above approach with **no** changes to the on-chip decoder, tailoring the compressed test set is generally applicable to decoders which require a FIFO-like structure.

Golomb [122] uses a serial decoder to provide decompressed test data to the CUT. Hence, to reduce TAT the decoder *requires* DIB extensions and/or on-chip serialisation units, as illustrated in Figure 4.3 (Section 4.1.1). To reduce this synchronisation overhead, in [122] an interleaving architecture was introduced which decompresses the bit stream for multiple cores. The architecture is illustrated in Figure 4.7. The main idea behind this architecture is to exploit the particularities of the Golomb decoder, i.e., when a '1' bit is encountered by the Golomb decoder a run of '0's of length equal to the group size is generate, and using a FSM and a de-multiplexer multiple cores can be tested. While the architecture can use a single ATE channel, it still requires the FIFO-like structure. This is because, while processing the encoded bit stream, the FSM can notify the ATE that no more data should be sent [134]. It should be noted that the architecture does not exploit the frequency ratio, and it reduces TAT by supplying data to multiple cores in an interleaved manner by activating the corresponding decoder when required. Since Golomb's decoder

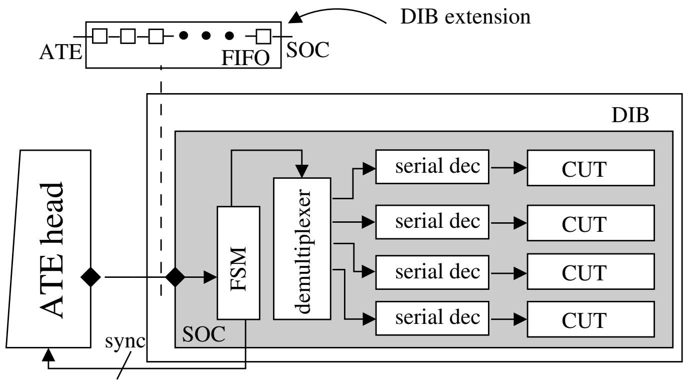


Figure 4.7. Interleaving architecture

is a serial decoder, the FSM will have to explicitly control each decoder connected in the interleaving architecture. In order to synchronise the FSM and the decoders, *changes* to the on-chip decoders are *necessary*. In addition, the interleaving architecture imposes the following two restrictions: all the cores which are driven by it have their test sets compressed using the same group size and the number of cores tested using this architectures is constrained by the group size [134]. These two restrictions are due to the exploitation of the above mentioned Golomb decoder property. Indirectly, these two restrictions will eliminate the need for the DIB extension at the expense of imposing constraints on the system integrator.

To remove the above restrictions for multiple core test, Section 4.2.2 proposes a distribution architecture which does **not** require any changes to the on-chip decoders, and it is also **scalable** and **programmable**. The proposed architecture is applicable to any parallel decoder. Furthermore, when combining the solutions described in Section 4.2, a complete RPCT scenario for SOCs is provided with low storage requirements and low TAT without any constraints or changes to the test or interface equipment, e.g., ATE or DIB.

4.2 Reducing synchronisation overhead

The solution proposed to reduce the synchronisation overhead is divided into two parts. The first part, illustrated in Section 4.2.1, reduces the FIFO-like structure by introducing *dummy* bits within the compressed test set, hence tailoring the compressed test set. The second part, illustrated in Section 4.2.2, exploits the inserted dummy bits and provides a complete solution for system test using a distribution architecture. While the above are

Pattern	Occurrence	Code
L_0 = 1	1	000
L_1 = 01	1	001
L_2 = 001	1	010
L_3 = 0001	1	011
L_4 = 0000	4	1

(a) Dictionary

$$m_h = 4 \qquad |t_{init}| = 26 \text{ bits} \qquad |t_{cmp}| = 16 \text{ bits}$$

$$t_{init} \quad \underbrace{1}_{000}\ \underbrace{01}_{001}\ \underbrace{0000}_{1}\ \underbrace{0000}_{1}\ \underbrace{0000}_{1}\ \underbrace{0001}_{011}\ \underbrace{0000}_{1}\ \underbrace{001}_{010} \quad t_{cmp}$$

(b) Test sets

Figure 4.8. VIHC - an example

introduced using the Variable-length Input Huffman Coding (VIHC) scheme illustrated in Chapter 3, they are generically applicable to any method which employs a parallel on-chip decoder.

The VIHC scheme splits the test set into runs of ’0’s of length smaller than, or equal to, the group size (m_h). The obtained patterns are then encoded using Huffman coding. This is exemplified, for $m_h = 4$, in Figure 4.8, where one test vector is processed. Using VIHC the initial test vector (t_{init}) is compressed (t_{cmp}) (see Figure 4.8(b)). The dictionary employed in the compression process is illustrated in Figure 4.8(a). The VIHC method uses a parallel on-chip decoder comprising a Huffman decoder as the code identifier and a control and generation unit (CGU) as the pattern generator. When the Huff-decoder identifies a codeword, the CGU starts generating the pattern after two on-chip clock cycles (see Section 3.3 on Page 48). For the distribution architecture proposed in this chapter, the following signals are of interest. The *ATE sync* represents the signal used by the CGU to notify the ATE to stop sending data, the *FSM clk* represents the clock used to drive the code identifier unit, the *dec clk* represents the CGU clock, the *data out* represents the data output of the decoder, and the *scan clk* represents the clock generated by the CGU to control the scan chains.

4.2.1 Tailoring the compressed test set

It was shown in Section 3.3.2 (see Page 52) that the optimum frequency ratio for the VIHC decoder is given by $\alpha_{max} = \frac{m_h}{w_{min}}$, where w_{min} is the minimum codeword length. Hence if $\alpha < \alpha_{max}$, the CGU will notify the ATE to stop sending data. This is illustrated using the following example for the case considered in Figure 4.8.

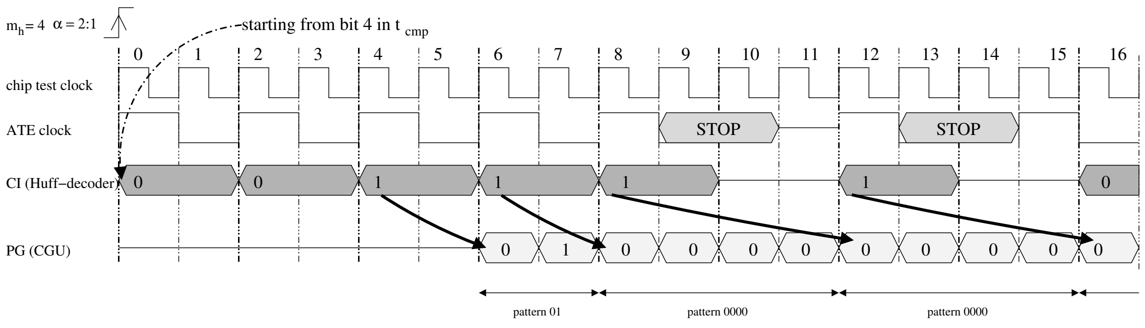


Figure 4.9. Timing diagram to illustrate the stopping of the data stream

Example 4.1 Based on Figure 4.8, it can be immediately derived that $\alpha_{max} = \frac{4}{1} = 4$. To illustrate the stopping of the data stream, for the remainder of this section, $\alpha = 2$. For the example in Figure 4.8(b), Figure 4.9 shows a timing diagram when t_{cmp} is decompressed using the VIHC decoder. The arrows in the figure point from the compressed test bit which lead to the identification of a codeword, to the clock cycle when the generation of the corresponding pattern starts. The diagram captures the activity of the decoder's parts starting with the second codeword from t_{cmp} (001 in Figure 4.8(b)). The reference clock has been considered the on-chip test clock. The time frame of interest is from the $6th$ to the $15th$ clock cycle, where there are two stop cycles in which no data from the ATE is processed. At the $6th$ clock cycle the codeword "1" is identified. The CGU, at clock cycle 8, starts generating the corresponding pattern "0000". However, since the next codeword is "1" there are not enough clock cycles to generate the pattern, i.e., the code identifier requires *1 ATE clock cycle* to determine the next codeword, while the CGU requires *4 on-chip clock cycles*, or *2 ATE clock cycles*, to generate the current pattern. Hence, the ATE will be stopped for 1 clock cycle. Similar for clock cycle 13.

The reason for stopping the ATE is to make sure that no data is lost while the on-chip decoder is unable to process it. However, if care is taken to ensure that the data which could be lost is not relevant, there is no need to stop the ATE in the first place. To achieve this the compressed test set is *tailored for the current frequency ratio*. This works as follows. For the number of ATE clock cycles in which the CGU sets *ATE sync* to low, thus notifying the ATE to stop sending data, *dummy* (D) bits are inserted in the compressed test set. For a given frequency ratio (α) the number of dummy bits is obtained using $\lceil \frac{|L_i| - w_{i+1} \cdot \alpha}{\alpha} \rceil$, where $|L_i|$ represents the length of the current pattern and w_{i+1} represents the length of the next codeword (see Section 3.3.2 on Page 52). Hence, since there is no need to stop the ATE the FIFO-like structure can be removed thus eliminat-

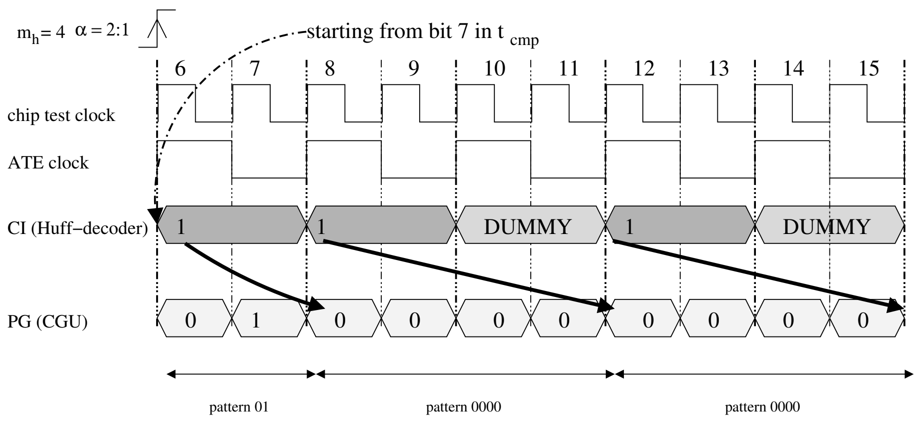


Figure 4.10. Timing diagram to illustrate tailoring of the compressed test set

ing the synchronisation overhead. This is exemplified, for the time frame of interest, in Figure 4.10 where instead of stopping the ATE, the dummy bits are sent to the on-chip decoder. As these bits will not be processed by the decoder they will not affect the correct generation of the decompressed test set. Applying this strategy to the entire t_{cmp}, t'_{cmp} = 000 001 1 1D 1D 011 1D 010 is obtained. Note that the Huff-decoder will not process the dummy bits since it is halted by the CGU when the pattern was not fully generated (see Section 3.3.1 on Page 49). Since, tailoring the compressed test set inserts dummy bits into the test set, it increases the VTD, however, the TAT is equal to the TAT of the initially compressed test set. In addition, since the ATE must not start/stop when requested by the CGU, the size of the tailored test set will equal the TAT.

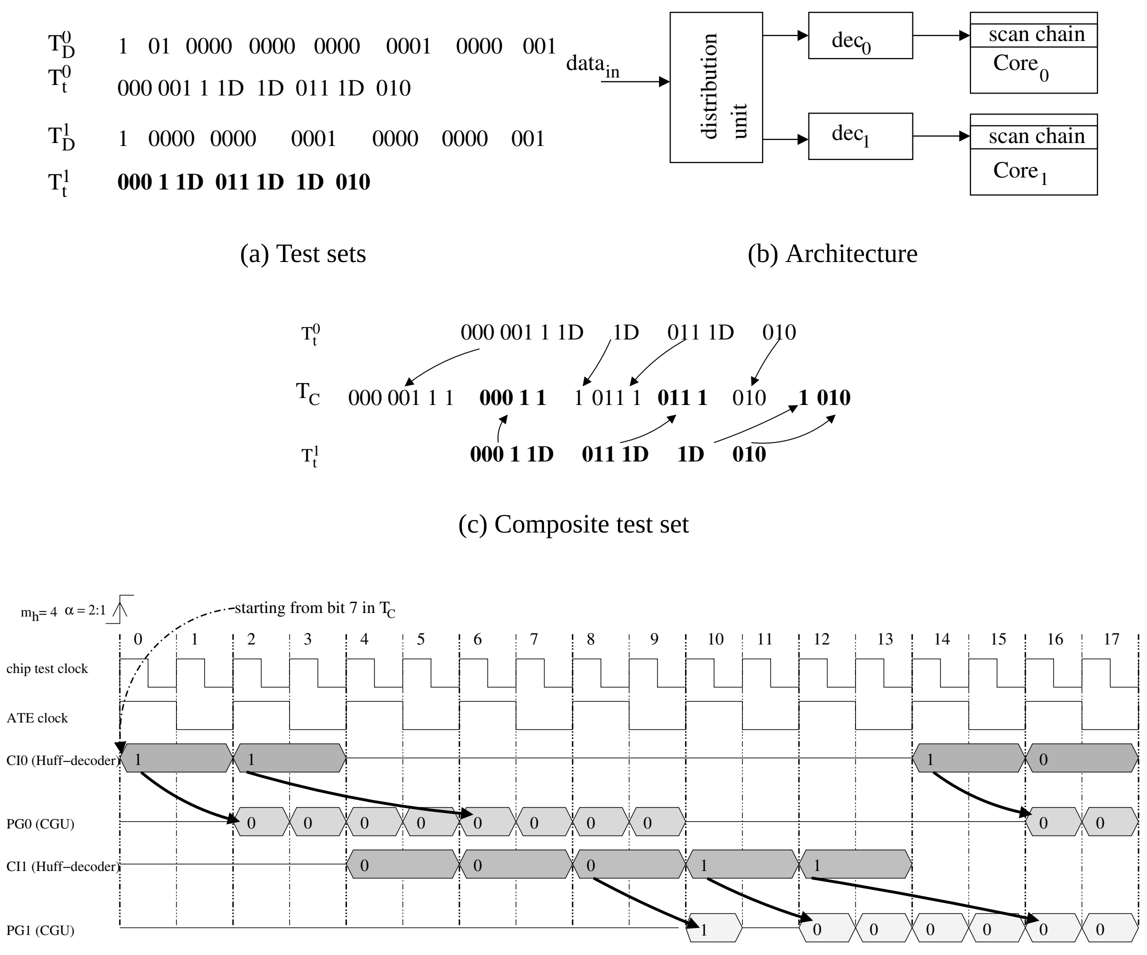
As illustrated above this method is easily applicable with no changes to the on-chip decoder. It will be seen in Section 4.3 that this approach is especially suitable when the ratio between m_h and α is rather large (e.g. $\frac{m_h}{\alpha}$ *is* 4 *or* 8). This is due to the large number of padding bits required, by tailoring the compression method, in these cases. For example assume that when the test set is compressed using $m_h = 16$, $w_{min} = 1$. If the TDCE has $\alpha = 2$, then in order to reduce synchronisation overhead, w_{min} will be 8. Thus, tailoring the compression method will increase all the codewords to be of size 8, hence reducing the compression ratio.

Since the proposed approach does not change the code used for compression nor the on-chip decoder, it is generally applicable to methods which require a FIFO-like structure for synchronisation. Despite this advantage, tailoring the compressed test set and tailoring the compression method increase the volume of test data stored on the ATE. Hence, these methods bring forth a trade-off between the synchronisation overhead and the amount of memory required on the ATE. To solve this problem in the following section a distribution architecture to test multiple cores is proposed. In contrast to the interleaving architecture from [122], the new architecture does **not** require changes to the on-chip decoders, and is **scalable** and **programmable**.

4.2.2 Distribution architecture for multiple core test

Testing multiple cores using only *one ATE channel* implies exploiting the clock cycles when the decoder corresponding to a core is not capable of processing data. For example, when using the MinTest [131] test set for the s13207 full scan ISCAS89 [132] benchmark circuit, and applying VIHC with $m_h = 8$, the compressed test set is 38123 bits. However, the TAT for $\alpha = 2$ is of 89646 ATE clock cycles. Hence, there are 51523 ATE clock cycles in which the decoder is not processing data, or there are 51523 *dummy* bits in the tailored compressed test set stored on the ATE. Exploiting these clock cycles, for multiple core test when one ATE channel is used, leads to merging the test sets into one composite test set, using one decoder per core and adding the necessary, core and decoder, selection hardware. The above are illustrate using the following example.

Example 4.2 Assume a system composed out of two cores ($Core_0$ and $Core_1$) and their corresponding decoders (dec_0 and dec_1), with the test sets given by T_D^0 and T_D^1 in Figure 4.11(a). Both test sets have been compressed using the codes given in Figure 4.8(a) and tailored for a frequency ratio of $\alpha = 2$ (T_t^0 and T_t^1 in Figure 4.11(a)). It can be observed that, in order to reduce the synchronisation overhead, both test sets require three dummy bits. Using only one ATE channel the test data can be send to the cores by careful selection of the decoder which receives data. This is illustrated generically in Figure 4.11(b) with the distribution unit. The unit is receiving and sending data to the selected decoder at the ATE operating frequency. The details of the unit will be detailed later in this section. Using only one ATE channel, the data sent to the distribution unit, referred to as the composite test set, has to correspond to the selected decoder. The process of generating the composite test set (T_C) and the distribution's unit functionality are detailed next. As noted in the beginning of the section, testing multiple cores using one ATE channel is possible if the dummy bits within the tailored compressed test set are exploited. Hence, starting with the tailored compressed test sets, the composite test set is generated as illustrated in Figure 4.11(c). Firstly, the bits from the first test set are copied


T_D^0 1 01 0000 0000 0000 0001 0000 001
T_t^0 000 001 1 1D 1D 011 1D 010
T_D^1 1 0000 0000 0001 0000 0000 001
T_t^1 **000 1 1D 011 1D 1D 010**
data$_{in}$
distribution unit
dec$_0$
dec$_1$
scan chain
Core$_0$
Core$_1$
T_t^0 000 001 1 1D 1D 011 1D 010
T_C 000 001 1 1 **000 1 1** 1 011 1 **011 1** 010 **1 010**
T_t^1 **000 1 1D 011 1D 1D 010**
$m_h = 4$ $\alpha = 2:1$
starting from bit 7 in T_C
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
chip test clock
ATE clock
CI0 (Huff–decoder)
PG0 (CGU)
CI1 (Huff–decoder)
PG1 (CGU)

(a) Test sets (b) Architecture

(c) Composite test set

(d) Timing diagram

Figure 4.11. Distribution architecture for two core test

into T_C up to the first *dummy* bit. At this point, the next test set is chosen, T_t^1 (bold is used to represent the data from T_t^1), and the bits up to the first dummy bit are copied into T_C. To illustrate the implications in the distribution's unit functionality when the changing of the test set occurs, a timing diagram is presented in Figure 4.11(d). Each decoder has been represented with the code identifier (CI) and the pattern generator (PG) in the figure. It can be seen that at the $4th$ on-chip clock cycle CI_1 becomes active. Hence, instead of sending a dummy bit to decoder CI_0 as in the case of the tailored test set, useful test data is sent to decoder CI_1. Since, the test data from T_t^1 has been added to T_C, CI_1 will receive the correct compressed test bits. While switching of the test sets based on the occurrence of the dummy bits is usually performed, there are some exceptions which have to be accounted for. For example, bit stream "1D 011 1D" in the tailored compressed test set for T_t^0 was copied into T_C without the first dummy bit. This is due to the fact that when first codeword 1 from the above bit stream is received by CI_0, the pattern generator does not generate any more data, and therefore it can commence the generation of the pattern immediately after the codeword has been identified (clock cycle 14 in Figure 4.11(d)). Note that this is contrary to Figure 4.9 (clock cycle 13), where the pattern generator was still busy generating the pattern corresponding to a previous codeword, and therefore the dummy bit was inserted in the T_t^0. In general, if $\frac{|L_i|-w_{i+1}\cdot\alpha}{\alpha}$ is negative then there is no need to insert the dummy bit into T_C. It is important to note that, if the number of bits copied into T_C are not enough to account for the number of dummy bits from the previous test set, and if there are no more test sets available, then dummy bits are inserted into T_C. Timing diagrams illustrating all the ATE data stream stops for both compressed test sets, and the changing of the active decoder for the next time frame of interest are given in Appendix C. In addition, the schematic for the distribution architecture and a gate level timing diagram for this example are also provided.

The distribution architecture which can make use of the composite test set as generated above is given for the general case of k cores in Figure 4.12. The distribution architecture comprises: a log_2k bit counter, a k to 1 multiplexer, a $\log_2 k$ to k decoder and a k bit register (*prog reg*). The counter drives the $\log_2 k : k$ *decoder* and the $k : 1$ *mux*. The en_i output of the $\log_2 k : k$ *decoder* is 1 if the counter has the appropriate value enabling the decoder dec_i. The synchronisation elements (*sync elem*) illustrated in the figure, represent blocks of logic which gate the *FSM clk* of the corresponding decoder as follows: if the decoder is enabled, the *FSM clk* is transmitted; otherwise, the *FSM clk* is low. Alternatively, in order to completely disable the decoder after the pattern has been generated one may also consider gating the *dec clk* signal (this is illustrated with *add. gating* in Figure 4.12). This however is not necessary since, as illustrated in Figure 3.8(c) (see Page 51), the "load control FSM" will remain in state $S1$ unless the FSM clock is 1. With the FSM clock gated, the pattern generator will load a new pattern only after the decoder has been enabled again and a new code has been identified.

After reset, the counter is selecting decoder dec_0. As long as the *ATE sync* signal is high, dec_0 processes the data from the $data_{in}$. Once the *ATE sync* is set to low, the counter will increment enabling the next decoder which has the *ATE sync* line high to process data from the ATE. It should be noted that the *ATE sync* is set low when the pattern generator is still busy processing the previous pattern and can not commence the generation of the new one. This state corresponds to the changing of test sets as illustrated in Example 4.2.

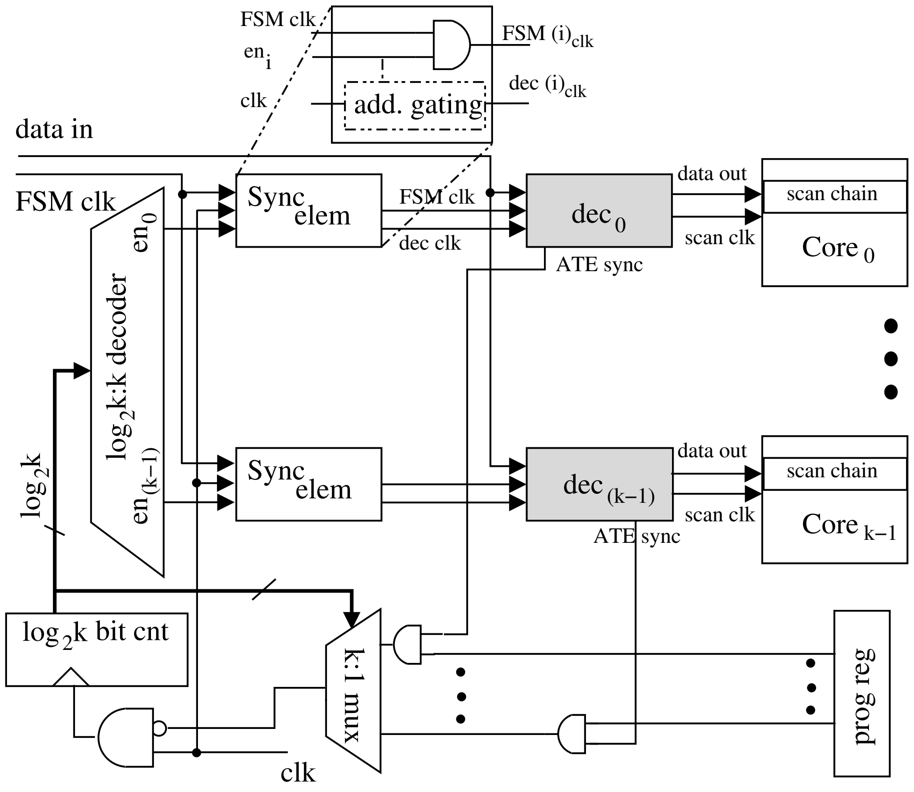


Figure 4.12. Distribution architecture for multiple core test

To summarise, for a system with *k* cores, employing this architecture implies the following steps. Firstly, each core's test set is compressed such that the compression ratio, area overhead and TAT meet the system's constraints. Secondly, for the decoders which have α_{max} greater than the available frequency ratio, the tailored compressed test sets are determined. Finally, the tailored compressed test sets are merged together into a composite test set (T_C) as illustrated with Example 4.2. If there are test sets which do not require tailoring, and dummy bits have to be inserted into T_C, then the distribution architecture can be extended with one more decoder. This decoder corresponds to the test sets which do not require tailoring. When all the decoders connected to the distribution architecture are busy, then the data can be redirected to this separate decoder, and thus no dummy bits are required. It should be noted that, the size of T_C is lower bounded by the sum of the sizes of the initially compressed test sets and upper bounded by the sum of the sizes of the tailored compressed test sets. This is because, if no dummy bits are inserted into the composite test set, the size of the composite test set is given by the sum of sizes the initially compressed test sets. If however, dummy bits are required, then the number of dummy bits is always smaller than the sum of the dummy bits in the tailored compressed test sets.

Since the distribution architecture exploits the notification signals from the VIHC decoder, it does not have to explicitly control each decoder, but rather disable them when appropriate. This leads to **scalability**. Therefore, the architecture can handle *any number of VIHC decoders of any group size*. An additional advantage of the proposed architecture is **programmability**. This is achieved by using the *prog reg* register. If some cores have finished the test faster than the others, then the decoders corresponding to these cores are no longer needed. To disable these decoders, the register can be loaded with the appropriate value. It is important to note that, these two features, **scalability** and **programmability**, are missing from the interleaving architecture proposed in [122, 134]. Furthermore, multiple cores must be assigned to the same decoder in order to exploit the interleaving architecture [122, 134], thus increasing the length of the scan chains in the SOC, which can lead to excessive power dissipation.

While both architectures can be used to test multiple scan chain cores, similar to the case of multiple core test, the interleaving architecture [122, 134] imposes the same restrictions (see Section 4.1.2). In addition, care must be taken to ensure that the scan chains' lengths are equal. It is important to note that the scalability feature of the proposed architecture has important advantages in reduced pin count test. For example, all the cores in a system can be compressed to achieve maximum compression, and after the composite test set is constructed, with virtually no synchronisation overhead and **one ATE channel** the entire system can be tested. As explained previously, the interleaving architecture [122] does not have this feature, and hence, more interleaving architectures are required in order to fully exploit the compressed test sets. An additional important advantage of the proposed distribution architecture is that in combination with a parallel decoder it provides data to the core under test at on-chip test frequency; this, while using only one ATE channel. Hence, the architecture has the potential to bridging the gap between ATE bandwidth availability and SOC test bandwidth requirement.

4.3 Experimental results

To validate the efficiency of the proposed methods, experiments have been performed on the full-scan version of the largest ISCAS 89 benchmark circuits [132] using the MinTest [131] test sets. The experiments have been executed on a Pentium III 500MHz workstation with 128 MB DRAM. Details about the tools and the benchmark circuits used in the

Circuit	m_h	$\lvert T_E \rvert$ (bits)	Tailored $\lvert T_E \rvert$ (bits)			
			$\alpha = 2$		$\alpha = 4$	
			cmp	*set*	*cmp*	*set*
s5378	4	13983	**16554**	17703	13983	13983
	8	13575	21892	**18705**	**13622**	15080
	16	12604	34952	**17668**	17547	**13740**
s9234	4	21494	**25730**	27144	21494	21494
	8	21494	29888	**25931**	21494	21494
	16	21151	44736	**27334**	22610	**21785**
s13207	4	53523	**88997**	90322	53523	53523
	8	38123	93496	**89646**	**54419**	55146
	16	32444	107008	**90920**	**53862**	55229
s15850	4	32326	**45405**	47123	32326	32326
	8	27922	50500	**47196**	**32137**	34195
	16	27666	66904	**49070**	**33940**	35266
s35932	4	19813	28952	**20050**	19813	19813
	8	15730	48376	**20021**	24751	**15881**
	16	13585	88120	**19908**	44060	**15641**
s38417	4	103470	**123525**	129395	103470	103470
	8	92610	171060	**119297**	101568	**94355**
	16	86012	279648	**118989**	140134	**92777**
s38584	4	90730	**121347**	127263	90730	90730
	8	82003	139336	**129604**	**89993**	97081
	16	80392	193416	**127849**	97590	**92377**

Table 4.1. Tailoring compression method vs. tailoring compressed test set

experiments are given in Appendix F. The algorithms to tailor the compression method, and to tailor the compressed test sets have been implemented for VIHC. In addition, in order to provide a comparison between the two architectures for multiple core test (the interleaving architecture [122, 134] and the proposed distribution architecture), the generation of the composite test set for the two architectures has been also implemented. It should be noted that, since the synchronisation overhead is removed, the results reported in this section represent both volume of test data (VTD) and test application time (TAT).

Tailoring the compressed test set The comparison between tailoring the compression method and tailoring the compressed test set is given in Table 4.1. The table lists the circuit, the group size (m_h), the volume of test data ($\lvert T_E \rvert$) obtained after compression, the size of the test set after tailoring the compression method (*cmp*) and after tailoring the compressed test set (*set*), when the group size is 4, 8 and 16. Tailoring the compression

Method	Tailored $\lvert T_C \rvert$ (bits)		$min = 273854$ % overhead	
	$\alpha = 2$	$\alpha = 4$	$\alpha = 2$	$\alpha = 4$
cmp	450510	332150	39.21	17.55
set	446528	320231	38.67	14.48
distr	**293851**	**279837**	**6.80**	**2.13**

Table 4.2. Comparison for S1

method and the compressed test set has been performed for frequency ratios equal to 2 ($\alpha = 2$) and 4 ($\alpha = 4$). As illustrated in Section 4.2.1, tailoring the compression method will lead to large test sets especially when the ratio between m_h and α is large. For example, in the case of s35932 for $m_h = 16$ and $\alpha = 2$, tailoring the compressed test set (column 5) will lead to a reduction of 77% when compared to tailoring the compression method (column 4). Similarly, for $\alpha = 4$ and $m_h = 16$ a 64% reduction is obtained, when the two methods are compared. While there are cases when tailoring the compression method leads to smaller test sets than tailoring the test set, the difference is on average of less than 5%.

Distribution architecture The usage of the distribution architecture (*distr*) is illustrated next. Two experimental setups are considered: system **S1** comprising all the circuits from Table 4.1 and **S2** comprising 2x s5378, 2x s9234, 2x s13207 and 2x s38584. The first system (**S1**) has been chosen to illustrate the potential of the proposed distribution architecture to provide test data to all the cores in a SOC and to illustrate the trade-off reduction between the VTD and synchronisation overhead introduced by tailoring the compression method *cmp* and tailoring the compressed test set *set*. The second system (**S2**) has been chosen to illustrate the scalability of the proposed distribution architecture when compared to the interleaving architecture from [122, 134]. Both systems are tested using one ATE channel, with frequency ratios of $\alpha = 2$ and $\alpha = 4$. The results for system **S1** and **S2** are presented in Table 4.2 and 4.3, respectively.

For system **S1**, Table 4.2 illustrates the results obtained using the three methods (*cmp*, *set* and *distr*) for the two frequency ratios. It should be noted that for *cmp* and *set*, for each circuit, the best result reported in Table 4.1 has been chosen in computing the total test set size for **S1**, when **S1** is tested one core at a time and the synchronisation overhead eliminated. For example, for core s5378, for $\alpha = 2$, $cmp = 16554$ and $set = 17668$. As also noted in Section 4.2.1, *cmp* and *set* bring forth a trade-off between reducing the syn-

Method	Tailored $\lvert T_C \rvert$ (bits)		$min = 293182$ % overhead	
	$\alpha = 2$	$\alpha = 4$	$\alpha = 2$	$\alpha = 4$
inter	483162		39.32	
distr	**310105**	**302337**	**5.45**	**3.02**

Table 4.3. Comparison for S2

chronisation overhead and VTD. Therefore, the minimum test set for **S1**, when one core at the time is tested (*min*), and the increase in VTD, are also illustrated in Table 4.2. The minimum test set has been computed by summing up the minimum $|T_E|$ from Table 4.1 column 3, e.g., for core s5378 the chosen test set size has been 12604. For *distr*, the compressed test sets obtained using a group size of 16 has been chosen for each core. It can be observed from Table 4.2 that in order to test **S1**, one core at a time and reduce the synchronisation overhead, tailoring the compression method (*cmp*) obtains larger test sets than tailoring the compressed test set (*set*). For example, *set* obtains a 3% reduction over *cmp* for $\alpha = 4$. The trade-off between the VTD and synchronisation overhead becomes obvious when the values obtained for *cmp* and *set* are compared to the *min* value. For example, there is 39% and 38% increase in VTD when *cmp* and *set* are compared to *min* for $\alpha = 2$. However, employing the distribution architecture (*distr*) the VTD can be reduced. For example, for $\alpha = 2$, 34.77% and 34.19% reductions are obtained when compared to the *cmp* and *set* approaches. Similarly, for $\alpha = 4$, reductions of 15.74% and 12.61% are obtained when compared to the *cmp* and *set* approaches. Note that, the sizes of the composite test sets used in conjunction with the distribution architecture are only 6.80% and 2.13% larger than *min* for the circuits in **S1** for $\alpha = 2$ and $\alpha = 4$ respectively. Thus, employing the distribution architecture there is virtually no trade-off between reducing the synchronisation overhead and the VTD. Since the entire system can be connected to the distribution architecture without any restrictions, the proposed solution can easily fit into a design and test flow without any further modifications to the test and interface equipment, e.g. ATE or DIB. Moreover, as illustrated in Figure 4.12 (see Section 4.2.2), the architecture is driven using only one data input. Hence, the distribution architecture provides an easy integration to the IEEE 1149.1 standard [36] for test data compressed SOC testing.

In the following the second experiment is detailed. As noted in Section 4.1.2, the interleaving architecture proposed in [122, 134] imposes certain constraints on how the cores are connected to the system, i.e., all the cores' test sets must be compressed with

Golomb using the same group size, the number of the decoders driven by the architecture is restricted to the group size and the length of the test sets for each decoder should be approximately equal. To conform with these requirements and to illustrate the scalability of the proposed distribution architecture, a group size of 4 has been considered for the interleaving architecture. Hence, the architecture drives 4 decoders of group size 4. The assignment of the cores to decoders is as follows: 2x s5378 and 2x s9234 to one decoder, 2x s13207 to the second decoder, 1x s38584 to the third decoder and the 1x s38584 to the forth decoder. This is done to obtain approximately equal length test sets for each decoder. For the distribution architecture a group size of 16 has been used for all the cores, as in the previous comparison, since the distribution architecture does not impose any constraints on the decoder corresponding to a core. It is important to note that this experiment shows the scalability of the proposed approach and the benefits derived from it. Also note that when the same compression ratio is obtained by both Golomb and VIHC, and when all the requirements are fulfilled both, the interleaving architecture and the distribution architecture will yield the same TAT. Since, the interleaving architecture [122, 134] is based on the serial Golomb decoder, it does not exploit the frequency ratio. Hence, the size of the test set does not change with the frequency ratio. As illustrated in the table, the distribution architecture ($distr$) yields smaller test sets than the interleaving architecture ($inter$). For example, reductions of 35.81% and 37.42% in VTD are obtained for $\alpha = 2$ and $\alpha = 4$. Hence, when cores come with different decoders and different group sizes, the proposed distribution architecture does not require any modifications to the decoders fully exploiting them. This is contrary to the interleaving architecture which requires that the same group size is used for all the cores, and that the compressed test sets are combined to yield almost equal test sets per decoder. In addition, the CUTs are driven with the on-chip test frequency, contrary to the interleaving architecture which uses the ATE operating frequency.

Therefore, in addition to the features which allow an easy design flow integration, the proposed distribution architecture also reduces volume of test data when compared to previous methods for synchronization overhead reduction. Moreover, because when no the synchronization overhead is reduced, the VTD data on the ATE gives the TAT in ATE clock cycles, the reduction in VTD also leads to reduction in TAT. Furthermore, high-speed test is facilitated due to driving of the cores at the on-chip test frequency.

4.4 Concluding remarks

This chapter has proposed a method to reduce the synchronisation overhead by exploiting the frequency ratio in test data compression environments. The proposed solution comprises two steps, **tailoring the compressed test set** for a given frequency ratio, and the usage of the **distribution architecture** for multiple core test. It has been shown that tailoring the compressed test set obtains similar and smaller test sets than tailoring the compression method without any changes to the on-chip decoder. In addition, the distribution architecture provides a complete system test solution since it is scalable and programmable and it is easy integrable in the design flow since no restrictions are imposed on the system integrator. Therefore, in this chapter, a complete scenario for testing SOCs is provided with low pin count test, low storage requirements and low TAT without any constraints or changes to the test or interface equipment, e.g., ATE or DIB, when single scan chains cores are targeted.

Chapter 5

Test data reduction through elimination of useless test data

In the previous two chapters test data compression and the synchronisation problems raised by test resource partitioning have been addressed. In this chapter a complementary approach to test data compression is proposed which reduces the memory requirements of the ATE through elimination of the *useless test data*. Useless test data is identified as one of the contributors to the total amount of test data, comprising the padding bits necessary to compensate for the difference between the lengths of different chains in multiple scan chains designs. Although *useless test data* does not represent any relevant test information, it is often unavoidable, and it is the cause of the trade-off between the test bus width and the volume of test data in multiple scan chains-based cores. Ultimately this trade-off influences the test access mechanism (TAM) design algorithms leading to solutions that have either short test application time (TAT) or low volume of test data (VTD). Therefore the objective of this chapter is to provide a new test methodology, which, by employing wrapper scan chain (WSC) partitioning in core wrapper design and efficiently exploiting new design for test (DFT) automatic test equipment (ATE) features, is capable of reducing the VTD in core based SOCs.

The remainder of this chapter is organised as follows. Section 5.1 introduces the useless memory problem for multiple scan chain designs, and Section 5.2 illustrates how it scales to core wrapper design. Section 5.3 describes the new test methodology, and Section 5.4 provides an analysis of the effects of WSC partitioning on VTD and TAT. Sections 5.5 and 5.6 provide experimental results and conclusion respectively.

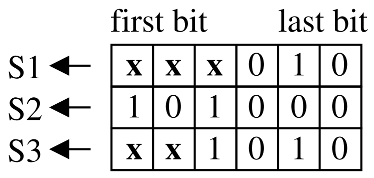


Figure 5.1. Useless memory allocation

5.1 Useless memory allocation (UMA)

Two main factors which drive the cost of testing complex SOCs are the TAT and VTD. Viewed from a core level perspective, a common solution to reduce TAT is to use multiple scan chains-based embedded cores. However, due to various design constraints (e.g., routing overhead and scan path length) the multiple scan chains are not always balanced, i.e., not all the scan chains have equal length. In order to reduce on-chip control when feeding embedded cores' multiple scan chains, the test vectors are augmented with "don't care" bits to account for the differences between the scan chains' lengths. This is exemplified in Figure 5.1, where for three scan chains of lengths 3, 6 and 4 the test tools "pad" the scan patterns, with "don't cares", to make them of equal length[1]. These "don't cares" are shown as **X**s in the figure. Therefore, due to unbalanced scan chains, the *volume of test data* comprises *useful* test data (the scan chain data) and *useless* test data (the padded data). For example when minimum test time is attained for core Module6 of system p93971, from the ITC02 benchmarks [135], the amount of test data is 915.42k, of which useful test data is 517.06k while useless test data is 398.36k. Hence, the useless test data represents 44% of the total amount of test data. Since this useless test data is explicitly allocated within the ATE memory, it will be referred to as useless memory allocation (UMA). The UMA for one test vector represents the number of bits required to make the scan chains of equal length.

As emphasised in Chapter 1, VTD is an emerging concern for testing complex SOCs [4, 9] since it influences directly the ATE memory requirements, and hence, the cost. In this chapter, the VTD is reduced by reducing the UMA. This is achieved by efficiently exploiting the memory management support of the new generation ATEs.

Memory management support comes with ATEs that implement "sequencing-per-pin" [136], i.e., the capability of controlling a pin, or a group of pins, individually. The relevant

[1]It should be noted that, unless stated differently, the term scan chain refers to fixed-length scan chain, i.e., the size of the scan chain cannot be changed by the system integrator

sequencing-per-pin tester's feature is the ability to make a larger number of transfers on a group of pins while others remain unchanged. The minimum number of pins in a group is referred to as pin-group granularity. For example, if a sequencing-per-pin ATE has 64 pins and the pin-group granularity is 32, then it can control separately the number of transfers on two groups of 32 pins. Intuitively, the greater the number of groups, the greater would be the control on the ATE.

While sequencing-per-pin is an expensive extension for functional testers, the recently advocated design for test (DFT) ATEs present the same feature, however, with the advantage of reduced cost [12]. This is because, DFT ATEs do not need all the functional sequencing-per-pin tester's features "behind" each pin. In this paper, this ATE feature is referred to as reconfigurable memory pool (RMP) [12]. Note that this is contrary to conventional ATEs which are capable of performing only sequencing-per-vector – all the ATE channels transfer data at the same rate. With reference to the previous example, a sequencing-per-vector ATE will transfer data on all the 64 pins.

Since the UMA is a result of the unequal length of the scan chains, approaches which aim at balancing the scan chains [85, 137] also reduce the UMA. However, these require internal scan chain modifications and they do not take into account the inputs and the outputs of the core. As the IEEE P1500 standard requires that each core has a wrapper, and, since, depending on the business model, the system integrator is often restrained from modifying the core's internal structure [4], the approaches [85, 137] may not always be applicable in core-based SOC test. This limitation is overcome in this chapter by viewing UMA as a byproduct of core wrapper design. While previous core wrapper design algorithms aimed at minimising the TAT [56], or minimising the test bus width and the TAT [57], in this chapter, core wrapper design is exploited to reduce the UMA. It will be shown how the UMA problem scales from multiple scan chain designs to core wrapper designs, and how the UMA can be reduced in core-based SOCs by efficiently exploiting the memory management capabilities of the new generation ATEs.

5.2 Core wrapper design, UMA and the ATE

This section provides a brief overview of core wrapper design, and illustrates the relationship between WSC partitioning, UMA and the ATE.

Core wrapper design, which is equivalent to constructing WSCs, was shown to be an $NP - hard$ problem [56]. Given a core, the WSCs are composed from the inputs, the outputs and the internal scan chains. An input WSC (WSC^i) refers to the part of the WSC which comprises the core's inputs and the internal scan chains. Similarly, the output WSC (WSC^o) refers to the part of the WSC which consists of the core's outputs and the internal scan chains. Since there exists a one-to-one association between the test bus lines and the WSCs, the TAT of the core will be a function of the WSC's length: $\tau(c) = (1 + max\{wsc^i, wsc^o\}) * n_v + min\{wsc^i, wsc^o\}$ [56], where wsc^i/ wsc^o are the length of the maximum input/ output WSC respectively, and n_v is the number of test vectors in the test set for core c. It should be noted that for the remainder of this chapter it is considered that the TAT is a function of only $max\{wsc^i, wsc^o\}$, since the last member of the above formula has a small influence on the overall TAT.

Exploiting the reconfigurable memory pool (RMP) ATE feature implies dividing the WSCs into disjoint partitions such that the ATE can control the number of transfers on each partition. The number of WSCs in a partition, also referred to throughout the chapter as the partition's cardinality, is a multiple of the pin-group granularity. Due to the one-to-one association between the WSCs and the ATE channels, the number of transfers on each partition (the depth of the corresponding ATE channels) is given by the length of the maximum WSC in the partition, also referred to throughout the chapter as the partition's length. In addition, while the RMP feature allows a different number of transfers on each partition, the greater the number of partitions, the more complex is the control on the ATE. Hence, to efficiently exploit the RMP ATE feature, the depth of the ATE channel, the number of partitions and the pin-group granularity have to be accounted for.

Having illustrated the link between WSC partitioning and the RMP ATE feature, the following section exemplifies the relationship between UMA and core wrapper design. Section 5.2.2 illustrates the control requirements for the ATE when WSC partitioning is considered with the core wrapper design.

5.2.1 UMA and core wrapper design relationship

As noted previously, in order to efficiently exploit the ATE RMP features, the number of partitions and the pin-group granularity have to be accounted for. These two are illustrated using Example 5.1.

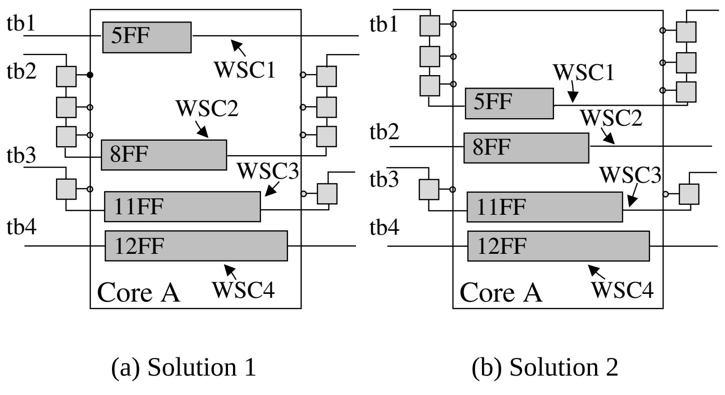


Figure 5.2. Alternative core wrapper design solutions with equal TAT

Example 5.1 For a core with 4 inputs, 4 outputs, and 4 internal scan chains of length 5, 8, 11 and 12 respectively, Figure 5.2 shows two possible core wrappers for a test bus width of 4. Since max$\{wsc^i, wsc^o\}$, for the two core wrapper designs, are equal the two solutions are equivalent with respect to TAT. The four WSCs (WSC_1, WSC_2, WSC_3 and WSC_4) are marked in Figures 5.2(a) and 5.2(b) respectively. The WSCs representation for Figure 5.2(a) is given in Figure 5.3(a), and the corresponding ATE memory bitmap (AMB) is shown in Figure 5.3(b) – AMB_1. The “I”s and “O”s in the WSC representation and AMB, denote the inputs, respectively the outputs of the core. Because the inputs are loaded last, they are shown at the end of the memory bitmap. For the second wrapper design, (see Figure 5.2(b) and its WSC representation in Figure 5.4(a)) the AMB is illustrated in Figure 5.4(b) – AMB_2. The main difference between AMB_1 and AMB_2 is that the latter could be split into a smaller number of partitions with WSCs of equal length, i.e., AMB_1, has 3 partitions: $p_1 = \{WSC_1\}$, $p_2 = \{WSC_2\}$ and $p_3 = \{WSC_3, WSC_4\}$, while AMB_2 has 2 partitions: $p_1 = \{WSC_1, WSC_2\}$ and $p_2 = \{WSC_3, WSC_4\}$. Since conventional ATEs cannot benefit from WSC partitioning, both core wrappers will have the same memory requirements. For a DFT ATE with RMP, assuming a per-pin granularity, i.e., the group of pins has the cardinality of 1, the UMA can be eliminated in both cases. However, the control overhead should not be neglected. Analysing the first core wrapper solution it becomes clear that the control overhead is greater due to the larger number of partitions which have to be controlled. In addition, since the number of WSCs differ from one core to another, the number of obtained partitions varies as well, and thus, the number of parameters required to characterise the AMB shape are core dependent. In contrast, since AMB_2 is shaped such that it can be easily split into two partitions ($p_1 = \{WSC_1, WSC_2\}$ and $p_2 = \{WSC_3, WSC_4\}$) it reduces the number of parameters required to characterise

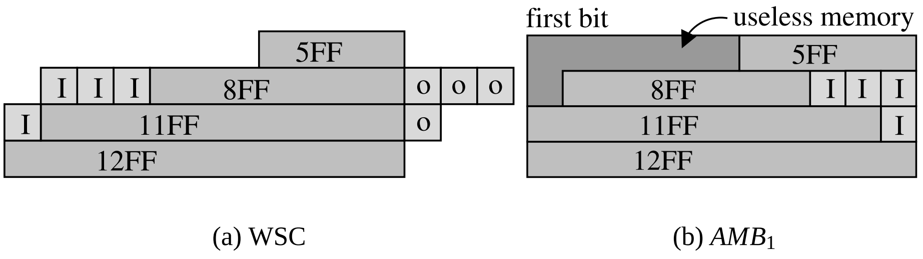


Figure 5.3. WSC and ATE memory bitmap for Solution 1

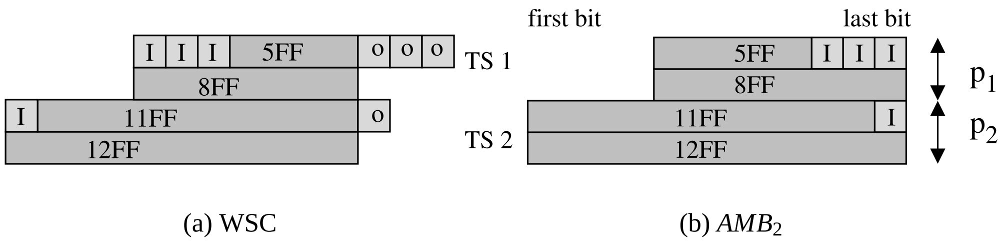


Figure 5.4. WSC and ATE memory bitmap for Solution 2

its shape to 3, i.e., the length of the longest partition, the difference between the length of the two partitions and the cardinality of partition p_1. Furthermore, if the test set is divided into two test sets (TS_1 and TS_2, see Figure 5.4(b)), one per partition, then the memory management is very simple and equally applicable to any AMB which has the shape illustrated in Figure 5.4(b) (see Section 5.2.2). Thus, the sought core wrapper design solution is one which reduces the UMA using a minimum number of partitions. Therefore, it is considered that the first case introduces the useless memory as illustrated in the figure. An additional important point is the fact that the partitions are formed from consecutive WSCs. As illustrated in the above example, $p_1 = \{WSC_1, WSC_2\}$ and $p_2 = \{WSC_3, WSC_4\}$. This is justified as follows. Since the WSCs are connected to the ATE channels when the partitions are not composed out of consecutive WSCs, the ATE will have to provide an interface, which maps the test data stored in the ATE memory to the ATE channels corresponding to the correct WSCs. This, will then lead to a core dependent solution and will require an additional ATE interface. Therefore, only partitions composed out of consecutive WSCs are considered.

As noted in Section 5, the testers can control separately only groups of pins of a given size, e.g., 2, 4, 8, 16. In the best case scenario per-pin granularity is available, however, if this is not the case, the pin-group granularity (g) can also affect the UMA. To illustrate this, consider Figure 5.4(b) where in order to reduce the UMA, two partitions of size 2

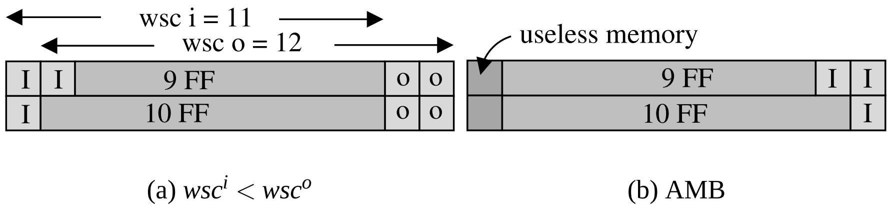


Figure 5.5. WSC and memory bitmap when $wsc^i < wsc^o$

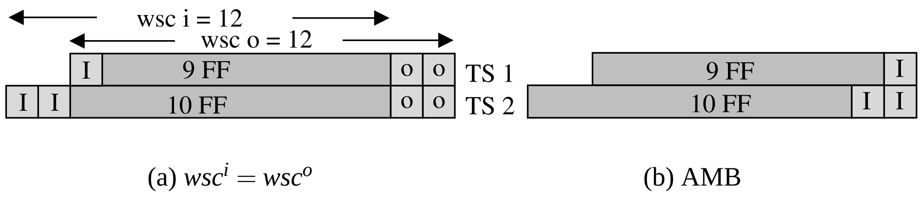


Figure 5.6. WSC and memory bitmap when $wsc^i = wsc^o$

were created. Hence, a tester pin-group granularity smaller than, or equal to, 2 is required. If the tester pin-group granularity is greater (e.g., 4) then no partitioning is possible, hence the memory requirements cannot be reduced. Therefore, with the increase in ATE pin-group granularity, the UMA tends to increase.

It should be noted that, for the wrapper designs discussed in this chapter, all the partitions are loaded in parallel using the same clock and the ATE deploys data on the partitions at different moments. This will be further detailed later on in this section. Hence, if the number of inputs is greater than the number of outputs, then the ATE memory will have to account only for the UMA caused by the input scan chains as explained in Example 5.1 and Figure 5.3. However, if the number of inputs is smaller than the number of outputs, then the wsc^i can be smaller than wsc^o and, in order to ensure that all the data has been shifted out from the output WSCs, the ATE memory has to account for the difference between the wsc^i and wsc^o. This is another source of UMA, as explained in the following example.

Example 5.2 The second source of UMA, caused by the difference in WSC size when $wsc^i < wsc^o$, is illustrated in Figure 5.5. Figure 5.5(a) gives the WSC representation of a core with 3 inputs, 4 outputs and 2 internal scan chains of length 9 and 10 respectively. An optimum WSC design with respect to TAT, leads to $wsc^i = 11$ and $wsc^o = 12$. Since the responses have to be unloaded from the output WSC onto the test bus, the ATE memory

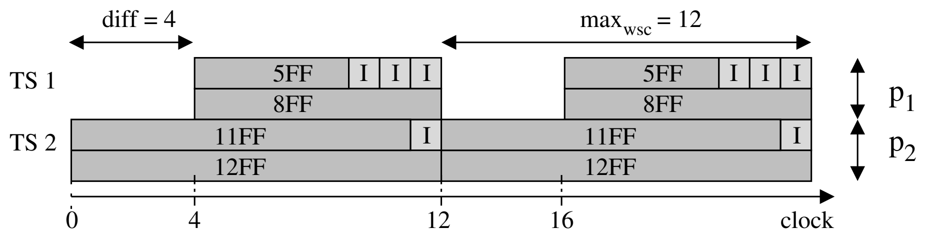


Figure 5.7. ATE test vector deployment information

has to account for the difference $wsc^o - wsc^i$ (Figure 5.5(b)). This problem could be easily solved by using the repeat fill feature of some ATEs and adding special “scan op-codes” to account for the repeat [138]. However, if the number of repeats is not considerable, then adding the extra scan op-codes does not provide a viable solution [138], as it increases the memory requirements instead of reducing them. Furthermore, in order to provide a uniform solution for both cases, when $wsc^i \geq wsc^o$ and $wsc^i < wsc^o$, the WSCs should be constructed such that the shape from the Figure 5.4(b) is obtained. This can be achieved if the inputs are rearranged such that the length of the wsc^i is equal to wsc^o (see Figure 5.6(a) and its WSC configuration in Figure 5.6(b)). In this case, since there is only one test bus that connects the core under test to the TAM and only one clock is used to drive all the WSCs, storing the test data for the longest WSC in the ATE memory will be satisfactory to load/unload the data from all the WSCs in the core under test. Therefore, reducing UMA when the number of core’s outputs is greater than the number of core’s inputs, requires that the number of outputs are taken explicitly into consideration in the design of the input as well as the output WSCs. It should be noted that previous core wrapper design algorithms [56, 57] make a clear distinction between the input and output WSCs design phases. Also note that, since the TAT is a function of $\max\{wsc^i, wsc^o\}$, considering the number of outputs to drive the input WSCs construction will not affect the TAT.

5.2.2 UMA and ATE test vector deployment

The UMA reduction, illustrated with Examples 5.1 and 5.2, is due to partitioning of the WSCs and division of the test set according to the WSC partitions. As the length of the two partitions differ, the ATE will have to account for this difference when deploying the two test sets. This is illustrated next for the test sets given in Figure 5.4(b). Based on the partitions’ length, the intervals at which the ATE deploys the test vectors are shown in Figure 5.7 ($diff = 4$ is the difference between the length of the two partitions and

$max_{wsc} = 12$ is the length of the longest partition). For the first 4 clock cycles, the data is read from test set TS_2 and deployed on partition p_2. For the remaining $max_{wsc} - diff = 8$ clock cycles, data from both TS_1 and TS_2 is loaded onto partitions p_1 and p_2 respectively. It should be noted that having only one clock driving all the WSCs for the first 4 clock cycles the data loaded on the test bus lines corresponding to partition p_1 represents “don’t cares”. This is allowed since valid test data is required at the input WSCs of p_1 only after the $4th$ clock cycle.

Since the core wrapper design is an intermediate step in SOC test, the *proposed approach does not incur any extra overhead*. Hence, the modifications on the ATE are the *only* changes implied by the proposed approach. These can be achieved at the expense of an external module [38] to support custom ATE behaviour employed when IEEE P1500 compliant SOCs are tested.

5.3 Novel test methodology for UMA reduction

It has been exemplified in the previous section that WSC partitioning in conjunction with the ATE deployment procedure lead to UMA reduction. In this section a new test methodology is given which comprises two components: (i) a core wrapper design which accounts for WSC partitioning and considers the number of outputs to drive the WSC’s construction; and (ii) a generic ATE deployment procedure which exploits the features of the core wrapper design ensuring correct test set deployment. The core wrapper design is illustrated in Section 5.3.1 and the ATE deployment procedure is introduced in Section 5.3.2.

5.3.1 Wrapper design algorithm for reducing UMA

Prior to providing the new core wrapper design problem, which accounts for UMA, two recently proposed approaches [56, 57] are analysed. Since the core wrapper design problem was shown to be $NP - hard$, several heuristics have been proposed such as: Largest Processing Time (LPT), MultiFit and Combine in [56], and Best Fit Decreasing (BFD) [57]. Both, the Combine and MultiFit heuristics [56] employ the First Fit Decreasing (FFD) heuristic [56] to assign scan chains to WSCs. The FFD [56] assigns a scan chain to the first WSC which will not lead to an overflow on the maximum WSC capacity. Hence, it tends to unequally distribute the WSCs' lengths, thus leading to UMA. The BFD heuristic [57] aims to equal all the WSC lengths such that the minimum number of WSCs are used, however, since TAT minimisation is the primary design objective, it does not explicitly target the reduction of UMA. For example, when applied to the core considered in Example 5.1, both algorithms lead to the core wrapper design shown in Figure 5.2(a). Hence, these heuristics lead to the UMA marked in Figure 5.3(b) as they do not target minimum WSC partitioning and minimum UMA. There are two interesting conclusions described in [56] and [57]: the minimum TAT for a core is lower bounded by the length of the longest scan chain, and there exists a range of test bus widths for which the TAT does not change. In addition, as shown in the previous section, there are alternative core wrapper design solutions which do not incur any penalties in TAT but which can be exploited to reduce UMA. The new core wrapper design problem, when minimum number of partitions, UMA and TAT are considered, can be formulated as follows:

mUMA *Given a core with n inputs, m outputs, s scan chains* $\{S_1, S_2, \ldots, S_s\}$*, and a test bus width of w, find the minimum number of partitions (*np*) and a wrapper design for the core such that both TAT and UMA are minimised*

It should be noted that the **mUMA** problem is $NP-hard$. This can be easily shown by assuming that the number of partitions equals the test bus width. In this particular case, there is no UMA and the problem reduces to the core wrapper design problem as presented in [56] and [57], which was shown to be $NP-hard$. However, as illustrated in Example 5.1 and as shown later in Section 5.3.2, the number of partitions influences the complexity of the ATE program. Hence, finding the minimum number of partitions is important. Therefore, in the following, a new core wrapper design algorithm is proposed which accounts for the minimum number of WSC partitions, minimum TAT and minimum UMA. In contrast to previous heuristics [56, 57], which always aim at minimising the TAT taking into account only the number of inputs or only the number outputs for WSC construction, in order to reduce the UMA, the proposed algorithm uses the number of outputs to drive the design of both the input and the output WSCs (see Example 5.2 in Section 5.2). The proposed heuristic can be divided into two parts, an algorithm which manages the WSC partitioning, and an algorithm which constructs the WSCs for each partition such that UMA is minimised. As also justified in the previous section (see Example 5.1), only partitions composed from consecutive WSCs are considered.

Algorithm 5.1 - mUMA

INPUT: C,w

OUTPUT : WSC^i, WSC^o

```
for np = 1 to $w$ do
1. while exists partitions $P = \{p_k\}$ do
2.     $WSC$ = mA(S,P)
3.     $WSC^o$ = mA(m,P,WSC)
4.     if $m > n$ then set $cap = max\{WSC^o\}$
5.     else set $cap = 0$
6.     $WSC^i$ = mA(n,P,WSC,cap)
7.     if $uma = 0$ then break
8.     else record uma
   done
```

mUMA heuristic The heuristic illustrated in Algorithm 5.1 gives an iterative solution to the **mUMA** problem. The algorithm firstly generates the WSC partitions and secondly it constructs the core wrapper for the obtained partitions. The algorithm takes as input the core and the test bus width. The partitions P are generated such that $\sum_{k=1}^{np} |p_k| = w$ where $p_k \in P$. For example when $w = 4$, the set of iterated partitions is $\{P\} = \{\{4\},\{1,3\},\{2,2\},\{3,1\},\{1,1,2\},\{1,2,1\},\{2,1,1\},\{1,1,1,1\}\}$. Where the numbers represent the length of the partition. For example, for $np = 3$ and $P = \{1,1,2\}$ there are three partitions of length $|p_1| = 1$, $|p_2| = 1$ and $|p_3| = 2$. As also justified in the previous section (see Example 5.1), for each P, only the case when the partitions are composed out of consecutive WSCs is considered (e.g., $p_1 = \{1,2\}$ is a valid partition, while $p_2 = \{1,3\}$ is an invalid partition). This will reduce the search space and will simplify the ATE test vector deployment procedure, as described in Section 5.3.2. In general for a w, there are 2^{w-1} distinct partitions [139]. For each P, an algorithm to generate the WSCs, called **mA** (Algorithm 5.2), is applied to the internal scan chains (step 2), the outputs (step 3) and the inputs (step 6) of core C. If the number of outputs is greater than the number of inputs, the maximum capacity cap is computed (step 4). This will be used to drive the construction of the input WSCs, hence contributing to UMA reduction as shown in Example 5.2 (see Section 5.2). If the UMA for the newly designed wrapper is 0 (step 7), the algorithm is halted, otherwise, the UMA is recorded. When all partitions from set P for a given np have been processed and no solution with $UMA = 0$ was found, then the number of partitions is increased. The algorithm finishes when $np = w$ or $UMA = 0$. If the algorithm finishes and there is no solution such that UMA is 0, the solution with the minimum UMA and minimum number of partitions is chosen. The UMA is computed using equation (5.3) (see Section 5.4). Considering the ATE pin-group granularity as a constraint in the above

Algorithm 5.2 - mA

INPUT: S, P, WSC_{init}, cap

OUTPUT : WSC

```
sort S descending
set $WSC = WSC_{init}$
for all $S_i \in S$ do
1. set $WSC_{min} = min\{WSC_j\}$
2. set $WSC_{max} = max\{WSC_j\}$
3. for all $p_k \in P$ do
4.     assign $S_i$ to first $WSC_j$ such that $uma(p_k)$ is minimized
5.     if $S_i$ assigned then break
6.     if $WSC_{max} + S_i \leq cap$ then assign $S_i$ to $WSC_{max}$ break
7. done
8. if $S_i$ not assigned then assign $S_i$ to $WSC_{min}$
9. sort WSC
```

algorithm implies filtering the partitions set P such that each partition's length is divisible by the ATE pin-group granularity. Alternatively, one could generate the partitions P such that each partition's length is divisible with g. For the remainder of this chapter paper $mUMA(np)$ will denote the mUMA heuristic when applied for np partitions.

mA heuristic The Minimum Area (**mA**) heuristic used to generate the WSC of the core wrapper is illustrated in Algorithm 5.2. The algorithm assigns the internal scan chain S_i to the *first* wrapper scan chain WSC_j such that the UMA ($uma(p_k)$) on partition p_k is minimum (step 4) without affecting the maximum WSC length. The importance of assigning the scan chain to the first WSC will be detailed in Section 5.4. The UMA on a partition p_k is computed using equation (5.3) see Section 5.4. If there is no such assignment, then if the length of the maximum WSC (WSC_{max}) added with the current scan chain S_i is smaller than the capacity cap (step 6), then scan chain S_i is assigned to WSC_{max}. This ensures that wsc^i and wsc^o will have close to equal length, hence reducing the UMA as illustrated in Example 5.2 (see Section 5.2). It should be noted that this step is performed only for the inputs, when the internal scan chains and the outputs are processed the $cap = 0$ (see step 2 and 3 in Algorithm 5.1). If the scan chain S_i was not assigned to partition p_k then the next partition is chosen. If S_i was not assigned to any partition (step 8), then it is assigned to the WSC with the minimum length (WSC_{min}). After every scan chain is assigned, the WSCs are sorted ascending (step 9). It is important to note that the algorithm aims at generating a WSC representation like the one given in Figure 5.4(b), such that the control overhead on ATE is minimum. While alternative algorithms for designing the core wrapper aiming at minimum UMA can be envisioned, care must be taken to ensure that reducing the trade-off between the UMA and test bus width will not results into a trade-off between UMA and ATE control.

The complexity of Algorithm 5.2 is given by $O(s\ w + s\ w\ log(w))$, i.e., in the worst case scenario there w partitions, and each scan chain has to be assigned to one; in addition the reordering step is performed for each assignment. As illustrated in Algorithm 5.1, the *mA* algorithm is used first for the internal scan chains of the core (step 2), then for the outputs (step 3) and then for the inputs (step 6). The inputs and the outputs are considered as scan chains of length 1. Hence, the complexity of Algorithm 5.1 is given by $O(2^{w-1} \cdot (n+m+s) \cdot w \cdot (1+log(w)))$.

To achieve reduction in memory requirements by exploiting WSC partitioning, ATEs need memory management support. ATE test vector deployment methods which account for this requirement are detailed in the following section.

5.3.2 Test vector deployment procedure for reduced UMA

This section illustrates two possible implementations of the proposed test methodology when different ATE features are considered. Firstly, an ATE test vector deployment procedure is given for the particular case of $np = 2$ partitions, and secondly, the “Split Timing Mode” architecture is examined [140].

In order to fully exploit the new core wrapper design, the initial test set is divided into a number of test sets equal to the number of partitions. The ATE program will deploy test vectors from the different test sets at separate times. Hence, the increase in the number of partitions will lead to a more complex ATE program (see Example 5.1 in Section 5.2). However, if the number of partitions is limited to 2, the changes on the ATE are minor. The pseudo-code for the ATE program for this particular case is discussed next.

Consider two partitions p_1 and p_2 with the maximum WSC length for the two partitions wsc_{p_1} and wsc_{p_2} respectively. Since $|p_1| + |p_2| = w$ (this is how the partitions are constructed, see Section 5.3.1) and $wsc_{p_2} >= wsc_{p_1}$ (the WSCs are ordered ascending after each iteration in the **mA** heuristic, see Algorithm 5.2 – step 9), lets define $max_{wsc} = wsc_{p_2}$ the maximum WSC, $diff = max_{wsc} - wsc_{p_1}$ the difference between the length of the two partitions, and $sp = |p_1|$ the *split point*. Using this information, the initial test set can be divided into two sets. The deployment of test vectors at differ-

Procedure 5.3 - test vector deployment

INPUT: max_{wsc},$diff$,sp,n_v,w,TS_1,TS_2

```
while ($n_v > 0$) do
1. for ($i = 0 \ldots max_{wsc}$) do
2.     if $i > diff$ then
3.         $load[1 \ldots sp] = read\_mem(TS_1, n_v \cdot diff + i, sp)$
4.     $load[sp \ldots w] = read\_mem(TS_2, n_v \cdot max_{wsc} + i, w - sp)$
5. done
6. dec $n_v$
```

ent intervals can be easily achieved by supplying the ATE, in addition to the two test sets (TS_1 and TS_2), with three parameters max_{wsc}, $diff$ and sp. The pseudo-code for a simple ATE procedure which accounts for the mentioned parameters is shown in Procedure 5.3. The procedure takes as inputs the two test sets (TS_1 and TS_2), the width of the test bus (w), the number of patterns (n_v), and the three parameters: max_{wsc}, $diff$, and sp. $read\_mem(TS_i, offset, length)$ denotes a procedure which reads a word of $length$ bits from the test set (TS_i) from the position indicated by $offset$, and $load[a \ldots b]$ denotes the loading of data on the ATE channels between $a \ldots b$. The procedure is detailed next.

For max_{wsc} clock cycles, the test data from TS_2 is loaded onto the test bus. Since the first partition is smaller than the second one, the ATE will read the test data for TS_1 only after $diff$ clock cycles. It should be noted that since all the WSCs are driven by the same clock, the data loaded into the WSCs corresponding to the first partition represents don't cares in the first $diff$ clock cycles. This is allowed since valid test data is required in this partition only after $diff$ clock cycles (see Example 5.1 in Section 5.2). It is important to note that the three parameters suffice to characterize any core wrapper design with **mUMA** for $np = 2$. Hence, the three parameters provide the benefit of independence between the test control and the test data, which is the view put forth by the core test language (CTL) [54] developed in parallel with the IEEE P1500 standard. In Section 5.5 is shown that even though for $np = 2$ the UMA is not always 0, the particular case leads to a good solution from the UMA standpoint, at the benefit of simplifying extra ATE requirements.

In this paragraph an interesting particular case when the *Split Timing Mode* (STM) [140] architecture is available is examined. The STM architecture has been used in [140] for dual-frequency test. The basic idea behind this architecture is to configure a tester as two independent virtual test systems using the same master clock [140], but providing data to the chip under test at two different frequencies. The feature of interest in the investigated scenario is the fact that each virtual test system has its own memory and pattern generator [140]. This feature can be exploited, in the case of the proposed approach, as follows. When the difference ($diff$) between the two partition's length is considerable, the test set corresponding to the shorter partition can be augmented with scan op-codes for repeat fill [138]. These will, then, automatically generate the padded data for the shorter partition. Hence, the test vector deployment procedure is no longer needed since the deployment information is already included within the first test set.

5.4 Analysing wrapper scan chain partitioning trade-offs

Having illustrated the proposed test methodology, in the following a theoretical analysis of WSC partitioning is given and the WSC partitioning, VTD and TAT trade-offs are examined.

5.4.1 Theoretical analysis

Consider that WSC_j represents the length of the WSC corresponding to test bus line j, and w represents the test bus width. Similar to multiple scan chain designs, WSCs also have different lengths, hence, the memory depth of the corresponding ATE channels will also differ. As illustrated in Figure 5.1 (see Section 5.1) for multiple scan chain cores, the UMA for one test vector represents the number of bits required to make the scan chains of equal length. For wrapped cores this translates into:

$$UMA(w) = w \cdot \max_{j=\overline{1,w}} \{WSC_j\} - \sum_{j=1}^{w} WSC_j \tag{5.1}$$

i.e., the number of bits required to equal the WSCs for a given test bus width. Basically, it is the difference between the maximum and minimum memory requirements for a given test bus width.

If, however, the WSCs are partitioned into np disjunctive partitions $\{p_1, p_2, \ldots, p_{np}\}$, the UMA on a partition p_k is given by:

$$uma(p_k) = |p_k| \cdot \max_{j \in p_k} \{WSC_j\} - \sum_{j \in p_k} WSC_j \tag{5.2}$$

i.e., the number of bits required to equal the WSCs from a given partition. Hence, the total UMA is:

$$UMA(w,np) = \sum_{k=1}^{np} uma(p_k) = \sum_{k=1}^{np} |p_k| \cdot \max_{j \in p_k}\{WSC_j\} - \sum_{j=1}^{w} WSC_j \tag{5.3}$$

Starting with an initial ad-hoc partitioning with np partitions, the number of partitions can be further increased through: (i) iterative partitioning – when one of the partitions is further divided; or (ii) repartitioning – when a new partitioning with more partitions, independent of the existing one, is performed. With respect to iterative partitioning, the following lemma holds.

Lemma 5.1 *For a wrapped core of test bus w and np disjunctive partitions such that $\sum_{k=1}^{np} |p_k| = w$, iterative partitioning will reduce the memory requirements.*

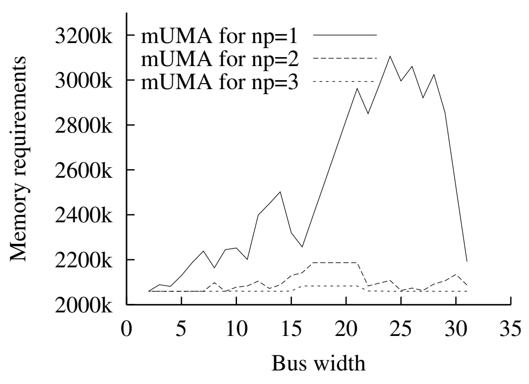
Proof: The proof given below is for the case when one partition is split in two. The general case can be easily derived from it. Hence, for this particular case, and using (5.3), the lemma translates into:

$$uma(p_k) \geq uma(p_k^{'}) + uma(p_k^{''}) \tag{5.4}$$

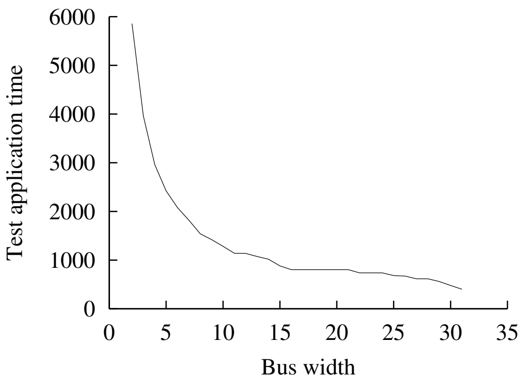
where $p_k = p_k^{'} \cup p_k^{''}$ and $p_k^{'} \cap p_k^{''} = \varnothing$. To prove the above, consider, without loss of generality, that p_k comprises $\{WSC_j, WSC_{j+1}, \dots, WSC_{j+q-1}\}$ and $WSC_j \leq WSC_{j+1} \leq \dots \leq WSC_{j+q-1}$. Consider also that $p_k^{'}$ comprises $\{WSC_j, WSC_{j+1}, \dots, WSC_{j+p'}\}$ and $p_k^{''}$ comprises $\{WSC_{j+p'+1}, WSC_{j+p'+2} \dots, WSC_{j+q-1}\}$. Thus, (5.4) becomes:

$$\begin{aligned} uma(p_k^{'}) + uma(p_k^{''}) &= |p_k^{'}| \cdot WSC_{j+p'} + |p_k^{''}| \cdot WSC_{j+q-1} - \sum_{j \in p_k} WSC_j \qquad (5.5)\\ &\leq (|p_k^{'}| + |p_k^{''}|) \cdot WSC_{j+q-1} - \sum_{j \in p_k} WSC_j \\ &\leq uma(p_k) \end{aligned}$$

The general case, when the number of partitions increases with an arbitrary number (r), can be easily reduced to the above, i.e., $uma(p_k) \geq uma(p_k^{'}) + uma(p_k^{''}) + \dots + uma(p_k^{r})$. $\square$



(a) Memory requirements vs. Bus width



(b) Test application time vs. Bus width

Figure 5.8. Trade-off between test application time and memory requirements for core Module26 of SOC p22810 [135]

Hence, when iterative partitioning is performed, then $UMA(w,np) \geq UMA(w,np+1) \geq UMA(w,np+2) \geq \ldots \geq UMA(w,w) = 0$. Note that the above relation also holds when repartitioning is done such that minimum UMA is obtained on each partition. If repartitioning is done ad-hoc, increasing the number of partitions may not necessarily lead to a reduction in UMA. It should be noted that the proposed **mUMA** heuristic (see Section 5.3.1 performs repartitioning and aims at selecting the solution with minimum UMA. Hence, the above relation holds for the proposed **mUMA** heuristic as also illustrated in the following section.

5.4.2 Volume of test data and test application time trade-offs

As illustrated with equation (5.1) in Section 5.4.1, the memory requirements are dependent on the test bus width. This implies that there is a trade-off between the VTD and test bus width, and consequently, there is a trade-off between the VTD and TAT. These trade-offs are analysed next.

The trade-off between VTD and test bus width is illustrated in Figure 5.8, where the memory requirements for **mUMA** with $np = 1,2$ and 3 (Figure 5.8(a)), and the TAT (Figure 5.8(b)) when the test bus width is varied between 1 and 31 are given for Module26 of SOC p22810 from the ITC02 benchmark circuits [135]. As can be seen in Figure 5.8(a), for $np = 1$ with the increase in test bus width, there is a variation of up to 1000k in the memory requirements. Hence, the trade-off between test bus width and the memory requirements. However, for $np = 2$ the variation is only of 200k. Thus, the inherent trade-off between the VTD and the test bus width is reduced for $np = 2$, and eliminated in most of the cases for $np = 3$, as can be seen in the figure. In order to keep the figure simple, the plot for $np = 4$ is not shown. In this case, however, there is no more trade-off between test bus width and VTD. It can be seen in Figure 5.8(b), that the TAT steadily decreases with the increase in test bus width. However, due to the trade-off between the test bus width and the VTD, for $np = 1$, there is a trade-off between the VTD and the TAT. Since increasing np leads to reducing the trade-off between the test bus width and the VTD, it also leads to reducing the trade-off between the VTD and the TAT. It is interesting to note that the reduction in the variation of VTD is considerable when the number of partitions increases from $np = 1$ to $np = 2$. When $np > 2$, the reduction is small. Therefore, while using the number of partitions as a constraint can diminish the effectiveness of the proposed algorithm, as long as at least two partitions are allowed the UMA reduction can be significant. From the above example, the following can be derived:

Observation 5.1 *Minimising the memory requirements and minimising the TAT can be viewed as orthogonal problems if WSC partitioning is considered with the core wrapper design.*

Thus, if the RMP feature is available, using WSC partitioning in the core wrapper design will allow simultaneous reduction in both TAT and ATE memory requirements. Hence, considering WSC partitioning can also reduce the trade-off between TAT and VTD in TAM designs.

The relation between the TAT obtained using the proposed core wrapper design and the one obtained using the previously proposed BFD [57] core wrapper design is analysed next.

Observation 5.2 *For a given test bus width w, the core wrapper designs obtained with the **mUMA** heuristic for $np = 1$ and $np = w$ are identical with the ones obtained using the BFD [57] heuristic when $n \geq m$.*

This observation is justified by the following. The BFD [57] heuristic tries to equalise the WSCs by assigning a scan chain to the WSC such that the length of the resulting WSC is closest to the maximum WSC length. Hence, it tries to exploit "horizontally" the scan chain to WSC assignment process. This is done to yield a minimum bus width core wrapper[2]. The **mUMA** heuristic (Algorithm 5.1) exploits both “vertically” and “horizontally” the scan chain to WSC assignment process, i.e., it tries to minimise the difference between the maximum and the minimum memory requirements for a partition (see equation (5.2)). As shown in Algorithm 5.2 (step 4), a scan chain is assigned to the *first* WSC such that UMA is minimised without an overrun on the maximum WSC. However, considering only 1 partition the UMA will be the same regardless of the WSC to which the scan chain is assigned. Since after each run the WSC are sorted (see Algorithm 5.2 – step 9), assigning a scan chain to the *first* WSC such that no overrun on the maximum WSC occurs, is equivalent to assigning a scan chain to a WSC such that the length of the resulting WSC is closest to the maximum WSC length. The latter is the strategy used in BFD [57]. Therefore, **mUMA** with $np = 1$ and BFD [57] will generate the same core wrapper. The same reasoning is applicable for $np = w$. In this case, there is no UMA. Hence, assigning a scan chain to the *first* WSC such that no overrun on the maximum WSC occurs will yield the same core wrapper design as the BFD [57] heuristic. Note that when $n < m$ and $np = 1$ or $np = w$, the TAT yielded by **mUMA** will equal the one given by BFD, since the output WSCs are constructed in the same manner.

It is important to note that in general the values for the length of the maximum WSC, which influence directly the TAT of the core, are comparable to the ones obtained by the BFD heuristic [57]. This is because, in both approaches, the scan chains are assigned to WSCs such that the current maximum WSC length is never exceeded. Hence, considering WSC partitioning in the core wrapper design algorithm has small or no penalty in TAT at the great benefit of significant reduction in memory requirements as will be shown in Section 5.5.

5.5 Experimental results

The experimental analysis has been performed on a Pentium II 366 MHz Linux workstation with 128 Mb of RAM using the largest ISCAS89 [132] and ITC02 [135] benchmark circuits. Details about the tools and the benchmark circuits used in the experiments are given in Appendix F. Exploiting wrapper scan chain (WSC) partitioning for reducing useless test data requires ATE with reconfigurable memory pool (RMP). As illustrated with

[2]It should be noted that in some cases, the FFD[57] WSC assignment heuristic obtains the same maximum WSC using a smaller number of test bus lines

Core	n/m	s	FFs	n_v	mem	w_{max}	TAT
s5378	35/49	4	179	97	20758	6	4507
s9234	36/39	4	211	105	25935	5	5723
s13207	62/152	16	638	233	163100	20	9593
s15850	77/150	16	534	94	57434	21	3324
s35932	35/320	32	1728	12	21156	38	714
s38417	28/106	32	1636	68	113152	34	3656
s38584	38/304	32	1426	110	161040	39	5105

Table 5.1. Core specification for ISCAS89 [132] benchmarks

Lemma 5.1, the UMA can be reduced by increasing the number of partitions, however, this will then increase the control overhead on the ATE. In addition, the ATE pin-group granularity may also influence the effectiveness of WSC partitioning. Using the cores' specifications detailed in Section 5.5.1, the above issues are investigated with the following three experiments:

Experiment 1 illustrates the trade-off between ATE control overhead and UMA through a comparison between the general case and the particular case of two partitions, for the **mUMA** algorithm, in Section 5.5.2;

Experiment 2 outlines the effectiveness of the proposed methodology, in terms of memory requirements, when compared to conventional ATEs (with sequencing-per-vector), in Section 5.5.3;

Experiment 3 investigates the influence of the pin-group granularity on the performances of the proposed **mUMA** and the importance of considering WSC partitioning within the core wrapper design algorithm, in Section 5.5.4.

It should be noted that for the first two experiments a per-pin granularity is assumed.

5.5.1 Core specifications

For the ISCAS89 benchmark circuits, the specifications are given in Table 5.1. The table lists the circuit, the number of inputs/ outputs (n/ m), the number of internal scan chains (s), the total number of internal scan cells (FFs), the number of test vectors (n_v) and the minimum memory required to store the test set computed as $mem = (FFs + n) * n_v$. It should be noted that the scan chains have been chosen to be as equal as possible. For example, for core s5378, three scan chains are of length 45, and one is of length 44.

Core	$n/m/q$	s	FFs	n_v	mem	w_{max}	TAT	$w_{max}^{[57]}$	$TAT^{[57]}$
SOC p22810									
Module1	28/56/32	10	1122	785	927870	11	102965	10	102965
Module21	115/76/64	10	1054	465	578925	12	87141	7	88073
Module26	66/33/98	31	11485	181	2108469	32	72981	30	86813
SOC p34392									
Module2	165/263/0	29	8856	514	4636794	30	294064	16	294064
Module10	129/207/0	19	4731	454	2206440	20	236599	10	247974
Module18	175/212/0	14	6555	745	5013850	15	544579	10	544579
SOC p93791									
Module6	417/324/72	46	23789	218	5292604	47	114317	47	114317
Module20	136/12/72	44	7450	210	3185728	46	75893	43	110921
Module27	30/7/72	46	3026	916	2865248	49	63272	46	63272

Table 5.2. Core specification for ITC02 [135] benchmarks

For each circuit, w_{max} represents an upper bound on the test bus width considered in the experiments. In order to ensure that the entire solution space is explored, with respect to core wrapper design, the values for w_{max} have been computed using the formula[3] $w_{max} = \lceil \frac{\max\{n,m\}+\sum_{i=1}^{s} S_i}{1/s \cdot \sum_i^s S_i} \rceil$. It should be noted that w_{max}, as computed above, will guarantee minimum TAT, however, it will not always represent the minimum test bus width for which the minimum TAT is obtained. The TAT given in the table is obtained for w_{max} as computed above.

From the ITC02 benchmark circuits [135] the systems p22810, p34392 and p93791 have been considered. While all the ITC02 benchmark systems have been taken into account in the performed experiments, only these three are reported as they better exemplify the variation in memory requirements. This is mainly due to the large number of scan chains and the scan chain length distribution. It should be noted, however, that the results for the other systems are within the range of the reported results in this section. For each system, the three modules with the largest memory requirements were considered. The specifications for the considered cores are given in Table 5.2, the detailed specification can be found at [135]. In addition to the information given for the cores in Table 5.1, in Table 5.2 the number of bidirectional pins (q) is given as well. It should be noted that, for the core wrapper design, the bidirectional pins (q) were added to both inputs and outputs as suggested in [56]. To illustrate the fact that the w_{max} formula provided in [57] (referred to as $w_{max}^{[57]}$) does not always yield the minimum TAT, the last two columns in Table 5.2 give the maximum test bus width for which the minimum TAT is obtained based on the

[3]It should be noted that the formula given in [57] for w_{max} accounts only for flexible-length scan chains, and when extended to fixed-length scan chains it does not guarantee the upper bound. For further details regarding w_{max} see Appendix D.

Core	w_{max}	**mUMA**				**mUMA** for $np = 2$			
		P	mem	UMA	E_t	P	mem	UMA	E_t
s5378	6	2-4	20758	0	0.01	2-4	20758	0	0.01
s9234	5	1-4	25935	0	0.02	1-4	25935	0	0.02
s13207	20	4-16	163100	0	0.13	4-16	163100	0	0.13
s15850	21	3-1-17	57434	0	0.31	3-18	57528	94	0.17
s35932	38	1-5-32	21156	0	0.85	6-32	21168	12	0.79
s38417	34	2-32	113152	0	0.19	2-32	113152	0	0.19
s38584	39	4-3-32	161040	0	2.01	7-32	161480	440	0.76

Table 5.3. mUMA for w_{max} with ISCAS89 [132] benchmark circuits

formula from [57]. For example, in the case of Module20 from SOC p93791, the formula from [57] yields $w_{max}^{[57]} = 43$, however, the TAT is not minimum for this test bus width, but rather for $w_{max} = 46$. Also, the fact that the w_{max} formula introduced above does not always represents the minimum test bus width can be observed in the table. For example, in the case of Module27 from SOC p93791, the TAT and $TAT^{[57]}$ are equal, however, the $w_{max}^{[57]} = 46$ and the $w_{max} = 49$.

5.5.2 Experiment 1: Trade-off between ATE control and mUMA

As illustrated in Section 5.2, the number of partitions affects the UMA and at the same time influences the control required on the ATE. In this experiment the performances of the **mUMA** heuristic for the general case and the particular case of $np = 2$ partitions are compared.

For the two benchmark sets ISCAS89 (see Table 5.1) and ITC02 (see Table 5.2), the results are reported in Table 5.3 and Table 5.4 respectively. The tables list the length of the partitions, the memory requirements, the UMA, and the execution time (E_t in seconds) needed to complete the **mUMA** algorithm for a test bus width of w_{max}, for both: the general case (columns 3 – 6), and for the particular case with only two partitions (column 7 – 10). It is interesting to note that even though for two partitions the UMA is not zero in all of the cases, it is still very small. For example, in the case of core s38584 (see Table 5.3), the increase in memory requirements is 0.27%, while in the case of Module26 from SOC p22810 (see Table 5.4) the increase in memory requirements is 4.04%. On average, the increase in memory requirements for the particular case of $np = 2$ is of less than 5%. This justifies the usage of the proposed heuristic for the particular case with two partitions, since minimum or close to minimum memory requirements are obtained with minor changes on the ATE (see Section 5.3.2). The execution time (E_t) is insignificant,

Core	w_{max}	mUMA				mUMA for $np = 2$			
		P	mem	UMA	E_t	P	mem	UMA	E_t
SOC p22810									
Module1	11	1-6-4	927870	0	0.04	10-1	973400	45530	0.04
Module21	12	1-1-7-3	573345	0	0.09	11-1	638910	65565	0.05
Module26	32	1-3-6-22	2108469	0	0.86	10-22	2197340	88871	0.11
SOC p34392									
Module2	30	7-8-15	4636794	0	0.44	14-16	4687680	50886	0.42
Module10	20	9-3-5-3	2206440	0	1.23	9-11	2591886	385446	0.20
Module18	15	1-3-6-2-1-1-1	5013850	0	4.13	5-10	5431050	417200	0.08
SOC p93791									
Module6	47	1-46	5292604	0	0.09	1-46	5292604	0	0.09
Module20	46	1-6-16-23	3185728	0	2.57	8-38	3303872	118144	0.35
Module27	49	3-46	2865248	0	0.08	3-46	2865248	0	0.08

Table 5.4. mUMA for w_{max} with ITC02 [135] benchmark circuits

e.g., for the general case it is up to 4 seconds and for the particular case of two partitions it is under 1 second. Having shown that the particular case of two partitions yields minimum or close to minimum UMA, for the remainder of the experiments this particular case will be considered for further comparisons. In the following the overall performance of the proposed test methodology is compared with the case when a conventional ATE is used.

5.5.3 Experiment 2: UMA($np = 2$) vs. conventional ATE

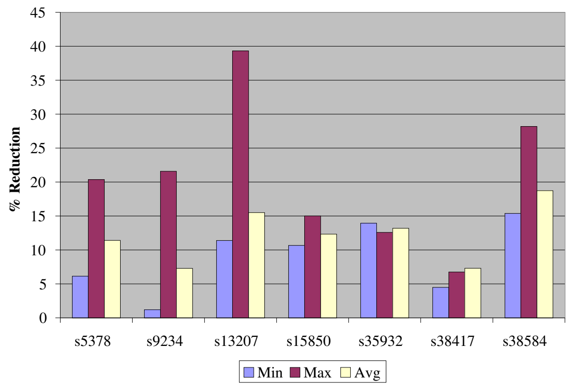
Although based on Lemma 5.1 (see Section 5.4.1), the VTD can be reduced when partitioning is considered, it is interesting to know how much reduction can be obtained. For this purpose a conventional ATE, and an ATE with RMP are considered. For the former, two core wrapper designs (First Fit Decreasing (FFD) [56], Best Fit Decreasing (BFD) [57]) have been used, while for the latter the $mUMA(2)$ has been employed. To provide a common ground for the comparison it has been imposed that for all the cases the TAT is the same and the test bus has been varied between 4 and w_{max}. As noted in Observation 5.2, BFD [57] and $mUMA(1)$ obtain the same TAT. In addition, for the performed experiments, it was found that the TAT obtained using $mUMA(2)$ will equal the TAT obtained using BFD. This can be explained by the fact that both approaches assign the scan chains to WSCs such that the current maximum WSC length is never exceeded. Therefore, there are no TAT penalties when compared to [57]. To also ensure that the TAT obtained with FFD [56] equals to the one obtained with BFD, the FFD algorithm was used considering the capacity given by the maximum WSC determined with BFD. It should be noted that, although this might give the impression of a disadvantage with

Core	FFD [56]			BFD [57]			**mUMA** for $np = 2$		
	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg
s5378	22116	26190	23457	22116	26190	23457	**20758**	**20855**	**20782**
s9234	26250	33075	27972	26250	33075	27972	**25935**	**25935**	**25935**
s13207	184070	272610	*193367*	184070	272610	198693	**163100**	**165430**	**163394**
s15850	64296	*68244*	*65638*	64296	93060	68593	**57434**	**57998**	**57552**
s35932	24576	*25920*	*24728*	24576	40176	27542	**21156**	**22656**	**21470**
s38417	118456	*124848*	*122675*	118456	215016	142889	**113152**	**116416**	**113714**
s38584	190300	*242000*	*201674*	190300	300080	209569	**161040**	**173800**	**163891**

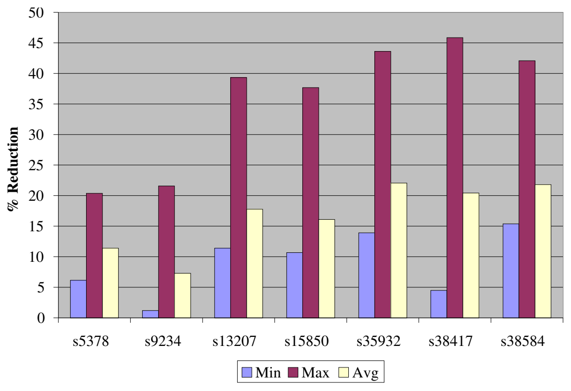
Table 5.5. Memory requirements comparison for ISCAS89 benchmark circuits [132] using mUMA for $np = 2$

respect to [56], it will actually lead to reduction in memory requirements when employing the FFD heuristics and comparing it to BFD. This is because, in some cases the BFD heuristic requires more WSCs to obtain the same TAT as the FFD heuristic. Hence, discarding the empty WSCs for the core wrapper design produced by FFD, will reduce the memory requirements. It is important to note that, due to the variation of w between 4 and w_{max}, the entire core wrapper design solution space is explored and therefore the TAT can be considered as a reference point in the comparison.

As illustrated in Section 5.4.2 for different test bus widths there are different memory requirements. Therefore, the three core wrapper designs have been employed for $w = \overline{4, w_{max}}$, and their minimum (*Min*), maximum (*Max*) and average (*Avg*) memory requirements over all TAM widths have been computed. The results are reported for the three core wrapper design methods in the case of ISCAS89 benchmarks circuits [132] in Table 5.5, and in the case of ITC02 benchmarks circuits [135] in Table 5.6. In the case of the ISCAS89 benchmark circuits, for the FFD and BFD approaches, the minimum (*Min*), the maximum (*Max*) and the average (*Avg*) memory requirements over all test bus widths are given in columns 2 – 4 and 5 – 7 in Table 5.5 respectively. The results for the proposed **mUMA** for $np = 2$ are reported in columns 8 – 10 in the same table. Note that the difference between minimum (*Min*) and maximum (*Max*) memory requirements is considerably greater in the case of the FFD and BFD methods than in the case of the proposed core wrapper design algorithm. For example, for core s13207, in the case of both FFD and BFD, the maximum memory requirements are 32.47% greater than the minimum memory requirements, hence, the trade-off between VTD and test bus width. This is contrary to the proposed approach where the increase is only 1.42%, which leads to trade-off reduction. The reduction in minimum, maximum and average memory requirements over the



(a) % Reduction over FFD [56]



(b) % Reduction over BFD [57]

Figure 5.9. Memory requirements reduction for ISCAS89 benchmark circuits [132] using mUMA for $np = 2$

two previous approaches, FFD and BFD, are given in Figures 5.9(a) and 5.9(b) respectively. For example, in the case of circuit s38584, the maximum memory requirement is reduced by 28.18% when compared to FFD and by 42.08% when compared to BFD. The average memory requirement for s38584 is reduced by 18.73% when compared to FFD and 21.80% when compared to BFD. Overall, the proposed test methodology achieves average and maximum memory requirement reduction of up to 22.05% and 45.86% respectively.

For the ITC02 benchmarks circuits [135], the results are reported in Table 5.6. Once again, note the difference between the minimum and maximum memory requirements in the case of the FFD and BFD methods. For example, for core Module20 from SOC p93691, in the case of FFD (columns 2 – 4 in Table 5.6), the maximum memory requirements are 36.30% greater than the minimum memory requirements. Similarly, for BFD (columns 5 – 7 in Table 5.6), the maximum memory requirements are 36.68% greater than the minimum memory requirements. Hence, the trade-off between the test bus width and the memory requirements. For the proposed **mUMA** with $np = 2$, the variation in memory requirements for the considered core are of less than 4%. Hence, the trade-off between the test bus width and the memory requirements is considerably reduced. Finally, the reductions in minimum, maximum and average memory requirements when compared to the FFD [56] and BFD [57] approaches are given in Figures 5.10(a) and 5.10(b) respectively. It can be observed that the reductions, in maximum and average

Core	FFD[56]			BFD[57]			**mUMA for** $np = 2$		
	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg
SOC p22810									
Module1	928	*1181*	*1056*	928	1518	1124	**906**	**951**	**917**
Module21	560	*760*	*663*	560	1014	714	**560**	**624**	**568**
Module26	2081	*3061*	*2417*	2081	3105	2523	**2059**	**2186**	**2104**
SOC p34392									
Module2	4578	*7955*	*4947*	4578	8583	6142	**4528**	**4578**	**4537**
Module10	2189	*2531*	*2441*	2189	4602	3016	**2155**	**2531**	**2335**
Module18	4952	*5454*	*5207*	4952	7956	5819	**4896**	**5304**	**5056**
SOC p93791									
Module6	5171	*9154*	*6016*	5171	9580	6637	**5169**	**5178**	**5170**
Module20	3111	*4885*	*3673*	3111	4914	3806	**3111**	**3226**	**3125**
Module27	2798	*4544*	*3308*	2798	4675	3442	**2798**	**2816**	**2802**

The memory requirements are given in Kilobytes.

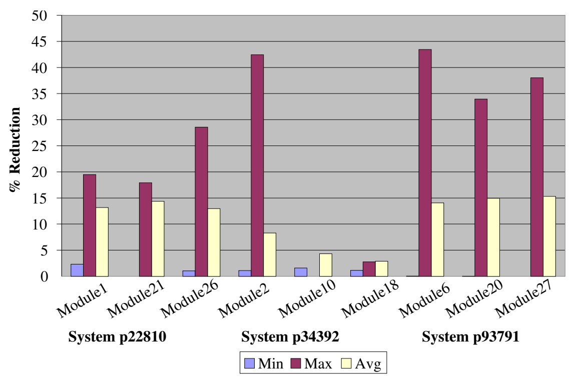
Table 5.6. Memory requirements comparison for ITC02 benchmarks circuits [135] using mUMA for $np = 2$

memory requirements, are considerable. For example, in the case of core Module27 from SOC p93691, the maximum memory requirements are reduced by 38.03% and 39.76% when compared to the two previous approaches (FFD and BFD) The reduction in average memory requirements over all test bus widths is 15.31% and 18.60% when compared to the FFD and BFD heuristics. Overall, the reduction in maximum memory requirements is up to 46.67%, while the reduction in average memory requirements is up to 26.13%.

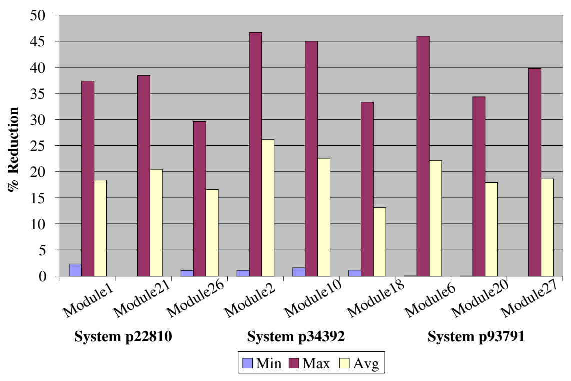
Based on the above results, it can be clearly seen that, considering WSC partitioning in the core wrapper design process reduces the trade-off between test bus width and memory requirements and consequently, as also illustrated in Section 5.4.2, between memory requirements and TAT.

5.5.4 Experiment 3: ATE pin-group granularity constrained WSC partitioning

In this section two issues are investigated. Firstly, the implications of the pin-group granularity on the performances of the proposed **mUMA**, and secondly, the importance of considering WSC partitioning within the core wrapper design algorithm. It is important to note that in the framework of the proposed test methodology, WSC partitioning has been considered as a step within the core wrapper design algorithm. However, WSC partitioning could also be seen as a post processing step. To provide a compar-



(a) % Reduction over FFD [56]

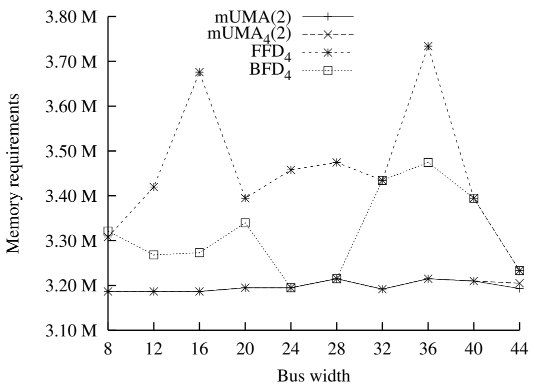


(b) % Reduction over BFD [57]

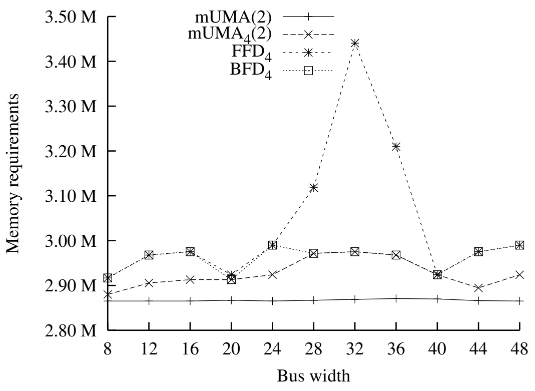
Figure 5.10. Memory requirements reduction for ITC02 benchmark circuits [135] using mUMA for $np = 2$

ison for these two cases, WSC partitioning has been considered as a post processing step for the FFD [56] and the BFD [57] heuristics. WSC partitioning has been implemented on top of the experimental setup illustrated in Section 5.5.3 as follows. For a given test bus (w), granularity (g) and $np = 2$, there are $(\frac{w}{g} - 1)$ possible solutions to the partitioning problem, e.g., when $w = 12$ and $g = 4$ there are $3 - 1 = 2$ possible solutions: $sol_1 = \{p_1 = \{1,2,3,4\}, p_2 = \{5,6,7,8,9,10,11,12\}\}$ and $sol_2 = \{p_1 = \{1,2,3,4,5,6,7,8\}, p_2 = \{9,10,11,12\}\}$, where the numbers enclosed within brackets represent test bus lines. For all these solutions the UMA has been computed and the one with the minimum UMA has been chosen. The UMA values for FFD and BFD when the post processing step is considered, for a granularity g, are denoted by FFD_g and BFD_g respectively. The UMA values for the **mUMA** heuristic when applied for $np = 2$ and a granularity g are denoted by $mUMA_g(2)$.

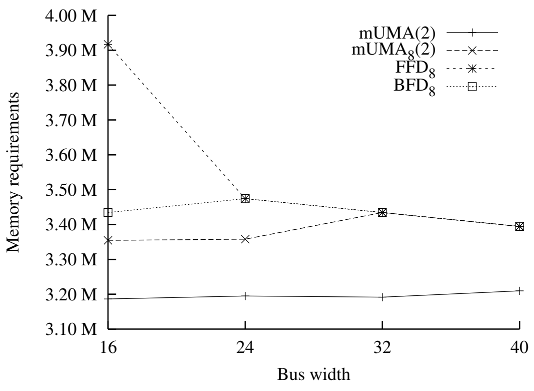
To summarise, for an ATE with a given granularity, for the same test bus width and the same TAT a comparison between $mUMA_g(2)$, FFD_g and BFD_g has been performed. Due to the large amount of results, in the following for $g = 4$ and $g = 8$, two cores, Module20 and Module27 from SOC p93791 [135] are considered. For the two cores, four core wrapper design scenarios, $mUMA(2)$, $mUMA_g(2)$, FFD_g and BFD_g, have been performed using test bus widths $w = \overline{4, w_{max}}$. It should be noted that only test bus widths where w is divisible with g have been used. The results for $g = 4$ are plotted in Figures 5.11(a) and 5.11(b), for the two cores. For $g = 8$ the results are plotted in Figures 5.11(c) and 5.11(d). Additional results are provided in Appendix D.



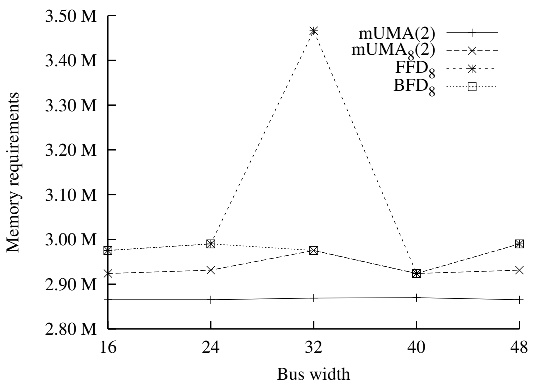
(a) Module20 of SOC p93791 [135] with $g = 4$



(b) Module 27 of SOC p93791 [135] with $g = 4$



(c) Module20 of SOC p93791 [135] with $g = 8$



(d) Module 27 of SOC p93791 [135] with $g = 8$

Figure 5.11. ATE pin-group granularity and WSC partitioning

With respect to the first issue, the influence of the ATE pin-group granularity on the performances of the **mUMA** algorithm, it can be seen in the figures that the influences are small. For example, in Figure 5.11(a) for Module20 and $g = 4$, $mUMA(2)$ and $mUMA_4(2)$ have the same performance for test bus widths up to $w = 40$. In Figure 5.11(b), for Module27, it can be observed that the difference between $mUMA(2)$ and $mUMA_4(2)$ is less than 100k. Similarly in Figure 5.11(d) for $g = 8$. For Module20 with $g = 8$ in Figure 5.11(c), the difference between the $mUMA(2)$ and $mUMA_8(2)$ is up to 200k.

Having illustrated the influence of ATE pin-group granularity on the proposed algorithm's performances, in the following the difference between considering WSC partitioning as a post processing step and as a step within the core wrapper design is illustrated. As noted previously, for this purpose, the FFD and BFD core wrapper algorithms have been extended with a post processing WSC partitioning step. Throughout the performed experiments it has been observed that considering WSC partitioning as a post processing step yields memory requirements which are lower bounded by the ones obtained using WSC partitioning within the core wrapper design (**mUMA**). This is best illustrated with Figure 5.11(a) for Module20 when $g = 4$. It can be seen in the figure that while $mUMA_4(2)$ has almost constant memory requirements, the memory requirements for FFD_4 and BFD_4 vary considerably, and they are almost always greater than $mUMA_4(2)$. Similar behaviour can be observed in Figure 5.11(b) where, while BFD_4 tends to be closer to $mUMA_4(2)$, FFD_4 has greater memory requirements for w from 26 to 36. When $g = 8$, the performances of the algorithms tend to become similar, however, $mUMA_g(2)$ obtains always the lowest memory requirements among the three ATE-constrained core wrapper designs.

5.6 Concluding remarks

This chapter analysed the test memory requirements for core based SOCs and identified *useless memory* as the source for the trade-off between test bus width and volume of test data in multiple scan chains-based cores. A new test methodology has been proposed, which based on employing wrapper scan chain partitioning in core based designs and exploiting ATE memory management features can obtain considerable reduction in useless memory. Extensive experimental analysis, on the ISCAS89 and ITC02 benchmark circuits, has been conducted to evaluate the proposed methodology. Thus, the work presented in this chapter demonstrates that with the advent of the new generation ATEs, which allow greater flexibility and provide memory management capabilities, methodologies complementary to test data compression can be used to reduce the volume of test data, and hence the cost of testing complex SOCs.

Chapter 6

Integrated test solution for core-based System-on-a-Chip

In the previous three chapters the volume of test data and the synchronisation overhead have been addressed. This chapter provides an integrated test solution for core-based SOCs which combines the approaches proposed in Chapters 3, 4 and 5. This is achieved by an integrated test data compression and core wrapper design methodology. The compressed test data can be transferred from the automatic test equipment (ATE) to the on-chip decompression architecture using only *one* test pin, thus providing an efficient reduced pin count test (RPCT) methodology for multiple scan chains-based embedded cores. In addition to reducing the volume of test data, the proposed solution decreases the control overhead, test application time and power dissipation during scan. Since it is based on the distribution architecture provided in Chapter 4 (see Figure 4.12) it also eliminates the synchronisation overhead between the ATE and the SOC. Moreover, it is scalable and programmable and, since it can be considered as an *add-on* to a test access mechanism (TAM) of a given width, it provides seamless integration with any design flow. Thus, the integrated core wrapper design and test data compression solution is an efficient low-cost test methodology for SOCs.

The remainder of this chapter is organised as follows. The following section gives preliminaries for this chapter and outlines the motivation behind this work. Section 6.2 introduces the new test data decompression architecture, outlines its features and illustrates how the proposed methodology can be easily integrated into the design flow. Section 6.3 and 6.4 give experimental results and conclusions respectively.

6.1 Preliminaries and motivation

As emphasised in Chapter 1, the cost of test is related to the volume of test data (VTD) [9], the test application time (TAT) [14], the channel capacity [15], the cost of automatic test equipment (ATE) [13], and, in core based SOCs, core redesign [4]. In addition, with the large number of gates and high chip frequency, high power dissipation during test can also considerably affect yield [141, 142], and thus cost.

It has been illustrated in Chapter 4, that test data compression (TDC) can reduce the VTD, the TAT and the number of pins required for testing, without imposing any constraints on the ATE and using only one test pin. While this yields an effective solution when single scan chain cores are comprising the chip, for cores with multiple scan chains (MSCs) the solution has limited applicability due to its increased area overhead. In this chapter, a TDC solution is proposed which, in addition to reducing the VTD, TAT and number of test pins, with no synchronisation overhead, can also reduce power dissipation when cores with MSCs are targeted.

The remainder of this section is organised as follows. Firstly, in Section 6.1.1, power dissipation during scan is introduced. Secondly, in Section 6.1.2, TDC approaches which target cores with MSCs are analysed with respect to the compression ratio, area overhead, test application time and power dissipation. Finally, in Section 6.1.3, the main idea behind the proposed test data compression architecture is outlined.

6.1.1 Power dissipation during scan

While scan based design is the foundation on which most design-for-test (DFT) techniques are based, one major disadvantage of this design technique is that with the increase in scan chain length and frequency ratio, the power dissipation during scan becomes a problem [141, 142, 4, 38, 143]. This is due to the switching activity incurred while loading the test data into the scan chain (see Appendix A). To reduce power dissipation during scan, test set dependent and test set independent approaches have been proposed. The former derive methods for reducing the scan power dissipation based on the properties of the test set [142]. The latter guarantees power reduction by using small frequencies while shifting and reducing the number of active scan cells [143, 144] in the core.

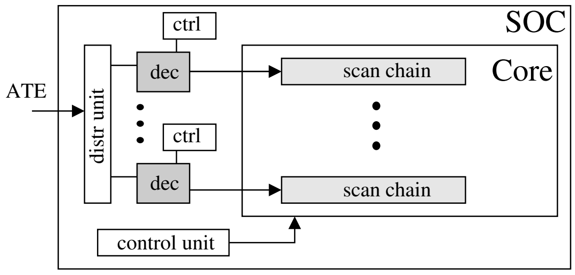


Figure 6.1. Extending single scan chain approaches to MSC test

With respect to TDC, reducing scan power has been addressed only from a test set dependent perspective. For example, the approaches in [145, 146] considered power dissipation as a parameter of the test set, hence, they manipulate the test set such that the power dissipation is reduced. This is possible due to the "don't care" bits within the test set which can be mapped to ensure minimum switching activity within the scan chain. Because switching activity in the scan chain can be correlated with the power dissipation of the chip [147], reducing the switching activity also reduces the chip power dissipated during scan. While the two methods [145, 146] can reduce the scan power during scan-in, if scan-out power is also considered, then scan latch reordering (SLR) has to be performed [148] in order to contain the total power dissipation. Since, SLR requires knowledge about the core's internal structure, it cannot be performed by the system integrator, and hence it implies core redesign. In the considered core based environment core redesign leads to cost increase. Hence, a solution which guarantees total power reduction during scan from the system integrator's perspective has to be non-intrusive and test set independent. As noted above, such a solution implies the usage of a slow clock or the exploitation of scan chain partitioning [143, 149, 150]. Since the architecture proposed in this chapter uses wrapper scan chain selection, it has an added benefit of reducing the power dissipation during scan. This will be further detailed in Section 6.2.

6.1.2 TDC for multiple scan chains cores

Testing multiple scan chains cores using TDC methods, can be performed by either (a) extending single scan chain based methods [119, 121, 122, 123] to multiple scan chains, or (b) using methods specifically proposed for multiple scan chains designs [116, 114, 107]. In the following these two categories are analysed.

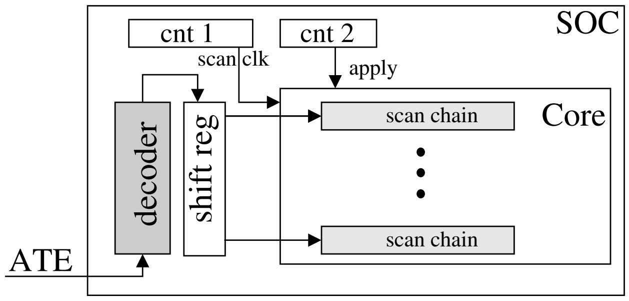


Figure 6.2. Extending FDR to MSC test [151]

Extending single scan chain approaches (a) to multiple scan chains (MSC) designs can be done using one decoder per scan chain. As already illustrated in Chapter 4 for multiple core test, if the distribution architecture is employed, the VTD, the TAT and the synchronisation overhead can be reduced at the expense of a simple on-chip unit. When extended to cores with MSCs, the architecture will resemble the one in Figure 6.1. Firstly, due to the employment of the distribution architecture not all decoders will be active at the same time, and therefore, not all the scan chains will be active at the same time. Thus, the switching sources in the circuit are reduced, and therefore also the power dissipation. Despite this advantage, the applicability of this method to MSCs cores is somewhat limited due to the large area overhead and excessive control required by the decoders. This control is caused by the fact that the scan chains are fed independently (hence, for each scan chain one counter is required to ensure that the entire scan chain is filled); and due to differences between test data in different scan chains, the rate at which each scan chain is filled differs (hence, a global control unit would be required to apply the test vector to the MSC core). Hence, while this approach has the potential to reducing the VTD and TAT (as illustrated in Chapter 4) and, as also noted above, the power dissipation during scan, it has the disadvantage of considerable area overhead.

To somewhat mitigate the area overhead problem, in [151] an alternative architecture has been proposed (see Figure 6.2), which instead of using one decoder per scan chain, employs a shift register of the length equal to the number of scan chains. This shift register is loaded by the decoder, and when the shift register is full its content is loaded into the scan chains. This process is repeated until the test vector has been loaded. To account for the above, two counters are also required. Hence, this approach has smaller area overhead than using one decoder per scan chain, and it can also reduce the average scan power dissipation. However, the disadvantage of this approach is that it destroys the

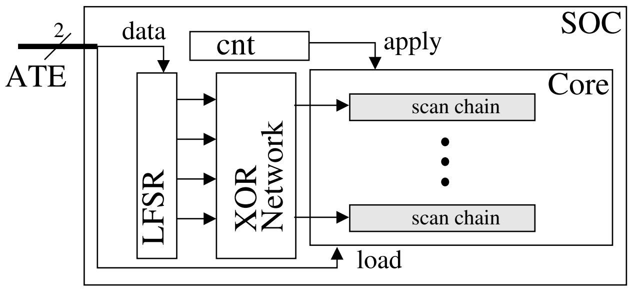


Figure 6.3. MSC test with SmartBIST [114]

properties exploited by compression methods based on variable-length input patterns: the runs of '0's. This is because, in order to ensure that the correct values are shifted into the scan chains, the test set is rotated, i.e., the rows in the initial test set become columns, with each column corresponding to a scan chain. To illustrate the above, the method has been applied to the s38417 ISCAS89 [132] benchmark circuit using the FDR [123] coding method for 8 and 16 scan chains. The results show that the volume of test data is actually *increasing* by 18.89% and 21.15% respectively when compared to compressing the initial test set for the single scan chain core. Hence, the architecture does not provide a viable solution for MSC cores.

Methods which are specifically proposed for MSCs cores exploit the spareness of care bits in the test set [116, 114, 107], i.e., the fact that the number of care bits within the test is small, usually $< 10\%$. This allows the usage of an on-chip decompression architecture as illustrated in Figure 6.3, i.e., the use of a Linear Feedback Shift Register (LFSR) and an *XOR Network*. Since it is based on an architecture with linear properties, one disadvantage of this scheme is that it can run into linear dependencies[1], i.e., the architecture is not capable of generating the required output at any given time. Hence, while attaining TAT and VTD reduction, due to linear dependencies the scan chain load has to be interrupted. To account for this case, a special gating signal is required (*load* in Figure 6.3). While these methods lead to small area overhead, VTD and TAT, they encounter limitation when applied to core based SOCs. This is because they are embedded into automatic test pattern generation (ATPG) programs in order to guarantee the small number of care bits. ATPG requires information about the core structures, which may not be always available in core based SOCs. In addition to the above, due to driving all the scan chains at a time, these schemes may also lead to high scan power dissipation even though they are using the slow external ATE clock to drive the on-chip scan chains.

[1]For further details on linear dependencies the reader is referred to Appendix E

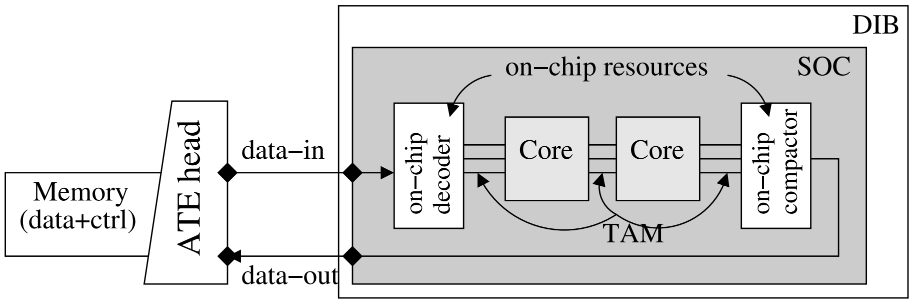


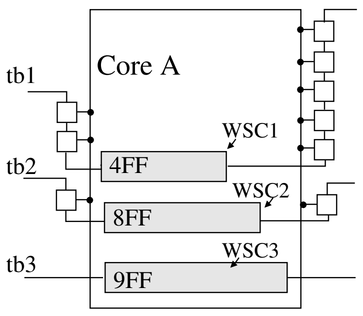
Figure 6.4. Low-cost core based SOC test

To summarise, the MSC decompression architectures [151, 116, 114, 107] drive *all* the scan chains in a clock cycle. Hence, high switching activity is caused during scan, which can lead to the destruction of the chip [143, 4]. In addition, with the emerging IEEE P1500 [51] standard, solutions for core based test should account for the core wrapper as well. Therefore, the aim of this chapter is to propose an integrated methodology which can be seamlessly incorporated into the design flow, based on extending single scan chain TDC method to multiple scan chains.

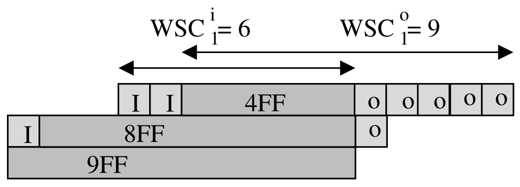
Using one decoder per scan chain has the benefit of reducing the power dissipation, however, at the expense of high area overhead. It has been shown in Chapter 5 that the **mUMA** heuristic can generate core wrapper designs which can be easily characterised using a small number of parameters. In addition, the distribution architecture proposed in Chapter 4 is generic enough to be extended for different needs. Based on these two, the architecture proposed in this chapter will resemble the one in Figure 6.4. It can be seen in the figure that the synchronisation overhead is eliminated and the architecture is an add-on to a given TAM. How, core wrapper design (from Chapter 5) and the distribution architecture (from Chapter 4) can be combined is illustrated next.

6.1.3 Exploiting core wrapper design for control overhead reduction

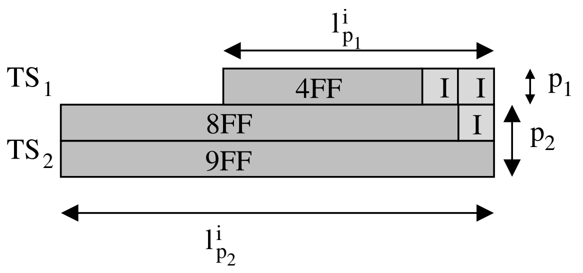
It has been shown in Chapter 5 that using partitioning in core wrapper design reduces the amount of useless test data. In addition, using the **mUMA** core wrapper design heuristic (see Section 5.3.1 on Page 95) the core wrapper can be characterised using a small a number of parameters. Furthermore, for a given number of partitions and a given test bus width no other parameters are necessary to characterise the core wrapper design. This is exemplified next.



(a) Core and its WSCs



(b) WSC representation



(c) Test set division and core wrapper characterisation

Figure 6.5. Core wrapper characterisation

Example 6.1 Consider a core with 3 inputs, 6 outputs and 3 scan chains of length 4, 8 and 9 flip-flops (FFs). A core wrapper designed using the **mUMA** heuristic, proposed in Chapter 5, is given in Figure 6.5(a) for a test bus width of $w = 3$. The wrapper scan chains (WSCs) are marked in the figure, and the WSC representation is illustrated in Figure 6.5(b), with the input and output WSCs (WSC_1^i and WSC_1^o) marked for WSC_1. As noted in Section 5.2 (see Page 89), the length of the partition is given by the longest input WSC, and the height of the partition (its cardinality) is given by the number of WSCs in the partition. For the considered example, the core wrapper can be characterised using the cardinality of the two partitions ($p_1 = 1$ and $p_2 = 2$), and the length of the two partitions ($l_{p_1}^i = 6$ and $l_{p_2}^i = 9$) as shown in Figure 6.5(c). The i from the previous notation stands for the input WSC. Corresponding to the two partitions, the test set is divided as shown in Figure 6.5(c) into TS_1 and TS_2. While in Chapter 5, WSC partitioning and the above parameters have been used to provide an ATE test set deployment procedure, in this chapter they will be used to reduce the control overhead when extending single scan chains TDC methods to MSCs, as follows. Firstly, in order to reduce the high number of decoders required when extending single scan chain approaches to MSCs, instead of

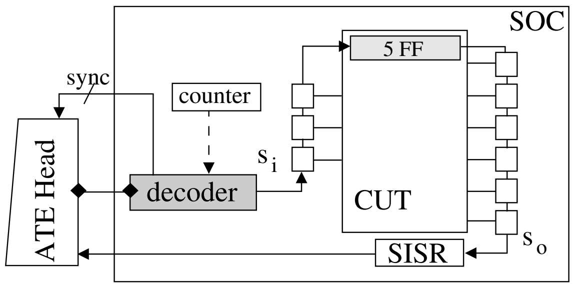


Figure 6.6. Single scan chain TDC

using one decoder per WSC, one decoder per partition is used. Secondly, only the necessary control imposed by the parameters mentioned above will be added for the introduced decoders as detailed next. Since the WSCs in one partition have near equal length, *one* counter can be used to ensure that each WSC in one partition is fully loaded. In addition, to account for the number of WSCs in one partition again *one* counter is required. Therefore, ensuring the correct load of all the WSCs in one partition *two* counters are needed. In the considered example, the number of outputs are greater than the number of inputs. This, as also illustrated in Section 5.2.1 (see Example 5.2 on Page 93), leads to having the output WSC's length greater then the input WSC's one. For the considered example, $WSC_1^o = 9$ and $WSC_1^i = 6$. Since, WSCs have to be, not only correctly loaded, but also correctly unloaded, one more counter is required to ensure that the data is unloaded from WSC^o before new data is loaded. Therefore, in the worst case scenario, *three* counters are required per partition. The importance of the extra counter is explained with the following example.

Example 6.2 To illustrate the importance of considering the case when the number of outputs are greater than the number of inputs, in this example, a single scan chain core is considered. Figure 6.6 illustrates a single scan chain core with 5 flip flops (FFs), 3 inputs and 6 outputs. The core is tested in a single scan chain mode using an on-chip decoder as the source, and a single input signature register (SISR) [36][2] as the sink. The "scan in" (s_i) and the "scan out" (s_o) are connected to the source and the sink as noted in the figure. In order to shift in the test stimuli, 8 clock cycles are required, however to shift out the test responses 11 clock cycles are required. For the difference of 3 clock cycles, the on-chip decoder has to shift in *useless* data. As shown in Chapter 5, for MSC cores, depending the width of the TAM, up to 45% of the total amount of data sent to a

[2]While various techniques can be used to compact the test responses from the core, in this chapter it is assumed that a SISR is used

core can be useless test data. This is also true for single scan chain cores. For example, for the s38584 ISCAS89 benchmark circuit with 38 inputs and 304 outputs, using the MinTest [131] dynamic compacted test set comprising 136 vectors, the amount of useless test data is 36176 bits, which will increase the size of the compressed test by 32.40%. Thus, although a compression method can be used to compress this useless test data, the obtained volume of compressed data will clearly be greater than compressing only the initial test set. Therefore, in order to refrain from compressing useless test data, a counter is used to stop the decoder for a number of clock cycles equal with the difference between the number outputs and the number inputs (see Figure 6.6).

To summarise, in order to ensure correct loading of the WSC two counters per partition are required (see Example 6.1), in addition, in the case when the number of outputs are greater than the number of inputs, one more counter is required (see Example 6.2). As noted in Chapter 5 (see Section 5.3.1 on Page 95) the **mUMA** heuristic will aim at equalising the longest input and output WSC, hence, the counter which accounts for the difference between the input and output WSC is needed only for the first (shorter) partition.

For the remainder of this chapter, $l^i_{p_k}$ and $l^o_{p_k}$ will denote the input and the output length of partition p_k respectively. It should be noted that, while the **mUMA** algorithm can be used to create more than two partitions, for the remainder of the chapter it will be considered that two partitions with WSCs of equal length, and thus two test sets, are created. It is also important to note that while changing the cardinality of the partitions affects the amount of useless test data, the maximum WSC length, which determines the TAT, is not changed. Hence, with no TAT penalties, the width of the partitions can be changed even after the TAM architecture is designed using a mixed wrapper/TAM optimisation solution [57].

Based on the above observations the following section introduces the new test data decompression architecture for low-cost SOC test.

6.2 Extended distribution architecture

Let's consider a core c, and that the **mUMA** algorithm was used to construct two partitions of cardinality p_1 and p_2. Because of the properties of the **mUMA** algorithm, the WSCs

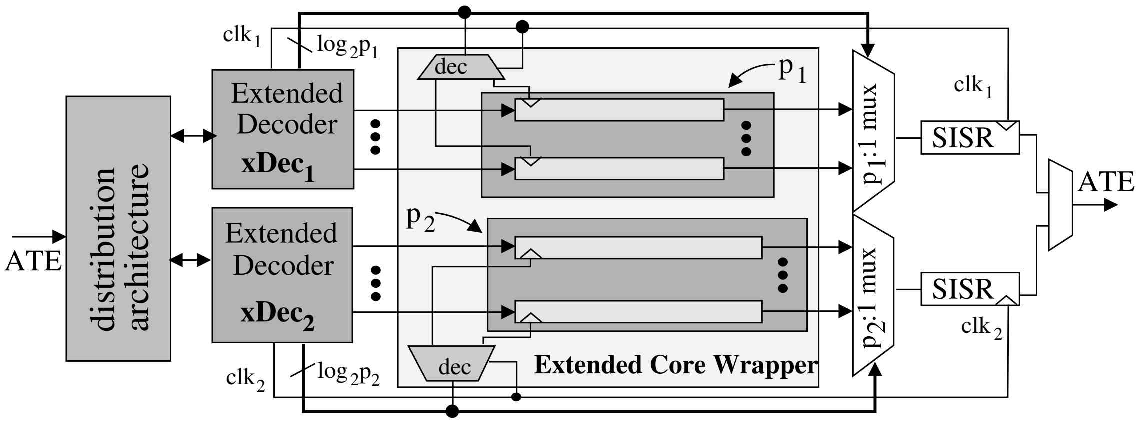


Figure 6.7. Extended distribution architecture (xDistr)

on one partition can be considered of equal length. The extended distribution architecture is illustrated in Figure 6.7. For two partitions (p_1 and p_2 in the figure), the architecture comprises two extended decoders ($xDec_1$ and $xDec_2$), two multiplexers and two SISR. The extended decoder (*xDec*) is illustrated in detail in Figure 6.8. It should noted that while for the remainder of this chapter the *xDistr* is illustrated for the variable length input Huffman coding (VIHC) scheme (see Chapter 3), it is applicable to any parallel on-chip decoder. Also note that the notations introduced in Chapters 3 and 4 are reused. For example, *dec clk* denotes the on-chip test clock used to drive the control and generator unit (CGU) part of the VIHC decoder[3]; *FSM clk* denotes the clock used to drive the Huff-decoder part of the VIHC decoder; *data in* denotes the data input from the ATE; *scan clk* is the clock generated by the CGU to drive the scan chains, and *data out* denotes the CGU data output. Also, α denotes the frequency ratio between the on-chip test frequency (f_{chip}) and the ATE operating frequency (f_{ate}).

xDec is a VIHC decoder augmented with counters, which accounts for the corresponding partition length and cardinality ($l^i_{p_k}$ and p_k) and the difference between the output and input WSC ($l^o_{p_k} - l^i_{p_k}$) if needed, as noted in the previous section. Note that this is *only* for the two partitions (p_1 and p_2) and *not* for every WSC. The main idea behind *xDec* is to decompress test data for one WSC at a time. Because the distribution architecture (see Section 4.2.2 on Page 77) allows multiple core test, the extended distribution architecture (*xDistr*) will facilitate decompression of test data for the two partitions. Hence, there are maximum two WSCs (one WSC in each partition) receiving data at any time. This leads to reducing TAT when compared to single scan chain test (see Section 6.3), and at the

[3]For more details regarding the VIHC decoder the reader is referred to Section 3.3.1 (see Page 49).

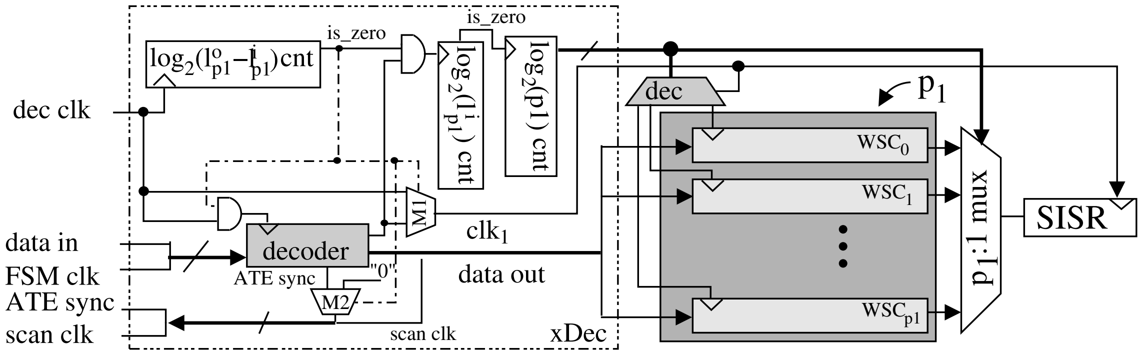


Figure 6.8. Extended decoder unit (xDec)

same time, to reducing power dissipation as detailed later in this section. To ensure WSC selection the core wrapper is augmented with one $log_2(p_i) : p_i$ *decoder* (*dec* in the figure) for each partition. The functionality of the *xDec* and the *xDistr* are illustrated next.

Let's consider that an *xDec* is attached to partition p_1 as shown in Figure 6.8. Upon reset, the counter $log_2(p_1)$ *cnt* is set to 0, the counter $log_2(l^o_{p_1} - l^i_{p_1})$ *cnt* is set to the difference $l^o_{p_1} - l^i_{p_1}$, and the counter $log_2(l^i_{p_1})$ *cnt* is set to $l^i_{p_1}$. The $log_2(p_1) : p_1$ *decoder*, part of the extended core wrapper (xWrapper), will select WSC_0. Let's assume that $l^i_{p_1} = l^o_{p_1}$, hence the *is_zero* line from the $log_2(l^o_{p_1} - l^i_{p_1})$ *cnt* will be high allowing the scan clock from the decoder to decrement the $log_2(l^i_{p_1})$ *cnt*. When this counter reaches the value 0, the $log_2(p_1)$ *cnt* is incremented and the next WSC is processed. At this point, the $log_2(l^o_{p_1} - l^i_{p_1})$ *cnt* and $log_2(l^i_{p_1})$ *cnt* will be reloaded with the initial values $(l^o_{p_1} - l^i_{p_1})$ and $l^i_{p_1}$ respectively, and the process is repeated until $log_2(p_1)$ *cnt* will reach the value p_1. When all the WSC lines have been processed, the decoder is disabled for the first partition until *xDec*$_2$, for the second partition has filled its WSCs. When both partitions have their WSCs filled, the test vector is applied and the process is repeated until the test set is completed. Similar to the distribution architecture, when one *xDec* raises the *ATE sync* line notifying that it cannot process data, the other *xDec* will be activated. This process and the construction of the test set which accompanies the *xDistr* will be detailed with Example 6.3. As noted with Examples 6.1 and 6.2, the data has to be unloaded from the WSCs into the SISR. This is done simultaneously with the loading of data into the WSCs. If $l^o_{p_1} > l^i_{p_1}$, then the counter $log_2(l^o_{p_1} - l^i_{p_1})$ *cnt* will ensure that all the data is unloaded into the SISR before any new data is loaded. To account for this case two multiplexers, *M1* and *M2*, in the figure, are required. *M1* will ensure that the SISR will receive a clock even when the decoder is disabled, and *M2* will force the *ATE sync* line low, such that no data is sent by the distribution unit during this period. It should be noted that the counter $log_2(l^o_{p_1} - l^i_{p_1})$ *cnt* is driven with the on-chip test clock, hence the number of external clock cycles required to account for the difference is given by $\lceil \frac{l^o_{p_1} - l^i_{p_1}}{\alpha} \rceil$. Since the counters which control the data flow in *xDec* can be programmed, the *xDec* is programmable, and therefore, the same *xDistr* can be reused for testing any core connected to a TAM where the tuple (p_1,p_2) is identical. The extension of *xDistr* to the case (p_1,p_2) varies among different cores is given in Section 6.2.1.

It is important to note that when compared to the number of counters and the control overhead required to test MSC cores using one decoder per scan chain, the proposed approach provides a considerable reduction in control overhead and area overhead. For example, consider a core with 16 scan chains, each of them is fed by a different decoder. Because the rate at which the data is fed into the scan chains differs, each scan chain requires its own counter. Hence, 16 on-chip decoders and 16 counters are required to feed data into the MSC core. In contrast, assuming two partitions of 8 scan chains each, the proposed approach requires *only* 2 decoders and 6 counters. Hence, the reduction in area and control overhead is obvious.

Employing the above architecture requires several steps. Firstly, the core wrapper design algorithm is performed, and the WSC configuration is obtained. From the WSCs the initial test set is split into two test sets, and each test set compressed. The compressed test sets are then tailored for the frequency ratio. Finally, the composite test set is created. The creation of the composite test set is similar to the one for the distribution architecture (see Section 4.2.2 on Page 77). However, accounting for the case when the number of outputs are greater than the number of inputs and imposing the constraint that after one partition is filled no more data is loaded until the test vector is applied, leads to some differences as exemplified next.

Example 6.3 Consider the core wrapper given in Figure 6.5 (see Example 6.1 Page 121). For the three WSCs, in Figure 6.9(a) the test vector (T_D) and the association between the WSCs and the corresponding test vector bits is given. Having the WSCs the following step is test set mapping. This is illustrated in Figure 6.9(b) where TS_1 and TS_2 are constructed from T_D. Since partition p_1 is composed out of WSC_1, the corresponding bits (*01 0000*) are copied into TS_1. Similarly for TS_2 corresponding to partition p_2, which is composed out of WSC_2 and WSC_3. For the compression of these two test sets, the VIHC dictionary given in Figure 6.9(c) is employed[4]. Note that $m_h = 4$, and that the spaces within the test sets are to illustrate the division in runs of '0's of length smaller or equal to 4. TS_1 and TS_2 are compressed and tailored[5] for a frequency ratio of $\alpha = 2$. The compressed and tailored test sets are noted with TS_1^t and TS_2^t in Figure 6.9(b). The codewords in the two test sets are delimited through spaces. When, generating the composite test set (T_C) for the distribution architecture (see Section 4.2.2 Page 77) the tailored compressed test set TS_1^t is chosen first. In this case however, this is not possible due to the fact that the $xDec_1$ corresponding to TS_1^t has firstly to ensure that the WSCs are correctly unloaded. Since the difference between the WSC_1^o and WSC_1^i (see Figure 6.5(b)) is 3, and the frequency ratio is $\alpha = 2$, $xDec_1$ will be disabled for two ATE clock cycles. Hence, when the composite test set for *xDistr* is constructed TS_2^t has to be chosen. The bits from TS_2^t are copied into T_C up to the first dummy bit (D) or until the partition is full. In the considered scenario both conditions are fulfilled for TS_2^t. This is due to the fact that TS_2 has 18 bits – which is the total amount of data to be loaded into p_2 for one test vector. With the second partition full, *xDistr* will switch to the first partition, and hence the composite test set must include bits from the first tailored test set. Having switched to TS_1^t the bits are copied into T_C until the partition is full or a dummy bit is encountered. In this case, the partition is full after the second codeword. With both partitions full, the test vector can be applied and the process repeated. It should be noted that the data is fed into *xDistr* and into the extended decoders at f_{ate} and the scan data is generated at f_{chip}.

It has been shown in [143] that the power dissipated during scan can be reduced if the number of active scan cells are reduced. For this purpose, in [143], the scan chain has been partitioned into a number of equal length scan chains and the time periods when these scan chain partitions are active are non-overlapping. The approach proposed in this section has a similar behaviour. Due to WSC selection, only one WSC in a partition can be active at any time. Having two partitions, in the worst case scenario, from the power dissipation perspective, two WSCs are active at any time. Hence, when compared to single scan chain scenarios, instead of having the entire scan chain active only two WSCs are active. Therefore, the proposed architecture achieves power reduction when compared to single scan chain designs. When compared to testing embedded cores with a single scan chain, the proposed architecture imposes some extra area overhead, however,

[4]For further details on the VIHC scheme, the reader is referred to Section 3.2 on Page 41

[5]The reader is referred to Section 4.2.1 (see Page 74) for details regarding tailoring the compressed test set for a given frequency ratio.

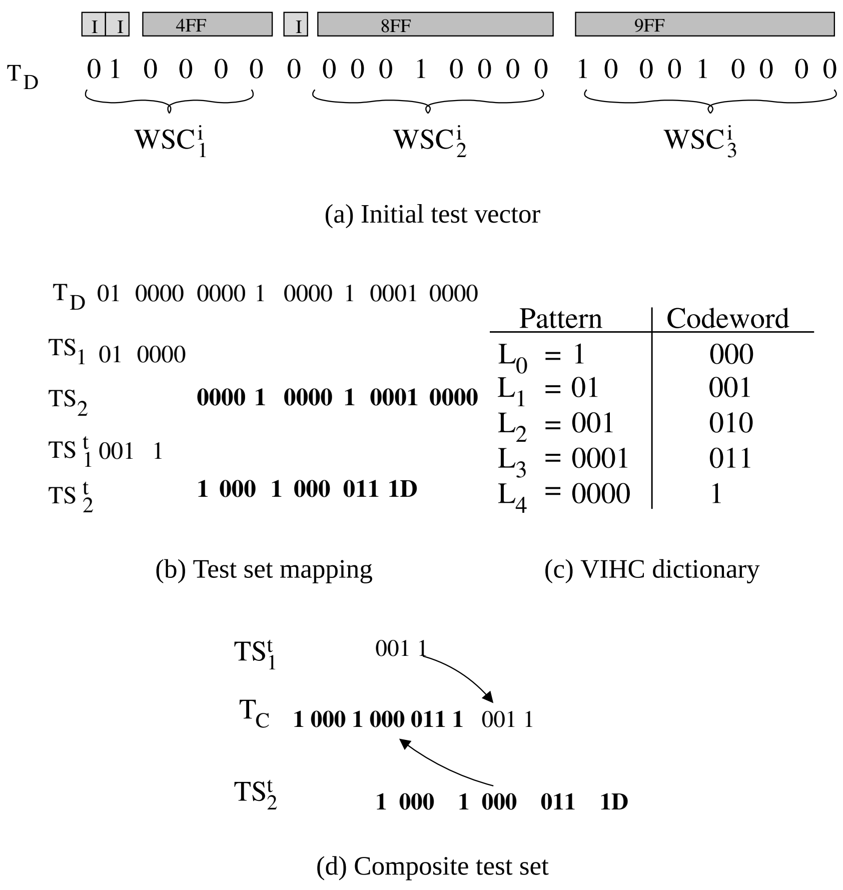


Figure 6.9. Composite test construction for *xDistr*

it inherently reduces the synchronisation overhead with the ATE due to the employment of the distribution architecture and it reduces power dissipation during scan[6].

When compared to the architecture proposed in [151], which requires test set rotation and hence it destroys the runs '0's in the test sets (see Section 6.1.2), due to WSC selection the proposed approach is capable of maintaining the test set properties hence leading to volumes of test data which are smaller than the once for single scan chain designs. In addition, due to shifting data only into maximum two WSCs at any time, the proposed approach is also capable of reducing peak scan power unlike [151].

6.2.1 TAM add-on decompression architecture

While the *xDistr* does provide flexibility with respect to testing MSC embedded cores, the approach requires that all the cores connected to the *xDistr* to have the same number

[6]An analytical analysis of the power reduction is given in Appendix E

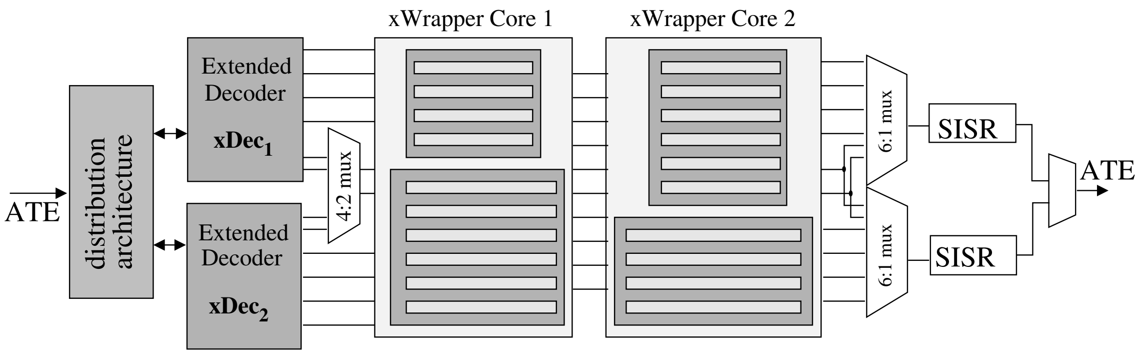

xWrapper Core 1
xWrapper Core 2
ATE
distribution architecture
Extended Decoder xDec$_1$
Extended Decoder xDec$_2$
4:2 mux
6:1 mux
6:1 mux
SISR
SISR
ATE

Figure 6.10. TAM add-on decompression architecture (add-on-xDistr)

of scan chains in a partition. This restriction is eliminated by introducing a multiplexer as illustrated in Figure 6.10.

Let's consider a TAM of width 10 and two cores connected to this TAM. The first core has two partitions of 4 and 6 WSCs respectively, and the second core has two partitions of 6 and 4 WSCs respectively. Extending the *xDistr* in this case requires the usage of *xDec*s for partitions of cardinality 6 and the addition of one multiplexer as illustrated in Figure 6.10. Depending on the tested core the multiplexer will select the last 2 lines from the first *xDec* or the first two lines from the second *xDec*. This extension will enable the *xDistr* unit to work as an *add-on* to a TAM of a given width. Hence, the proposed architecture can be inserted into the design **after** the core wrapper and the TAM have been designed. The *xDistr* architecture with the above mentioned extension will be further on referred to as *add-on-xDistr*.

As it is common to have SOCs with more than one TAM, it is shown next how an entire system can be tested using *one* test pin by employing the proposed *add-on-xDistr* architecture. It was stipulated in [121], that if the ATE operating frequency equals the SOC test frequency, *k* cores can be tested by having each core sample data once in *k* clock cycles. This approach is extended in the following, providing a conceptual architecture for SOC test using the *add-on-xDistr* architecture.

The proposed conceptual architecture is given in Figure 6.11 considering a SOC with 4 TAMs, each TAM connected to an *add-on-xDistr*. The selection logic has been omitted for clarity. Note that in the envisioned conceptual architecture, the general case of *n* partitions is considered. Therefore, each TAM will be connected at the inputs with an *add-on-xDistr* capable of handing *n* partitions, and at the outputs with *n* SISRs and the corresponding multiplexers. Each *add-on-xDistr* samples data once in 4 clock cycles, and

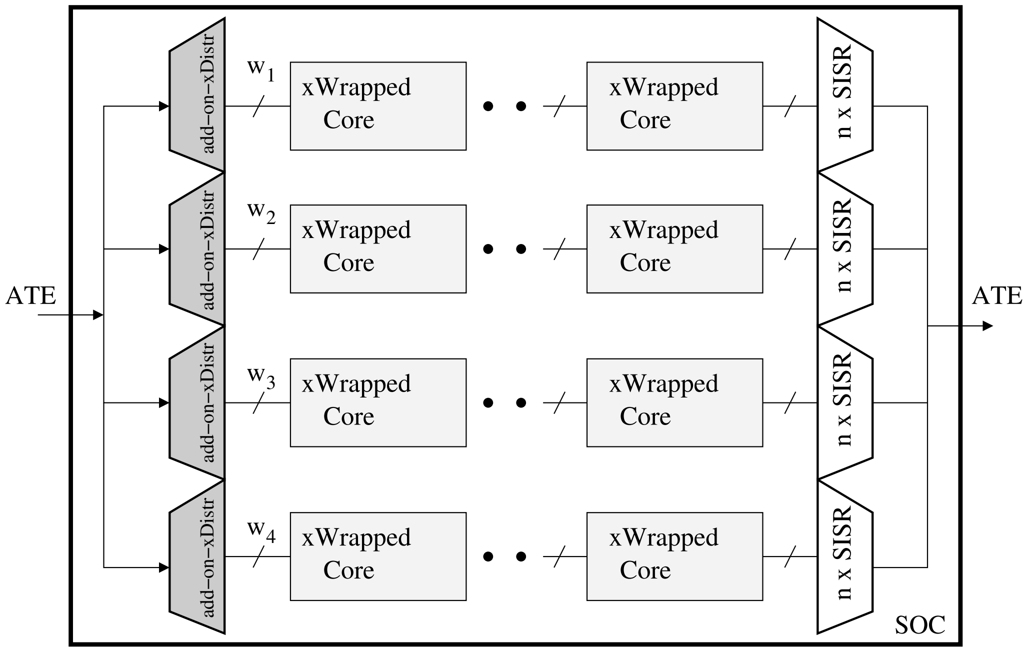


Figure 6.11. Conceptual architecture for SOC test

the *xDec*s part of the *add-on-xDistr* generate the data to scan channels at the SOC test frequency. Hence, the effective frequency ratio, between the frequency at which the data is sampled by the *xDec* and the frequency at which the data is generated by the *xDec,* is 4. Thus, the parallel on-chip decoder part of *xDec* will still be able to exploit the frequency ratio. From the design flow integration perspective, this step can be performed last, after the *add-on-xDistr* units have been designed for each of the SOC TAM's, as shown in the following section.

6.2.2 Design flow integration

In order for TDC methods to be fully exploited, they should be seamlessly integrable into any embedded core-based SOC design flow. The design flow integration issues are discussed in this section.

As illustrated in the previous section, the proposed *xDistr* architecture can be extended into an *add-on* like structure for any TAM width. This implies that the architecture can be easily introduced after the core wrapper and TAM have been designed. To illustrate the above, Figure 6.12 shows the modified design flow. As it can be seen in the figure, the TDC extension to the design flow is entered **after** the core wrapper design (CWD in the figure) and TAM design have been performed. At this point, the TAMs and the

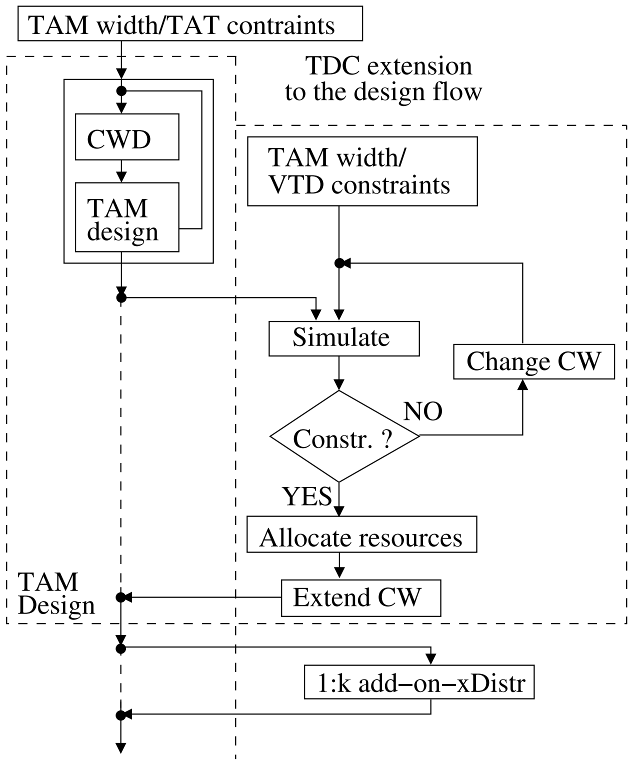


Figure 6.12. Design flow integration

core wrappers (CW in the figure) have been defined. Considering two partitions per core, the *xDistr* is simulated with each core taken separately. In order to simulate the current configuration, the steps described in the previous section are followed (i.e., based on the partitions the test sets are determined; the test sets are compressed and tailored for the frequency ratio; and the composite test set is created). If the volume of test data (VTD) constraints are not met, then the cardinality of the partitions is changed and the simulation is performed again. It should be noted that changing the cardinality of the partitions does not influence the TAT of the core but only the amount of useless test data. Hence, the TAT of the TAM design will not be influenced. When the VTD constraints are met, the required resources are allocated and the core wrappers are extended with the required decoders as illustrated in the previous section. At this point the initial design flow is reentered.

It has been experimentally observed that usually the best solution is found if the two partitions' cardinality is very close to the half of the TAM width. This is because if the test sets' lengths, obtained after the partitions have been created, do not have similar sizes then the composite test set can increase due to a large number of *dummy* bits. Using test data compression, the proposed solution reduces the ATE bandwidth requirements and the volume of test data. Furthermore, exploiting core wrapper design features, it reduces control and area overhead and provides a seamless integration with the design flow. In addition, it reduces the power dissipation during scan and it eliminates the synchronisation overhead. Hence, the proposed solution provides a low-cost test methodology for core based SOC test. Experimental comparisons with single scan chain and multiple scan chains TDC methods are given in the following section.

6.3 Experimental results

To validate the efficiency of the proposed test methodology two types of comparisons have been performed: (i) a comparison with the VIHC based single scan chain test data compression (TDC) method, using the MinTest [131] dynamically compacted test set; (ii) a comparison with a variant of the *XOR-network* [114] based architecture. In order to provide the flexibility with respect to the care bit density required by the *XOR-network* based approach, for the second experiment the Atalanta [152] ATPG tool has been used. In both cases the experiments have been performed on the largest ISCAS89 circuits [132] using a Pentium III 500 MHz Linux workstation with 256 Mb RAM. Details about the tools and the benchmark circuits used in the experiments are given in Appendix F.

6.3.1 Comparison with single scan chain TDC

Two experiments using ISCAS89 benchmark circuits are detailed in this section. Firstly, for a given TAM width and a given core, the optimum partitions' cardinality for which minimum composite test set size is obtained has been determined. And, secondly, a comparison between VIHC applied to single scan chain cores and the proposed xDistr, using equal cardinality partitions, for two fictive systems has been performed.

The specification of the ISCAS89 circuits are listed in Table 6.1. For each circuit, the number of inputs/outputs (n/m), the number of scan chains (s), the total number of FFs (FFs) and the minimum and maximum scan chain lengths (after the assignment of FFs to scan chains) are listed. It should be noted that the scan chains have been chosen to be as equal as possible. For example, for core s5378, three scan chains have the length equal to 45 and one scan chain length equals 44. For the experiments two frequency ratios have been considered: $\alpha = 2$ and $\alpha = 4$.

Core	n/m	s	FFs	min_{sc}/max_{sc}
s5378	35/49	4	179	44/45
s9234	36/39	4	211	52/53
s13207	62/152	16	638	39/40
s15850	77/150	16	534	33/34
s35932	35/320	32	1728	54/54
s38417	28/106	32	1636	32/33
s38584	38/304	32	1426	44/45

Table 6.1. Core specification

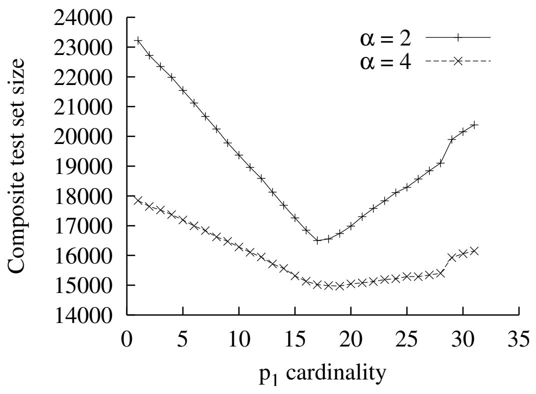


Figure 6.13. Composite test set size vs. p_1 cardinality

It has been shown in Section 6.2 that the proposed *xDistr* reduces both the area and the control overhead by efficiently exploiting the core wrapper design features of the **mUMA** heuristic proposed in Chapter 5. Furthermore, since no more than two WSCs can be active in any scan cycle, the power dissipation during scan is also reduced. Moreover, since the proposed architecture also reduces the synchronisation overhead, the volume of test data (VTD) in bits is equivalent to the TAT in ATE clock cycles.

Optimum partition's cardinality As noted in Section 6.2, the volume of test data depends on the partitions' cardinality. This is best illustrated in Figure 6.13, where the variation in composite test set size with the cardinality of the first partition is shown for core s35932 for a test bus of 32 when $\alpha = 2$ and $\alpha = 4$. It becomes clear from the figure that the minimum value is in the neighbourhood of 16, i.e., when the two partitions' cardinality are equal. The same behaviour can be observed for the other ISCAS89 benchmark circuits. These results are given in Table 6.2. The maximum TAM width for cores s5378 and s9234 has been considered 8, while for s13207 and s15850 it has been considered 24. This is because further increasing the TAM width will not lead to any TAT reduction and hence the cores will not be attached to a wider TAM. For the other cores, the parti-

Circuit	w	*part*	$l^i_{p_1} - l^i_{p_2}$	$l^o_{p_1} - l^o_{p_2}$	T_C	
					$\alpha = 2$	$\alpha = 4$
s5378	4	2-2	50-57	57-57	**13898**	**12941**
	8	4-4	9-45	12-45	**16405**	**13614**
s9234	4	2-2	61-63	62-63	**21664**	**21081**
	8	4-4	9-53	10-53	**25881**	**21578**
s13207	4	2-2	159-198	197-198	**59273**	**41932**
	8	4-4	80-99	99-99	**59137**	41585
		5-3	81-99	99-99	59675	**41561**
	16	9-7	40-50	49-50	**54795**	**39250**
		8-8	40-50	49-50	59510	41881
	24	12-12	40-40	40-40	**70430**	48542
		11-13	40-40	40-40	71350	**46826**
s15850	4	2-2	135-171	171-171	**34855**	**28454**
	8	4-4	67-86	85-86	**34933**	**28463**
	16	9-7	35-43	43-43	**34332**	29830
		8-8	33-43	43-43	34980	**28488**
	24	12-12	33-34	33-34	**38586**	32696
		10-14	33-34	33-34	41033	**31124**
s35932	4	2-2	432-512	512-512	**16526**	**14336**
	8	5-3	216-256	256-256	**16555**	14480
		4-4	216-256	256-256	16609	**14479**
	16	9-7	108-128	128-128	**16082**	**14348**
		8-8	108-128	128-128	16679	14577
	24	15-9	54-108	72-108	**15957**	**14072**
		12-12	54-108	63-108	18134	15174
	32	18-14	54-64	64-64	**16404**	**14687**
		16-16	54-64	64-64	17105	15048
s38417	4	2-2	409-436	435-436	**92167**	**87110**
	8	4-4	204-218	218-218	**92522**	**87321**
	16	8-8	102-109	109-109	**93102**	87781
		9-7	102-109	109-109	93469	**86870**
	24	16-8	53-102	58-102	**90965**	**86813**
		12-12	51-102	51-102	107866	93511
	32	17-15	51-55	54-55	**91849**	**87225**
		16-16	51-55	54-55	93233	88354
s38584	4	2-2	356-433	432-433	**98133**	**83761**
	8	4-4	178-217	216-217	**98164**	83799
		5-3	178-217	216-217	98273	**83572**
	16	9-7	89-109	108-109	**93302**	**82487**
		8-8	89-109	108-109	98351	84088
	24	16-8	47-89	64-89	**91239**	81546
		15-9	45-89	62-89	93624	**81427**
		12-12	45-89	56-89	109458	88123
	32	18-14	45-55	54-55	**93695**	**82869**
		16-16	44-55	54-55	101337	87442

Table 6.2. xDistr composite test set

tions and the composite test sets have been determined for TAM widths between 4 and 32. For each core and for every TAM width (w), the partitions which yield the minimum composite test set for a $\alpha = 2$ and $\alpha = 4$, and the case when both partitions are equal, are reported. The table lists the core, the test bus width (w), the partitions cardinality (*part*) and length of the longest input and output wrapper scan chain (WSC) for the two partitions ($l^i_{p_1} - l^i_{p_2}$ and $l^o_{p_1} - l^o_{p_2}$). The VTD obtained for each of the cases is given in columns 6 and 7. The values marked with bold in these columns represent the minimum VTD for the corresponding partition and frequency ratio.

As it can be observed in the table, for most cases, choosing two partitions of equal cardinality yields a good result. This leads to the conclusion, that while the *xDistr* can be extended to account for different partition cardinality for different cores, this may not always be required. Hence, to further shorten the duration of the design flow, with small, or no, penalties in VTD, the solution with two partitions of equal cardinality can be chosen as a default solution, or as a good starting point if further VTD reduction is required. To exemplify the above consider the case of circuit s38417 with a TAM width equal to 32, where the partitions' cardinality for minimum composite test set size when $\alpha = 2$ and $\alpha = 4$ is *17-15*. The equal cardinality partition, *16-16* has an increase in volume of test data of 1.48% in the case when $\alpha = 2$, and 1.27% in the case when $\alpha = 4$. Thus, choosing the equal cardinality partition of *16-16* provides a viable solution for this circuit. The simulation times, comprising compression of the test sets corresponding to each partition; tailoring the two test sets for the given frequency; and the generation of the composite test set, ranged from 1 second for the circuits with small test sets to 9 seconds for the circuits with large test sets, for each core for one configuration. Thus, the extra time needed to perform the simulation step will have a small influence on the overall duration of the design flow.

VIHC vs. xDistr To illustrate the amount of test data reduction when compared to single scan chain cores two systems have been considered: the first one comprising s13207, s15850, s35932, s38417 and s38584 attached to a 24 bit wide TAM and the second one comprising s35932, s38417 and s38584 connected to a 32 bit wide TAM. The VIHC method has been used to compress the initial test sets, and then the compressed test sets have been tailored for a given frequency ratio. In addition, the difference between the number of outputs and the number of inputs has been also taken into account as detailed in Section 6.1.3. The results are reported in Table 6.3. It is important to note that, since no more synchronisation overhead is imposed, the VTD given in this table is equivalent to the TAT in ATE clock cycles. For the frequency ratios $\alpha = 2$ and $\alpha = 4$, the table lists the

		T_C					
		$\alpha = 2$			$\alpha = 4$		
Circuit	*w*	*init*	*eq-parts*	*red %*	*init*	*eq-parts*	*red %*
s5378	4	18445	**13898**	24.65	13784	**12941**	6.51
	8		**16405**	12.43		**13614**	1.23
s9234	4	27474	**21664**	21.14	24019	**21081**	12.23
	8		**25881**	5.79		**21578**	10.16
s13207	4	101540	**59273**	41.62	60657	**41932**	30.87
	8		**59137**	41.75		**41585**	31.44
	16		**59510**	41.39		**41881**	30.95
	24		**70430**	30.63		**48542**	19.97
s15850	4	54091	**34855**	35.56	37759	**28454**	24.64
	8		**34933**	35.41		**28463**	24.61
	16		**34980**	35.33		**28488**	24.55
	24		**38586**	28.66		**32696**	13.40
s35932	4	22196	**16526**	25.54	16793	**14336**	14.63
	8		**16609**	25.17		**14479**	13.77
	16		**16679**	24.85		**14577**	13.19
	24		**18134**	18.30		**15174**	9.64
	32		**17105**	22.93		**15048**	10.39
s38417	4	122850	**92167**	24.97	94757	**87110**	8.07
	8		**92522**	24.68		**87321**	7.84
	16		**93102**	24.21		**87781**	7.36
	24		**107866**	12.19		**93511**	1.31
	32		**93233**	24.10		**88354**	6.75
s38584	4	145937	**98133**	32.75	101489	**83761**	17.46
	8		**98164**	32.73		**83799**	17.43
	16		**98351**	31.80		**84088**	17.14
	24		**109458**	24.99		**88123**	13.16
	32		**99166**	32.04		**84074**	17.15

Table 6.3. Volume of test data comparison

volume of test data using a single scan chain core (*init*) and using the *xDistr* architecture with two partitions of equal cardinality ($eq-parts$). The reduction in VTD is on average over 20% for $\alpha = 2$. For example, it is 41.62% in the case of s13207 for a TAM width equal to 4. For $\alpha = 4$, the reduction in VTD ranges from 1.23% in the case of circuit s5378 for a TAM width equal to 8, to 31.44% in the case of circuit s13207 for a TAM width equal to 8. Hence, considerable savings in VTD are achieved when compared to single scan chain designs. It is important to note that for one core, over all TAM widths, the VTDs are similar. For example, in the case of circuit s15850, the volume of test data is of 34855 bits for TAM width 4, and of 38586 bits for TAM width 32. This implies that the *xDistr* architecture can be attached to any TAM of any width without imposing any restrictions on the existing TAM and still obtain considerable savings in the VTD. Hence, when routing congestions do not allow a wide TAM the proposed architecture will be able to ensure considerable reduction in TAT.

		T_C					
		$\alpha = 2$			$\alpha = 4$		
w	Circuit	*init*	*eq-parts*	*red %*	*init*	*eq-parts*	*red %*
24	s13207	101815	71878	29.40	62492	51235	18.01
	s15850	55235	41143	25.51	39171	35279	9.93
	s35932	22512	21889	2.76	17215	18947	-10.06
	s38417	125781	116614	7.28	98942	100950	-2.02
	s38584	151357	117058	22.66	109940	96463	12.25
Total		456700	**368582**	**19.29**	327760	**302874**	**7.59**
32	s35932	22512	17499	22.26	17215	15477	10.09
	s38417	125781	95656	23.95	98942	89496	9.54
	s38584	151357	105938	30.00	109940	92297	16.04
Total		299650	**219093**	**26.88**	226097	**197270**	**12.74**

Table 6.4. add-on-xDistr for TAM of widths 24 and 32

Finally, the case when the cores from a system are connected to a TAM of a given width is analysed. The two systems described at the beginning of this section have been considered. The experimental setup for both systems is as follows: firstly, for all the cores test set partitioning has been performed assuming equal cardinality partitions; secondly, the test sets corresponding to each partition have been concatenated into one test set. The resulting test set for each partition has been compressed using the VIHC method and the resulting dictionary has been stored. Using this dictionary the composite test set has been constructed for each core taken separately. Using one dictionary over all the test sets will ensure the usage of one VIHC decoder to decompress all the test sets for one partition. To provide an accurate comparison, the same experimental setup has been used to compress the systems assuming single scan chain cores. The results are reported in Table 6.4. It should be noted that in some cases the test set obtained assuming one scan chain per core is smaller than in the case of the proposed solution. For example, for the s38417 circuit a 2.02% increase in the volume of test data can be observed. This is because concatenating the different test sets can affect the overall pattern distribution and hence the VIHC compression. However, the total amount of test data obtained using the proposed *xDistr*, is still considerably smaller than using a single scan chain core test approach. For example, in the case of the second system 26.88% reduction in the total volume of test data is obtained for $\alpha = 2$.

To give an estimation of the area overhead introduced by the proposed *xDistr* architecture, the systems considered in the previous comparison have been synthesised using the Synopsys Design Compiler [133]. The experimental setup is as follows: firstly, full scan design for test has been performed on the cores; secondly, each core has been wrapped for

Core	n/m	s	FFs	min_{sc}/max_{sc}	$\%cb$
s5378	35/49	4	179	44/45	20.58 – 24.64
s9234	36/39	4	211	52/53	16.28 – 28.72
s13207	62/152	16	638	39/40	6.01 – 8.88
s15850	77/150	16	534	33/34	6.77 – 15.11
s35932	35/320	32	1728	54/54	2.80 – 59.36
s38417	28/106	32	1636	51/52	2.63 – 21.73
s38584	38/304	32	1426	44/45	3.23 – 17.19

Table 6.5. Core specification

a test bus width of 1[7]. The proposed *xDistr* architecture has been synthesised for the two TAMs. The test area overhead is of 2.8% in the case of the first system and 3.5% in the case of the second one. It is important to note that the *xDistr* architecture has been implemented entirely, including the programmability features and the SISRs, and the extensions to the core wrappers have been also considered. Since the *xDistr* architecture (comprising the *xDecs* and the SISRs) has fixed area overhead for a given test bus width, increasing the number of cores connected to the test bus will have small impact on the overall test area overhead. This is because adding a core to the proposed *xDistr* architecture basically implies extending the core wrapper as detailed in Section 6.2. Since the overhead of the *xWrapper* when compared to the initial core wrapper is less than 2%, the greater the number of cores connected to a *xDistr*, the smaller the chip area overhead penalty. It should be noted that more than half of the area overhead is due to programmability, i.e., shadow registers for the counters in *xDec*. Also note that when compared to the single scan chain TDC method, the area overhead is approximately doubled. This, however, at the benefit of reduction in VTD and power dissipation.

6.3.2 Comparison with multiple scan chain TDC

In this section a comparison between the proposed *add-on-xDistr* and a variant of the *XOR-network* [114] based architecture is given. To assess the two compression architectures in a core based SOC environment three variables have been chosen: the care bit density of the test set (%*cb*), the test bus width (w), and the frequency ratio (α). All the TDC methods are at some extent dependent on the number of care bits in the test sets, hence, varying the care bit density will influence the amount of compression attainable by the two architectures. An important factor in reducing the TAT in core based SOCs is the

[7]Note that since no routing overhead has been considered, choosing a wider test bus will have minor influence on the area overhead

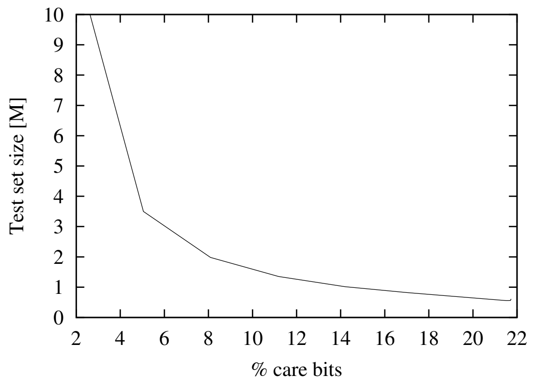


Figure 6.14. Test set size decrease with increase of $\%cb$

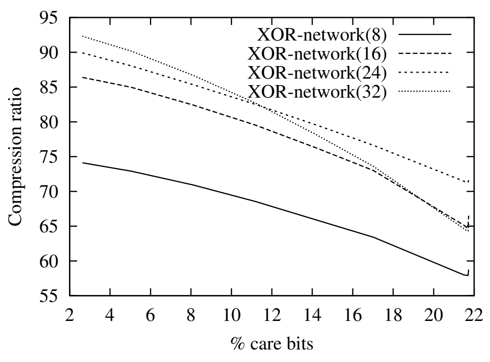
width of the test bus [55]. As it will be seen in this section this influences the TAT and the power dissipation when considered with the two compression architectures. Finally, the frequency ratio is, on the one hand, a requirement for the add-on-xDistr architecture (see Section 6.2), and on the other hand, it has also an impact on the power dissipation for this architecture. In order to provide the above experimental environment controlled static compaction has been performed, followed by core wrapper design, and finally, the two approaches have been applied and analysed.

The above steps have been performed on the largest ISCAS89 [132] benchmark circuits. The specifications of the circuits are given in Table 6.5. Similar to Section 6.3.1, for each circuit, the number of inputs/outputs (n/m), the number of scan chains (s), the total number of FFs (FFs), the minimum and maximum scan chain lengths (after the assignment of FFs to scan chains), and the range of considered care bit densities per test set ($\%cb$) are listed. To model the different care bit densities, Atalanta [152] has been used to generate test sets comprising one test vector for each fault. Subsequently, controlled static compaction has been performed using a simple greedy heuristic. The heuristic compacted the test set such that if the test vectors had a number of care bits smaller than a control value B, then they were merged with compatible test vectors if the resulting number of care bits did not exceed B. Controlled compaction has been performed for values of $B = \{50, 100, 150, 200, 250, 300, 400, 800, 1000, 2000\}$. The variation of the initial test set size with $\%cb$ is given in Figure 6.14 for core s38417. The system integrator could employ such a controlled compaction scheme, since, as it will be seen next, substantial reduction in VTD can be attained when combined with the two architectures.

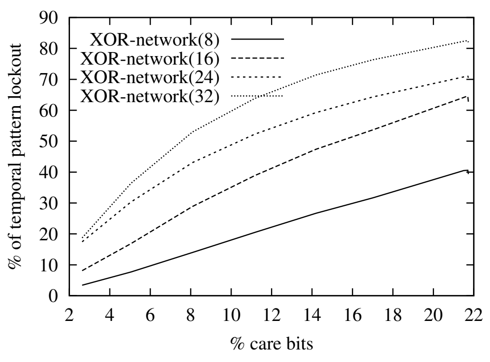
As noted in Section 6.1.2, one of the disadvantages of the *XOR-network* based approach is that it suffers from linear dependencies which can lead to temporal pattern lockout, i.e., the inability to justify the bits required at the inputs of the wrapper scan chain (WSC) at a given moment. In addition, care must be taken to avoid structural pattern lockout, i.e., when the architecture is unable to justify the required bits, regardless of how many cycles the WSC load is halted. Structural pattern lockout must be avoided as it can affect fault coverage. For details regarding the implemented *XOR-network* variant, the reader is referred to Appendix E.

Varying the care bit density (%cb), the test bus width (w) and the frequency ratio (α), in the following the influences on the compression ratio, test application time, area overhead and power dissipation are analysed. Note that only selected data is used to illustrated the various advantages and disadvantages of the two architectures. More experimental results can be found in Appendix E. As noted in Section 6.1.2, the XOR-network requires two ATE channels, hence the number of bits representing VTD will be double the number of clock cycles representing TAT. The add-on-xDistr requires only one ATE channel, and therefore, the VTD equals the TAT, which is given in ATE clock cycles. In the figures shown in this section “XOR-network(w)” denotes the XOR-network when used for a test bus width of w; and “add-on-xDistr(w,α)” denotes the add-on-xDistr when used for a test bus width of w and a frequency ratio of α.

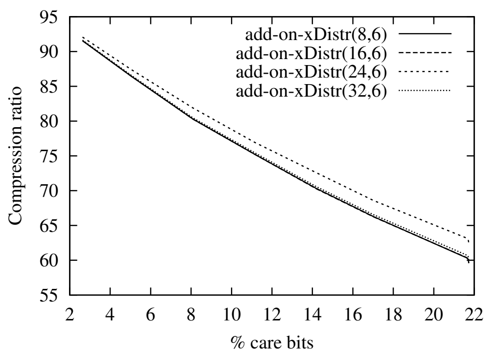
Compression ratio The variation in compression ratio with the increase in %cb is analysed in Figure 6.15. For core s38417, Figure 6.15(a) shows the influence of %cb on the compression ratio when different values of w are considered. Since the compression ratio for the XOR-network is directly related to the percentage of temporal pattern lockouts (%tpl), Figure 6.15(b) shows their variation when %cb increases. It is interesting to note that there is an improvement in compression ratio with the use of a greater w, however, the %tpl increases as well. This is due to a greater number of care bits which have to be justified through the XOR-network when w increases. At the same time, however, the number of WSC loads decreases for greater w’s, e.g., for core s38584 with %$cb = 17.19$, there are 53599, 26923, 21983, 13585 WSC loads for $w = 8$, 16, 24, and 32 respectively. Hence, as long as the decrease in the number of WSC loads is greater than the increase in %tpl, there is an improvement in compression ratio when increasing w as illustrated in Figure 6.15(a). Figure 6.15(c), shows the influence of w and %cb for $\alpha = 6$ on the compression ratio attainable with add-on-xDistr. What is interesting to note here is that, even though the compression ratio decreases when %cb increases, it is approximately constant



(a) s38417: Compression ratio



(b) s38417: Temporal lockout percentage

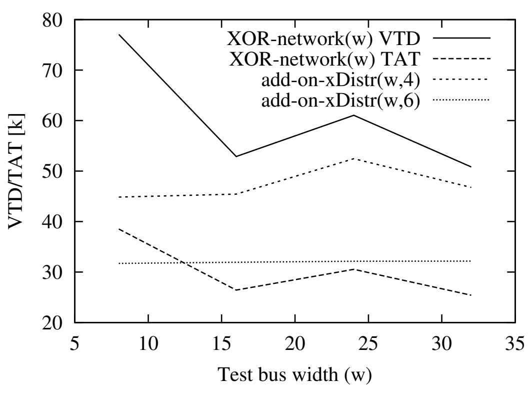


(c) s38417: Compression ratio

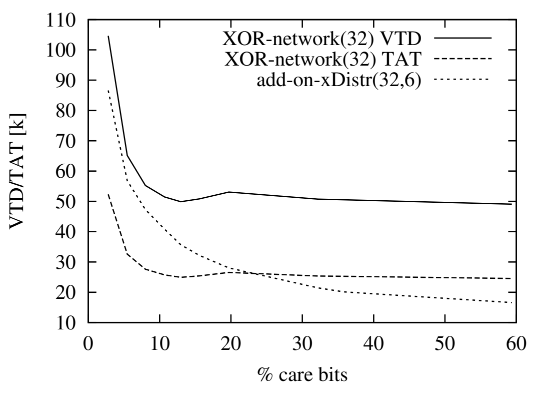
Figure 6.15. Compression ratio and temporal pattern lockout

for all the values of w for a given α. This is due to the runs of ’0’s based coding scheme used with the *add-on-xDistr* architecture and the fact that the proposed architecture does not destroy these runs of ’0’s. The influence of the three variables on the TAT is analysed next.

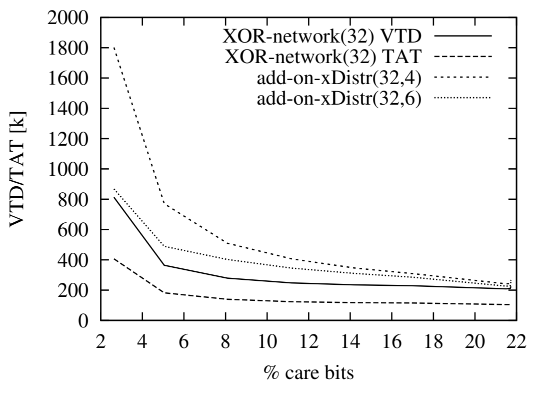
Test application time The TAT for the two architectures is analysed in Figure 6.16. Since in the case of the XOR-network, the VTD is twice the TAT, the VTD is also plotted in the figure. Figure 6.16(a) shows how VTD and TAT vary with w for core s35932. As also noted previously, the compression attainable by add-on-xDistr is almost constant for different w’s for the same α. On the other hand, the XOR-network has different VTDs and TATs with the variation of w. In this particular case the TAT obtained with XOR-network is smaller than the one for add-on-xDistr when $w \geq 16$, however, the VTD is greater. This varies depending on the circuit, α and $\%cb$. Two cases are illustrated with Figures 6.16(b) and 6.16(c). For example, for core s35932 in Figure 6.16(b), the TAT and VTD obtained



(a) s35932: VTD/TAT for $\%cb = 15.58$



(b) s35932: VTD/TAT for $w = 32$ and $\alpha = 6$



(c) s38417: VTD/TAT for $w = 32$

Figure 6.16. Volume of test data and test application time

by add-on-xDistr for $w = 32$ and $\alpha = 6$ are overall smaller than both the TAT and the VTD obtained by XOR-network for the same w. While for core s38417 in Figure 6.16(c) for $w = 32$, the VTD and TAT obtained with add-on-xDistr is greater than both the VTD and TAT obtained with XOR-network. Even though, the two architectures have different behaviours when w varies, if $\alpha = 6$ and $w = 32$ the TATs are similar. It is also interesting to note that the TAT and VTD tend to have a steep fall for $\%cb$ in the first half of the considered $\%cb$ interval. However, for the second half the curve is almost linear. This is contrary to the compression ratio behaviour, which tends to decrease in a linear manner (see Figures 6.15(a) and 6.15(c)). This is due to the steep decrease in the compacted test set size as illustrated in Figure 6.14. Therefore, the system integrator could consider using controlled compaction to reduce TAT and VTD regardless of the employed compression architecture. The analysis with respect to area overhead is given next.

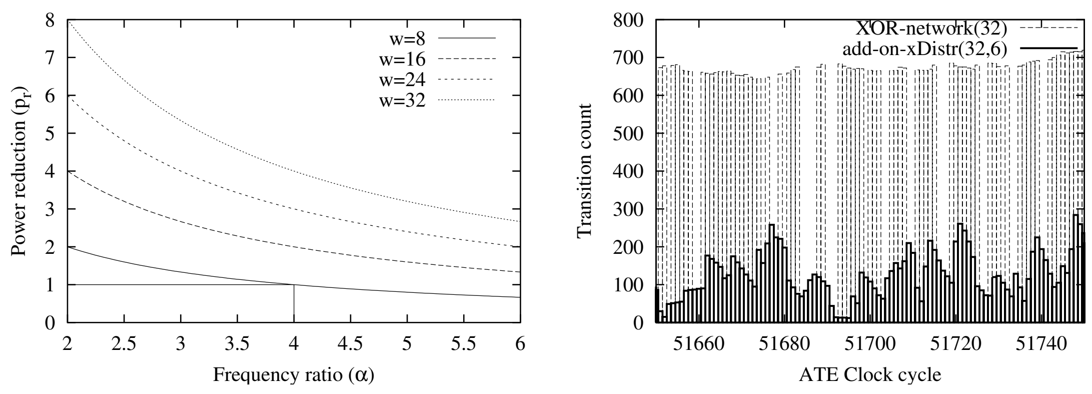
Area overhead To give an estimation of the area overhead variation with w, Synopsis Design Compiler [133] has been used. The results for core s38584 in technology units for the lsi10k library are tabulated below.

$w =$	8	16	24	32
XOR-network	370	698	1023	1570
add-on-xDistr	1886	1813	1813	1742

For the XOR-network based architectures, the area overhead is linearly dependent on the size of the test bus. This is because, in order to ensure good TAT and VTD, the XOR-network has been chosen with an average of $w/2$ inputs driving one XOR-network output. For add-on-xDistr, however, the changes are negligible. This is because by increasing w, the size of the WSCs get smaller, and hence, the size of the counter which accounts for the length of the partition decreases, while the size of the counter which accounts for the cardinality (height) of the partition increases (see Section 6.2).

Power dissipation The influences of w and α on the power dissipation are analysed next. Firstly it is interesting to note that due to the usage of maximum two WSCs at any time, the add-on-xDistr architecture can considerably reduce power dissipation, especially when the number of WSCs is large. This, despite the fact that the WSCs are driven by the on-chip test frequency. Assuming the same switching probability and all the WSCs of the same length, it can be easily derived that the power reduction factor[8] when compared to XOR-network is $p_r = w/(2 * \alpha)$. This is illustrated in Figure 6.17(a) for different w's. Hence, while the add-on-xDistr has almost constant TAT and VTD for different values of w, it can considerably reduce power dissipation when increasing w. Also, the power reduction is guaranteed for $w \geq 16$. For $w = 8$, reduction is attainable only for $\alpha \leq 4$. Another notable observation is the power profile of the two architectures. This is illustrated in Figure 6.17(b). It can be easily seen that while the XOR-network has almost constant number of transitions, the add-on-xDistr tends to have short periods of higher number of transitions. These, however, considerably smaller than the XOR-network's ones. This is due to the WSCs inputs generated by the two architectures. The XOR-network generates the WSCs inputs with almost random properties, hence the transition probability is high, while the add-on-xDistr generates runs of '0's with lower transition probability.

[8]How this formula has been derived is illustrated in Appendix E.



(a) Power reduction (b) s38584: Power profile

Figure 6.17. Power dissipation

To summarise, while the proposed architecture and the *XOR-network* architecture obtain often similar VTD, TAT, and power dissipation values, the main advantages of the proposed architecture are: (i) easy design integration, due to almost constant TAT and exploitation of the core wrapper design; (ii) reduced scan power even though the scan chains are driven with the on-chip test clock, and not with a slow clock as in the case of *XOR-network*; and (iii) the architecture is self synchronous, requiring only one test channel, unlike the *XOR-network* which requires two test channels.

6.4 Concluding remarks

This chapter has analysed the TDC approaches for multiple scan chain cores and proposed a new integrated solution which combines a new test data decompression architecture with the features of the core wrapper design algorithm proposed in Chapter 5. Using TDC, the proposed solution reduces the ATE bandwidth requirements and the volume of test data. Further, by exploiting the core wrapper design features, it decreases the control and area overhead and provides seamless integration with the design flow. In addition, it reduces power dissipation during scan and it eliminates the synchronisation overhead with the ATE. Hence, the proposed integrated solution provides an effective low-cost methodology for testing core based SOCs.

Chapter 7

Conclusions and future research directions

With the ever increasing demand for smaller devices and the requirement for shorter times to market, system-on-a-chip (SOC) becomes the driving force of the semiconductor industry. However, due to embedding cores from different intellectual property (IP) vendors, testing these SOCs can become the bottleneck of this new paradigm. In addition the high complexity of the chips requires expensive test equipment which leads to high test costs. Therefore, if the test issues raised by the SOC paradigm are not addressed, the advantages of core based SOC design are diminished.

Based on a cost of test analysis, this dissertation has identified a number of factors which need to be reduced in order to contain the cost of test. These include: the volume of test data (VTD), the number of pins for test, the bandwidth and the cost of test equipment. In addition, considering a core based SOC environment requires that the system integrator uses only basic core information and refrains from modifying the core's internal structure or test methodology. The above factors have been addressed and viable solutions which contribute toward low-cost test in core based SOCs have been provided. The main driver behind this dissertation has been test data compression, a test resource partitioning scheme, which in addition to reducing the VTD can also reduce the bandwidth requirements. The efficiency of the proposed solutions has been demonstrated using academic and industrial benchmarks. An analysis of the contributions presented in this dissertation is given next.

In Chapter 3, a compression method called Variable-length Input Huffman Coding (VIHC) has been proposed and using extensive experimental results shown that it can improve the parameters of the test data compression environment (TDCE): the compression ratio, area overhead and test application time (TAT). This is achieved by accounting for the multiple interrelated factors that influence the TDCE's parameters, such as preprocessing the test set, the size and the type of the input patterns to the coding algorithm, and the type of the decoder. With small VTD and small TAT, the proposed solution for test data compression/decompression reduces two of the three low cost test parameters: VTD and bandwidth requirements. Hence, the proposed solution contributes toward low cost SOC test. In addition to the above the VIHC decoder exploits the frequency ration between the on-chip test frequency and the ATE operating frequency. This leads to decoupling the ATE from the device under test. This is due to the fact that the entire on-chip decoder is driven only by on-chip test clocks, and the data synchronisation is performed using an on-chip generated clock. This clock, has to be active only once in the ATE clock cycle, easing the clock synchronisation between the chip and the ATE, hence decoupling the ATE from the device under test.

Due to test resource partitioning in general, and test data compression in particular, synchronisation between the ATE and the on-chip decoder is required. This is especially important when the operating conditions of the on-chip decoder cannot be fulfilled as illustrated in Chapter 4. Hence, synchronisation overhead is introduced which reduces the benefits of exploiting the frequency ratio between the on-chip test frequency and the ATE operating frequency. This issue has been addressed and a reduced pin count test method has been proposed which in addition to providing a complete scenario for SOC test when multiple single scan chain cores are targeted, is generic, scalable and programmable. Hence, it is easy integrable in the design flow, as no restrictions are imposed on the system integrator. Therefore, an efficient reduced pin count test solution has been proposed, which leads to low VTD and low TAT without any constraints or changes to the test or interface equipment when single scan chains cores are targeted.

Efficient usage of test resources becomes an important issue especially when these are critical. This is the case for the ATE memory, which directly influences the cost of the ATE. In Chapter 5 the test data stored in the ATE memory is analysed, and *useless memory* is identified as the source for the trade-off between test bus width and VTD in multiple scan chains-based cores. To eliminate this memory, in a resource efficient manner, a new test methodology has been proposed, which based on employing wrapper scan chain partitioning in core wrapper designs and exploiting ATE memory management features can obtain considerable reduction in useless memory. Hence, the proposed methodology demonstrates that with the advent of the new generation ATEs, which allow greater flexibility and provide memory management capabilities, methodologies complementary to test data compression can be used to reduce the VTD, and hence the cost of testing complex SOCs.

Finally, Chapter 6 provides an integrated test methodology for core based SOC by combining the solutions proposed in this work. The proposed methodology reduces the ATE bandwidth requirements and the VTD. Furthermore, by exploiting the core wrapper design features, it decreases the control and area overhead and provides a seamless integration with the design flow. In addition, it reduces power dissipation during scan and eliminates the synchronisation overhead with the ATE. Therefore, the proposed integrated solution provides an effective reduced pin count low-cost methodology for testing SOCs.

This research has shown that test resource partitioning facilitates a low-cost SOC test methodology, however, as also emphasised throughout the dissertation, care must be taken to ensure that all the factors which drive the cost of test are accounted for. If some of the factors are ignored, solutions which lead only to a partial low-cost test strategy are developed. Therefore, the compression method proposed in Chapter 3 aims at simultaneously improving the TDCE parameters, and the distribution architecture proposed in Chapter 4, not only reduces the synchronisation overhead between the ATE and the SOC, but also leads to TAT reduction. In addition, the novel test methodology, provided in Chapter 5, which targets useless memory reduction aims at reducing the VTD and minimising the control overhead on the ATE. Finally, the integrated core wrapper design and test data compression solution for core based SOC test, provided in Chapter 6, facilitates reduction of VTD, TAT, synchronisation overhead, and power dissipation whilst it provides seamless integration with the design flow.

Due to the reduction in feature size and the necessity to increase chip dynamics in order to identify new faults, providing data at high speeds to the chip under test becomes critical. This, however, by containing the total power dissipation of the chip in order to avoid destructive testing. The methodologies proposed in this thesis provide a possible scenario which accounts for the above requirements. However, due to the DFT dependency on scan to deliver the test vectors, the volume of test data will continue to increase.

To cope with this issue, and to maintain a high compression ratio, it is expected that area overhead requirements for the VIHC scheme will increase. Also, due to the different features of the test set, one can envision a programmable VIHC decoder which can be tuned for best performances depending on the features of the test set. Therefore, it is anticipated that the methods proposed in this dissertation, and validated on academic and industrial benchmarks, will scale to large industrial designs with small penalties in area overhead.

7.1 Future research directions

Based on work performed in this dissertation, a number of future research directions which will contribute toward low-cost test are briefly outlined next.

Improving test data compression through test set transformations As illustrated in Chapter 3, an efficient mapping and reordering algorithm can lead to significant improvement in compression ratio due to enhancing the properties exploited by the runs of ’0’s based compression schemes. In addition, an important conclusion from the comparison performed in Chapter 6 (see Section 6.3.2 on Page 139) is that even if the test sets are compacted there is still a considerable amount of compression. However, the two architectures considered in the experiment suffer from reduced compression ratios when the care bit density is considerably increased. This can be also attributed to the static compaction heuristic used in Section 6.3.2, since it does not account for any of the particularities of the two methods. Hence, the implications of test set transformations in correlation with test data compression requires further investigation.

Programmable core wrapper and test access mechanism design It has been shown in Chapter 5 that core wrapper design can be used in contexts different than the traditional “minimising TAT”. Therefore, it would be interesting to assess the potential of considering, for example, programmable core wrapper designs, which can dynamically switch between different wrapper scan chains configurations. In addition, as also illustrated in Chapter 5 tunning core wrapper design for different aims can lead to properties which can further exploited. For the UMA problem targeted in Chapter 5, the core wrapper properties have been exploited by the test vector deployment procedure. Therefore, it should be investigated whether a more efficient way of exploiting core wrapper properties is to embed them into test access mechanism design algorithms.

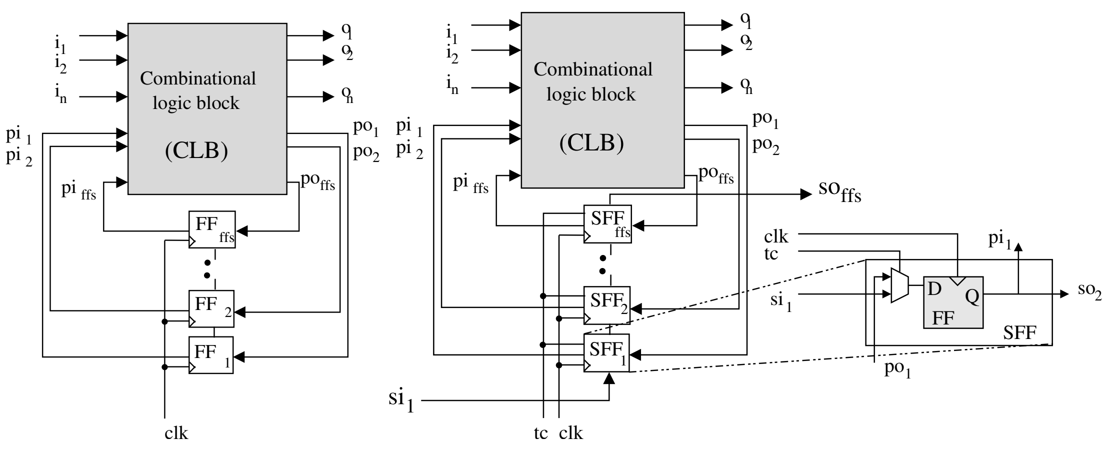
Power conscious high speed test using low-cost testers Advances in ATE technologies made available test infrastructures like low-cost DFT testers [12, 11, 153], which are not only specifically build to exploit DFT features but also support multi-site test. Multi-site test is a test technology which allows testing of multiple systems in parallel. In order for this technology to be efficient, reduce pin-count aware DFT methodologies are required. An increasingly important issue is the testing of high speed chips [4]. These require data delivery at very high frequencies while containing the total power dissipation during test. In addition, with the increased gap between ATE bandwidth availability and SOC test speed requirements [14], providing data at high speeds to the SOC using a small number of pins becomes a critical point in facilitating high speed test. Therefore there is a conflict between the limited number of ATE pins and the high frequencies required for high speed test. The test data compression methodology provided in this dissertation, by driving the cores with the on-chip test frequency, has the potential to increase chip dynamics leading to at-speed structural test without the problems imposed by the traditional slow/fast clock approach [154] and using a reduce pin-count strategy. Therefore extending test data compression to power conscious high speed test using low-cost testers, needs further investigation.

Appendix A

Core level DFT

In this appendix core level design for test (DFT) is briefly introduced. To facilitate core based SOC test, the core vendor must augment its IP with test features. The design process which embeds test features into a core is referred to as DFT. The main purpose of DFT is to increase the controllability, i.e., the ability to control the nodes within the circuit, and the observability, i.e., the ability to observe the nodes within the circuit. Two well known DFT methodologies are illustrated in this appendix: scan based DFT and built-in self-test (BIST) based DFT.

Scan based DFT A core can be generically viewed as illustrated in Figure A.1(a). The core in the figure has n inputs noted $i_1 \ldots i_n$ and m outputs noted $o_1 \ldots o_m$. The core comprises the combinational logic (shown in the figure with the combinational logic block (CLB)) and the sequential block (shown in the figure with the flip-flops (FF) – $FF_1 \ldots FF_{ffs}$). The outputs from the CLB into the FFs are also referred to as pseudo-outputs (po); the outputs from the FFs into the CLB are referred to as pseudo-inputs (pi). These are marked in the figure with $po_1 \ldots po_{ffs}$ and $pi_1 \ldots pi_{ffs}$ respectively. While there is control of the inputs and observation of the outputs of the core, there is no control or observation of the memory elements. Increasing the control and observation of the memory elements is achieved mainly using scan based DFT. Scan based DFT implies transforming the FFs in the figure into a chain of elements, in which each FF is replaced with a scan flip-flop (SFF) or scan cell. The new core with embedded scan DFT is illustrated in Figure A.1(b). The chain of SFFs is also referred to as a scan chain. There are various types of scan cells, e.g., multiplexed scan cell, double latched scan cells, level



(a) Generic core view (b) Core with embedded scan

Figure A.1. Scan based DFT

sensitive scan latches [36, 38]. The one illustrated in the figure and considered during this dissertation is the multiplexed scan cell. The multiplexed scan cell comprises a multiplexer and a FF. The SFF has two data inputs: the pseudo-output and the scan input, and two data outputs: the pseudo-input and the scan output; a control signal (tc) and the clock signal (clk). The chain of SFF is constructed by connecting the scan output of one SFF to the scan input of the following cell as illustrated in the figure. Applying a test vector using scan based DFT implies three steps: (i) scanning-in the test vector (i.e., the tc is set to 1, the data is loaded from the scan input si_1 into the scan chain formed by SFFs in ffs clock cycles); (ii) capturing the circuit response (i.e., tc is set to 0, the data is loaded from the pseudo-outputs of the CLB in 1 clock cycle); and (iii) scanning-out the test responses (i.e., similar to scan-in where the circuit responses are scanned-out). It should be noted that the last step is performed simultaneously with the first step, where new control values are loaded. The amount of time needed to perform the above procedure for all the test vectors in a test set is referred to as test application time (TAT).

Scan forms the foundation on which most other DFT techniques are built. Scan based DFT increases the controllability and observability of the device, making the core more testable, at the expense of penalties in area overhead and performances. Considering the area overhead and performance penalties as constraints leads to partial scan designs, where only part of the FFs are transformed into SFFs [36, 38]. Since, with the increase in the length of scan chains, the TAT increases as well another scan based DFT variant is multiple scan chains (MSCs) based design [36, 38]. A MSC design is illustrated in

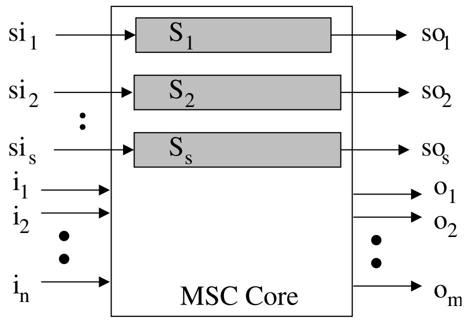


Figure A.2. Core with MSC

Figure A.2. The long SSC was split into *s* smaller scan chains. In order to keep the figure simple, the scan chains are represented as bars. In the figure the core has *n* inputs, *m* outputs and *s* scan chains $S_1 \ldots S_s$. Each scan chain has its own scan-input and scan-output. For example, si_1 and so_1 are the input and output for scan chain S_1 respectively. If all the scan chains have equal length, then they are referred to as balanced scan chains. Otherwise, they are referred to as unbalanced scan chains. Having more scan chains, more data can be loaded in parallel, and hence the TAT is reduced. It is assumed in this thesis that cores come with both single and multiple scan chains DFT.

Another issue raised with scan based DFT is the scan power dissipation, i.e., the power dissipated during shifting the data in and out from the scan chain. As can be seen in Figure A.1 the output of the FF is connected directly to the pseudo-input of the CLB. Hence, whatever data is loaded into the FF will be also available at the pseudo-inputs of the CLB. If, for example, in two consecutive clock cycles, the data from pi_1 is “0” and then “1”, this will cause a transition, from “0” to “1”, at this pseudo-input which can cause further transitions in the CLB. These transitions cause power dissipation in the CLB. Due to the high number of circuit nodes where a transition can occur and due to the long scan chains in complex designs, the scan power dissipation becomes an increasingly important parameter which directly influences yield. This is because chip packaging was designed considering functional requirements in mind, long periods of high switching activity will lead to chip over heating, and hence, to destructive testing. The yield loss leads to cost increase, since good chips are then thrown away. In the following paragraph BIST DFT is addressed.

BIST based DFT The main idea behind BIST is to have a test pattern generator (TPG) providing a set of test stimuli to the CUT, and a signature analyser (SA) computing a

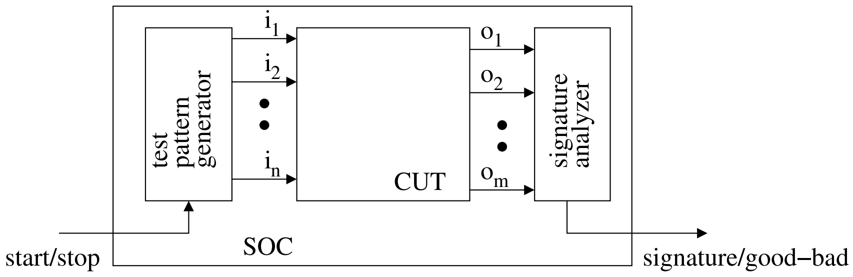


Figure A.3. Core with BIST

signature, or a good/bad decision, from the test responses. A typical BIST structure is illustrated in Figure A.3. Both, the TPG and the SA are on-chip. The TPG generates, for a number of clock cycles, a set of test stimuli with random properties. These are then applied to the CUT, and the test responses are analysed using the SA. If the computed signature differs from a fault free signature, then a fault was identified. The most common BIST setup is to use a linear feedback shift register (LFSR) as a test pattern generator, and a single input signature analyser (SISR) or a multiple input signature analyser (MISR) as a SA [36, 38]. There are essentially two types of BIST DFT: scan based BIST and embedded BIST DFT. Scan based BIST, uses the BIST methodology for the scan-inputs and scan-outputs of the core. Embedded BIST uses the BIST methodology within the core (mainly on the data-path of a design, i.e., the part of the design compounded of adders, multipliers) and at its inputs and outputs. A core which comes with BIST can be either “BIST-ready” or “BIST-ed”. The former implies that the core provider prepared the core for BIST. The latter is a BIST-ready core where the TPG and SA are embedded within the core. If not stated differently, BIST will refer to scan based BIST as introduced above. With respect to the type of test applied to the core, BIST can be pseudo-random or deterministic. The former implies that the TPG is allowed to run freely for a given number of clock cycles; the latter implies that the TPG is controlled using techniques such as “bit-flipping” [100, 155], “bit-fixing” [101], “re-seeding” [156] and “pattern-mapping” [157] in order to generate deterministic test vectors [98, 104, 36, 38]. When the two are combined, mixed-mode BIST solutions are derived. It should be noted that, unless stated differently, the term deterministic BIST will also refer to mixed-mode BIST.

Appendix B

VIHC decoder’s implementation

In this appendix gate level implementation details for the Variable-length Input Huffman Coding (VIHC) scheme provided in Chapter 3, including the synthesis and gate level simulation processes, are given. The tools and their employment in the implementation process are detailed below:

- firstly, the behavioural described VHDL code has been synthesised using Synopsys Design Compiler [133] for the Alcatel 0.5μ technology library (MTC35000);
- secondly, the gate level VHDL has been simulated using the ModelSim HDL Simulation tool [158].

As noted in Section 3.3.1, the proposed VIHC decoder has two parts: the Huffman decoder (Huff-decoder) used to identify the Huffman code and the control and generation unit (CGU) used to generate the current pattern and to synchronise with the Huff-decoder. A schematic of the decoder comprising these two parts is given in Figure B.1. The two units have been implemented and simulated for Example 3.1 (see Page 41). Hence, the Huff-decoder has 4 states and the CGU has been designed for a group size of 4. The decoder has been synthesised for a clock period constraint of 50 *ns*. The simulation has been performed for a clock period of 50 and 100 *ns*. The frequency ratio (α) has been considered 2. *sys_clk* represents the internal chip test clock and *fsm_clk* represents the clock used to generate the clock for the Huffman FSM. The other signals in the figure will be detailed as necessary. The Huff-decoder is illustrated in Figure B.2 and the CGU in Figure B.3.

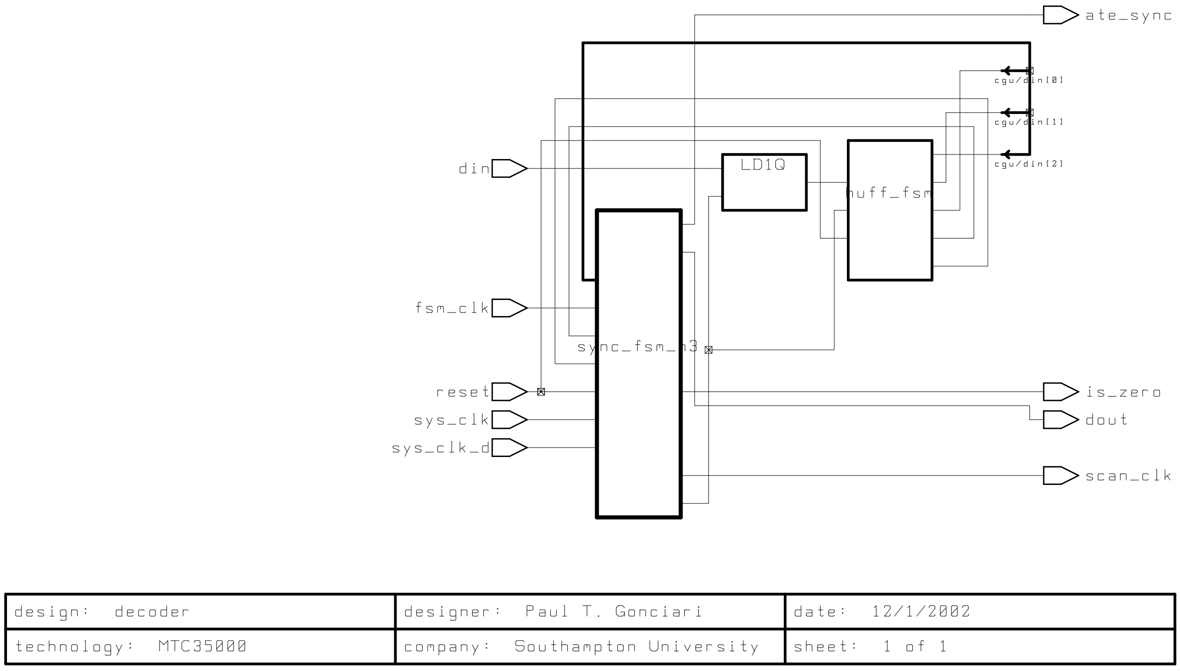
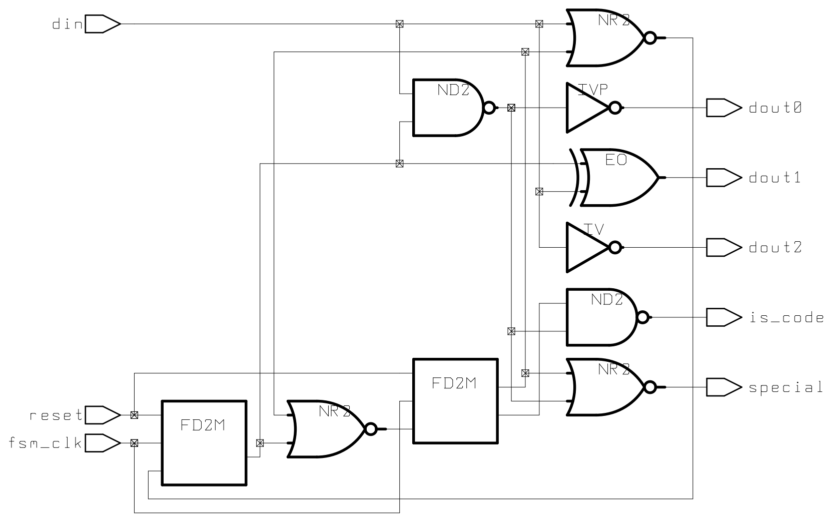


Figure B.1. VIHC decoder

The Huff-decoder has been implemented using a Mealy FSM (i.e., the output of the FSM is dependent on the current state and the input signal), which will guarantee that the output of the FSM will be available in the same *fsm_clk* clock cycle. In the case of the CGU, a normal counter has been used, and the load control FSM (see Figure 3.8 on Page 51) has been implemented using a Moore FSM (i.e., the output of the FSM is dependent only on the current state), hence the output of the FSM will be available only after one internal clock cycle. The counter has been driven using a non-overlapping clock derived from the chip test clock. The load control FSM has been driven using a clock delayed with 10 *ns* with respect to *sys_clk*. This delay will guarantee that the load control FSM will catch the *fsm_clk* on '1', hence changing the state from S_0 to S_1 (see Figure 3.8(c) on Page 51) when required.

In addition to the signals noted in Figure 3.8 (see Page 51) for the CGU, the *o_xx* signals have been used to capture internal signals easing the simulation and debug process. Also, in addition to the area overhead requirements given in Section 3.3.1, three more latches have been added to the design. One for the *din* signal coming from the ATE into Huff-decoder (see Figure B.1) and two for the *is_code* and *ate_sync* signals in the CGU unit (see Figure B.3). The first latch has been added to ensure that the output of the Huff-



design: huff_fsm	designer: Paul T. Gonciari	date: 11/30/2002
technology: MTC35000	company: Southampton University	sheet: 1 of 1

Figure B.2. Huff-decoder

decoder will not change whilst *ate_sync* is low, hence guaranteeing that the CGU will load the correct values once the current pattern has been generated. This latch is active when *sync_fsm_clk* is ’1’. The other two latches have been added to eliminate glitches. Glitches have been generated by the Huff-decoder whilst changing state and the counter while decrementing. The former has been eliminated by inserting a latch for the *is_code* signal active when *fsm_clk* is ’0’, leaving enough time for the output of the Huff-decoder to reach a stable state. For the latter, a latch was introduced on the *ate_sync* signal, also active when *fsm_clk* is ’0’, since the Huff-decoder must be halted only after the code has been identified. Hence, while parallel decoders have the advantage of working in two clock domains, the communication and synchronisation between these two domains must be accounted for, and therefore the addition of these latches. The advantage of having the two decoder units working in different clock domains is that it can decouple the ATE from the on-chip decoder – i.e., the *fsm_clk* signal is the one that drives the entire architecture; as long as *fsm_clk* is generated such that valid data is loaded into the decoder, data can be generated to the on-chip core with the on-chip test frequency. This is further detailed with the gate level timing diagram given next. Note that the schematics have been generated with the Synopsys Design Compiler.

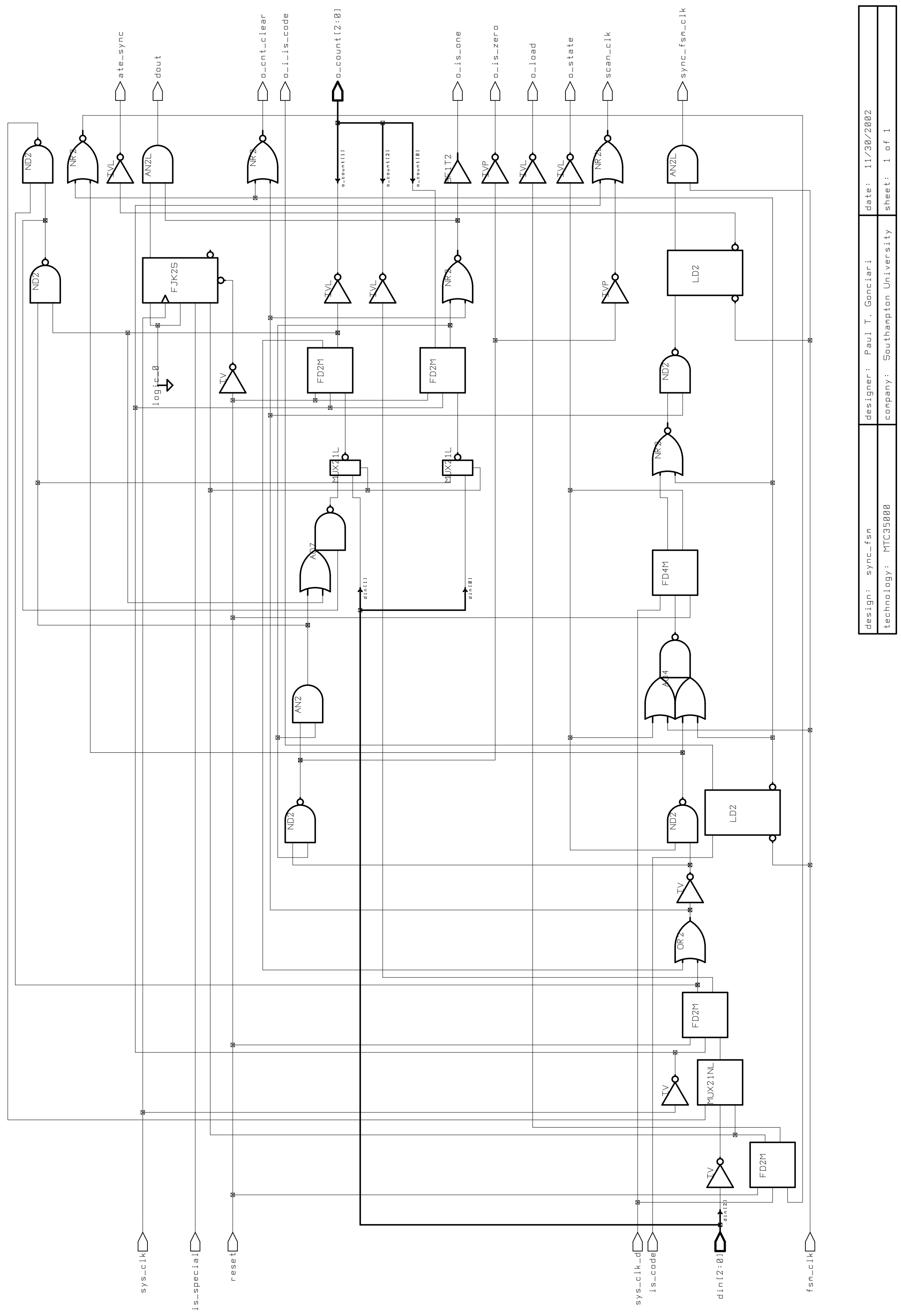


Figure B.3. Control and generation unit (CGU)

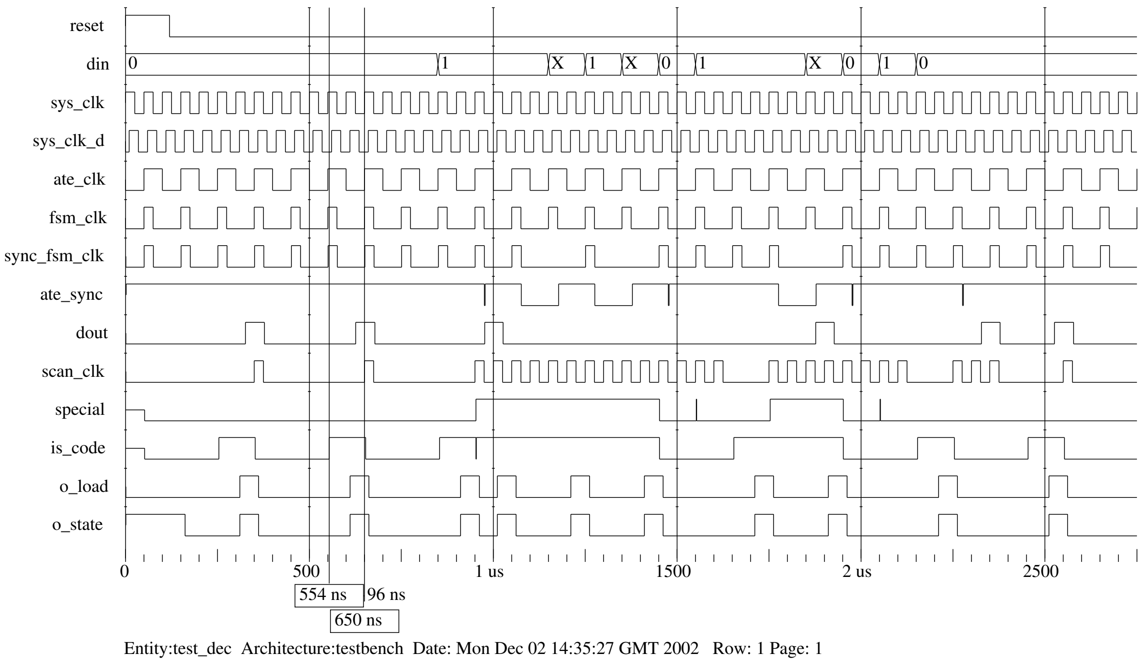


Figure B.4. Gate level timing diagram for the example given in Figure 3.2(d)

With the synthesised gate level design, a gate level simulation using ModelSim has been performed. The gate level timing diagram is given in Figure B.4. The figure corresponds to the simulation when the chip test clock period is 50 *ns*. To ensure that the Huff-decoder is in a stable state after reset, the code "000" has been fed into it. This brings the decoder in state "S0" (see Figure 3.7 on Page 51). After these three bits, the compressed test set from Example 3.1 (see Page 41) has been inserted. The first output bit, corresponding to the referred example, is at the seventh *fsm_clk* cycle. The difference between the moment when the codeword was identified and when the first bit of the pattern is generated is marked in the figure at 554 *ns* and 650 *ns*. It can be seen that the difference between the two is 96 *ns* which is $\approx$ 2 *sys_clk* cycles. Note that this corresponds to the timing diagram given in Figure 3.9 (see Page 53) where the CGU generates data after two on-chip clock cycles. It should be also noted that the *sync_fsm_clk*, the clock used to drive the Huff-decoder, is low when the CGU is not capable of loading the binary code, since it is busy generating the previous one. This is the time window between 1 and 1.5 *us*. Since in these clock cycles, the ATE is notified by the *ate_sync* signal to stop sending data, a "X" was assumed on *din* as illustrated in the figure. It can be also observed that there are some glitches on the *ate_sync* and *is_special* signals. These, however, will not affect the functionality of the unit, since they happen after the data on these lines has been used (i.e., the glitch on *ate_sync* is on the falling edge of *fsm_clk*, however, to affect the generation of the *sync_fsm_clk* it has to happen on the rising edge). Also plotted in the figure are the *o_load* and *o_state* signals from the CGU. These two are essentially the driving mechanism behind the decoder, as they ensure the synchronisation between the Huff-decoder and the CGU and correct pattern generation. Another important point is the dependence of the architecture on *fsm_clk* and not on the *ate_clk*. This leads to relaxing the timing requirements on the ATE, since the data can be captured anywhere in an ATE clock cycle.

Appendix C

Distribution architecture's implementation

Chapter 4 proposed a distribution architecture for synchronisation overhead reduction. In this appendix additional timing diagrams and the schematics for the distribution architecture are provided. The diagrams illustrated in Figure C.1 and Figure C.2 are for the compressed test sets t^0 and t^1 used in Section 4.2 (see Page 73). These have been provided to illustrate the occurrence of the stop clock cycles for the corresponding compressed test sets.

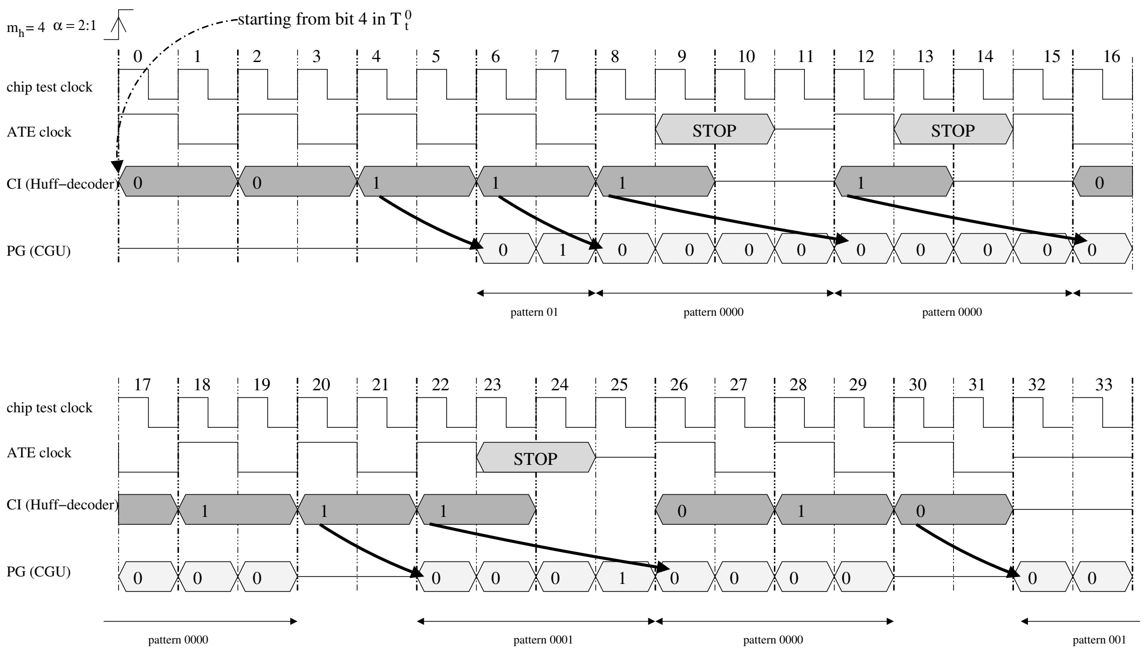


Figure C.1. Timing diagram to illustrate the stopping of the data stream for t^0

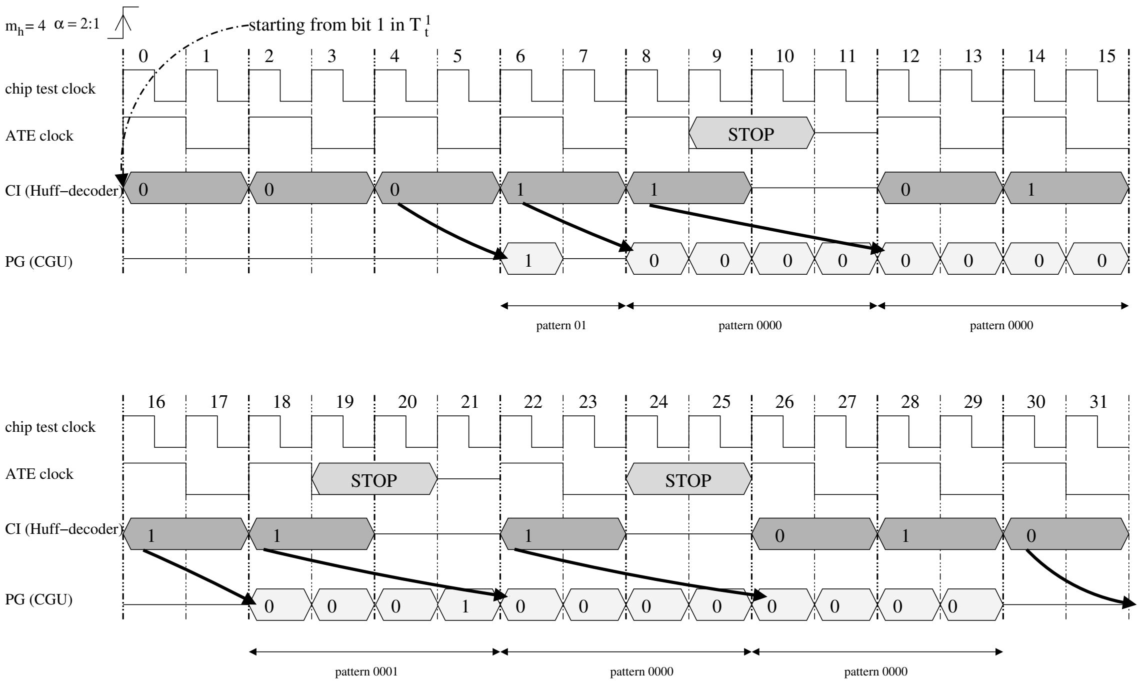


Figure C.2. Timing diagram to illustrate the stopping of the data stream for t^1

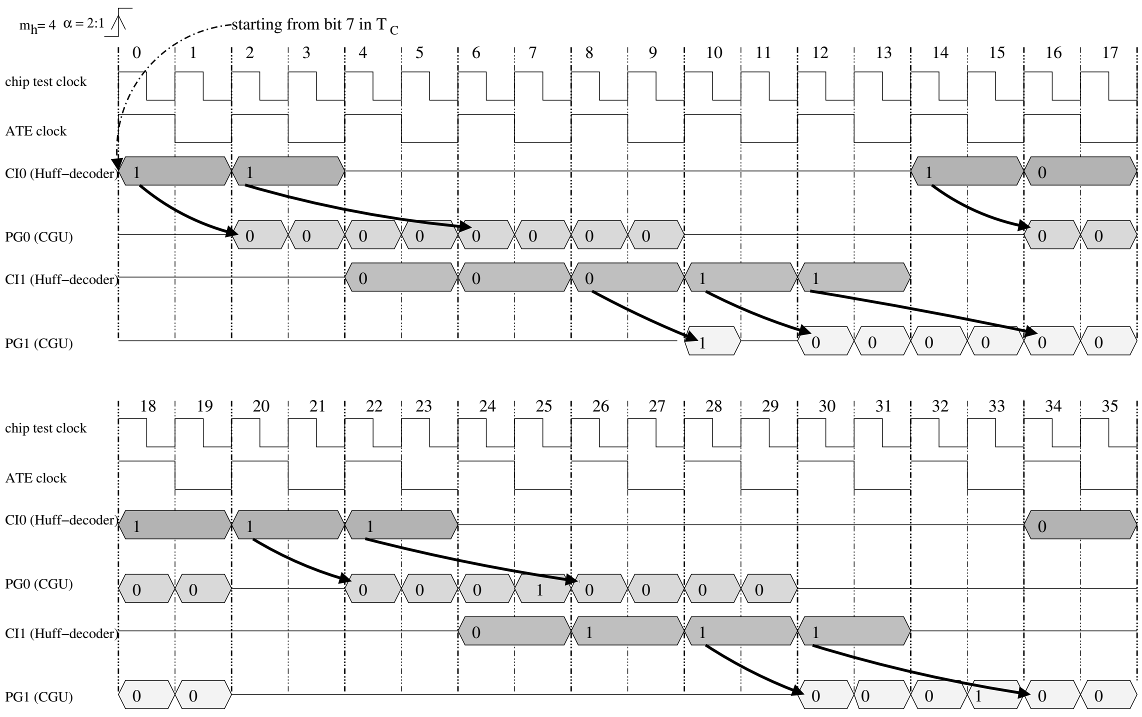
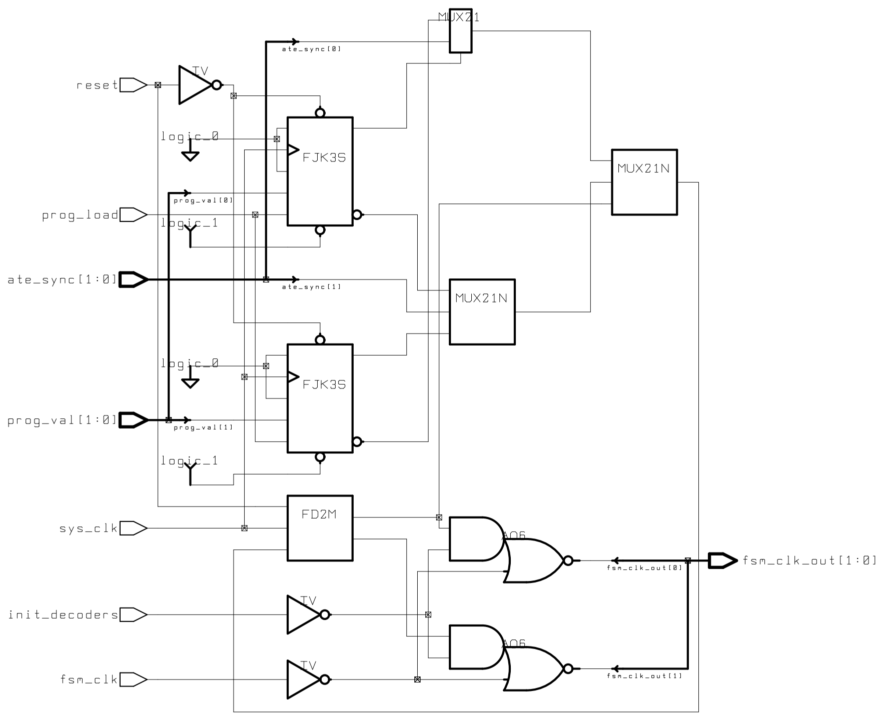


Figure C.3. Timing diagram to illustrate the distribution architecture and T_C



design: distr	designer: Paul T. Gonciari	date: 11/30/2002
technology: MTC35000	company: Southampton University	sheet: 1 of 1

Figure C.4. The distribution unit

To illustrate the changing of the active code identifier (CI) when the pattern generator (PG) is not capable of loading the newly detected code, a generic timing diagram for the distribution architecture is given in Figure C.3 considering the next time frame of interest from Example 4.2 (see Section 4.2.2 on Page 77)

Similar to Appendix B, the distribution architecture has been synthesised using Synopsys Design Compiler [133] and the Alcatel MTC35000 technology library, and the gate level design simulated using ModelSim [158]. The schematic for the distribution unit is given in Figure C.4, and the gate level timing simulation in Figure C.5. The schematic and the gate level simulation have been performed for the compressed test set given in Example 4.1 (see Page 75). The design has been synthesised for a clock period of 50 *ns* and simulated for clock periods of 50 and 100 *ns*. The timing diagram illustrated in Figure C.5 is for a clock period of 50 *ns*. The frequency ratio has been considered 2. The signal's notation is consistent with the one given in Appendix B. The *dout_bus*, *scan_clk_bus*, *ate_sync* and *fsm_clk_out* are the outputs of the two decoders, noted with (0) and (1), in the timing diagram, respectively. To ensure that all the decoders in the distribution archi-

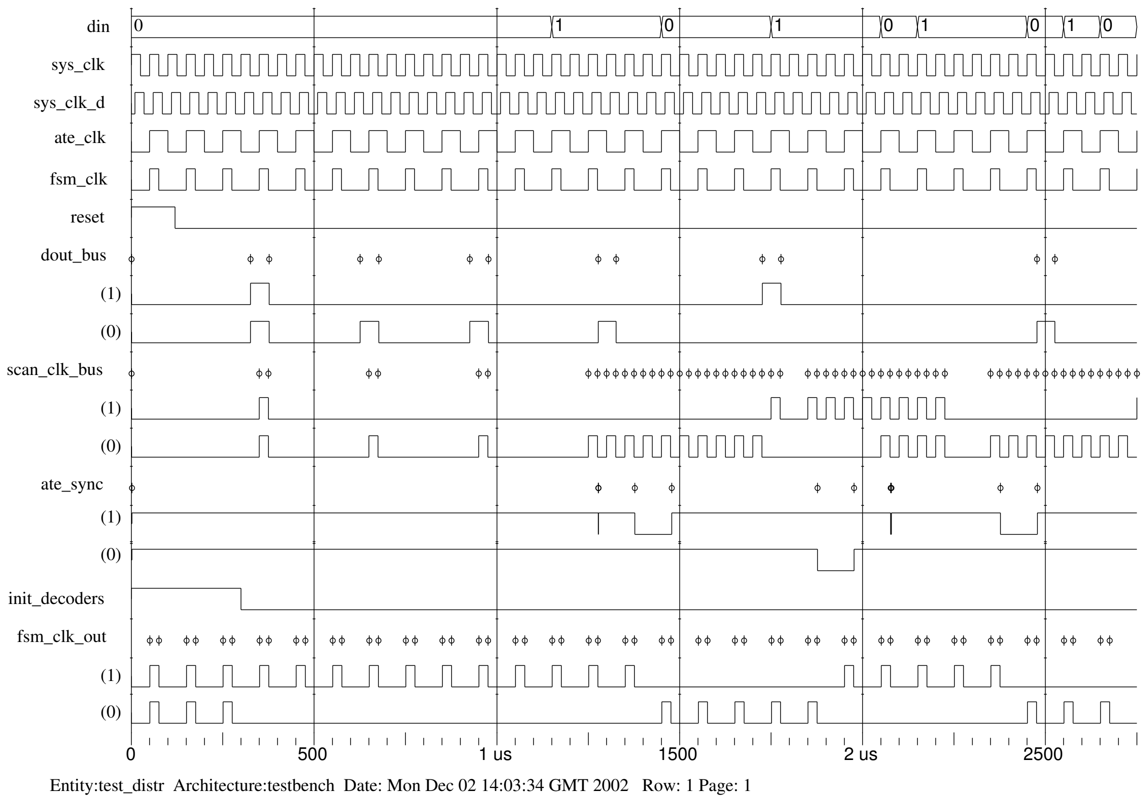


Figure C.5. Gate level timing diagram for the example given in Figure 4.11(d)

tecture reached a stable state before they start processing the composite test set, the signal *init_decoders* has been added to the distribution architecture. For the considered example, while this signal is active, all the decoders are receiving data. Similar to the Appendix B the codeword "000" has been fed into the decoders. This is illustrated in Figure C.5 for the first three *fsm_clk* cycles. Note that the changes in the *fsm_clk_out* lines, are consistent with the generic timing diagram provided in Figure C.3 (see the time frame between 1*us* and 1500*ns* for the first change in *fsm_clk_out* consistent with the change of active decoders in Figure C.3 at clock cycle 4).

Appendix D

Useless memory and test bus width relationship

In this appendix useful explications and additional results are provided for Chapter 5. Firstly, the reasoning behind relaxing the upper bound on the w_{max} is given. Secondly, an interesting particular case observed when the memory requirements obtained with **mUMA** with $np = 2$ are greater than FFD [56]. Finally, more results are given to illustrated the influence of pin-group granularity on the proposed **mUMA** when $np = 2$.

D.1 Upper bound on w_{max}

1

As noted in Section 5.5, the w_{max} determined using the formula provided in [57] ($w_{max}^{[57]} = \lceil \frac{\max\{n,m\}+\sum_{i=1}^{s} S_i}{\max\{S_i\}} \rceil$) will not guarantee that the minimum TAT is obtained for a core with fixed-length scan chains. In the above formula, s denotes the number of scan chains, S_i denotes the length of the scan chains, and n/m represent the number of inputs/outputs respectively. Note that, for the remainder of the section $\max\{S_i\}$ denotes $\max_{j=\overline{1,s}}\{S_j\}$, and w_{max}^1 denotes $w_{max}^{[57]}$. Also note that unless stated differently the term scan chains will refer to fixed-length scan chains, and it is assumed that the design has at least one scan chain.

[1]Some of the findings in this section have been derived based on a correspondence with Vikram Iyengar, the main author of [57]

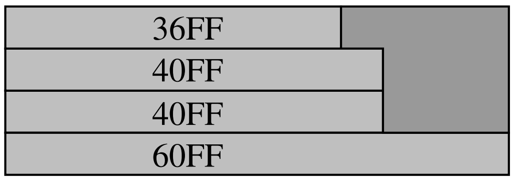


Figure D.1. Example of unequal scan chains

Considering a core with four scan chains of length $36, 40, 40$ and 60, four inputs and four outputs, $w^1_{max} = 3$. However, it can be seen that a minimum TAT of 60 is obtained for $w_{max} = 4$. Hence, w^1_{max} does not guarantee minimum TAT. It should be noted that as introduced in [57], w^1_{max} was the minimum test bus width for which the minimum TAT is obtained. For the experiments performed in Chapter 5 it was more important to ensure that the entire core wrapper solution space is explored, than to determine w_{max} such that it is the minimum test bus width for which the minimum TAT is obtained. In addition, the core wrapper design problem with fixed-length scan chains is NP-hard, hence, the minimum test bus width core wrapper design solution which attains minimum TAT is also NP-hard. Therefore, in the following the bounds for w_{max} are derived, i.e., the minimum and maximum values of w (test bus width) which ensure that minimum TAT is obtained.

It is clear that if flexible-length scan chains are available, then the bound given in [57] is correct. Actually, w^1_{max} denotes the lower bound for fixed-length scan chains, since it represent the best case scenario for fixed-length scan chains. Hence,

$$w^1_{max} \leq w_{max} \leq w^2_{max} \tag{D.1}$$

where w^2_{max} is the sought upper bound.

To determine w^2_{max}, in the following the worst case scenario with respect to test bus width is analysed. In the worst case scenario, the scan chains cannot be combined and the resulted core wrapper will have s WSCs with the TAT given by $\max\{S_i\}$. The question is how many more WSCs are required for the core's primary inputs. Starting from the above conditions, n primary inputs (assuming that $n \geq m$) must be distributed within the 'holes' generated by the s WSCs. This is illustrated in Figure D.1 for the core specifications given above, where the holes are marked with gray shades. The number of holes is given by the difference between the rectangle's area and the sum of scan chains.

Let's note

$$D = s \cdot \max\{S_j\} - \sum_{j=1}^{w} S_j \tag{D.2}$$

the number of holes. It is interesting to note that D is actually the useless memory as introduced for core wrapper designs in Section 5.4.1. Hence, if $n \leq D$, then no more WSCs are required to obtain the minimum TAT, which will then be $min_{TAT} = max\{S_j\}$. Hence,

$$w_{max}^2 = s, \text{when } n \leq D \tag{D.3}$$

However, if $n > D$, then $\lceil \frac{n-D}{\max\{S_j\}} \rceil$ WSCs must be added, in order to ensure a minimum TAT of $min_{TAT} = max\{S_j\}$.

Hence,

$$w_{max}^2 = s + \lceil \frac{n-D}{\max\{S_j\}} \rceil, \text{when } n > D \tag{D.4}$$

The above equation transforms as follows:

$$\begin{aligned} w_{max}^2 &= s + \lceil \frac{n-D}{\max\{S_j\}} \rceil \leq s + \frac{n-D}{\max\{S_j\}} + 1 \\ &\leq \frac{n + \sum_{j=1}^{w} S_j}{\max\{S_j\}} + 1 \end{aligned} \tag{D.5}$$

Hence, to summarise,

$$\begin{aligned} & w_{max}^1 \leq w_{max} \leq s && \text{if } \max\{n,m\} \leq D, \\ & w_{max}^1 \leq w_{max} \leq \frac{\max\{n,m\} + \sum_{j=1}^{w} S_j}{\max\{S_j\}} + 1 && \text{if } \max\{n,m\} > D. \end{aligned} \tag{D.6}$$

Interesting to note that in the second case, if $\frac{n+\sum_{j=1}^{w} S_j}{\max\{S_j\}}$ is an integer, then $w_{max} \in \{w_{max}^1, w_{max}^1 + 1\}$, otherwise $w_{max} = w_{max}^1$. As also noted in Section 5.5, it was considered that $w_{max} = \lceil \frac{\max\{n,m\} + \sum_{i=1}^{s} S_i}{1/s \cdot \sum_i^s S_i} \rceil$ since this formula comprises both cases.

D.2 UMA($np = 2$) vs. conventional ATE

It was illustrated in Section 5.5, that in some cases, the BFD [56] heuristic requires less test bus lines than the FFD [57] heuristic to obtain the same TAT. Hence, if the empty WSCs are not taken into account, the memory requirements are reduced. Throughout

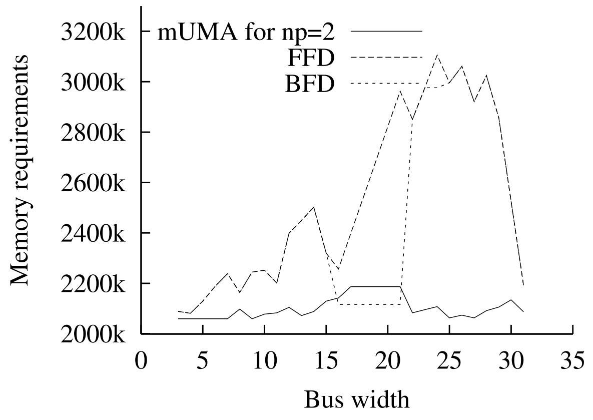


Figure D.2. Comparison between FFD [56], BFD [57] and mUMA with $np = 2$ for core Module26 of SOC p22810 [135]

the performed experiments one core has been identified for which using FFD can lead to smaller memory requirements than using **mUMA** for $np = 2$. This is illustrated in Figure D.2 where the memory requirements are given for FFD [56], BFD [57] and **mUMA** for $np = 2$. The figure shows that for test bus widths between 16 and 21, the memory requirements obtained by discarding the empty WSCs after employing the FFD heuristic are smaller than the ones obtained using the proposed **mUMA** heuristic for $np = 2$. However, this is a particular case as it can be seen in the figure, which has been observed only for Module26 from SOC p22810 [135]. Nevertheless, if the variation in memory requirements with the test bus width is compared, the proposed **mUMA** leads to less than 200k memory requirements variation, while the FFD and BFD heuristics lead to variations of up to 1000k. Hence, as also emphasised in Chapter 5 considering WSC partitioning in the core wrapper design process reduces the trade-off between test bus width and memory requirements and consequently between memory requirements and TAT.

D.3 ATE pin-group granularity constrainted mUMA

Section 5.5 (see Page 105) showed that when ATE pin-group granularity is considered, the performances of the **mUMA** heuristic may be affected. In the following more experimental data is presented to illustrate that the influence of pin-group granularity is small, and in some cases negligible.

The results for the cores specified in Table 5.1 and 5.2 (see Section 5.5 Page 105)

Core	$w=8$		$w=16$			$w=32$			
	min	$g=4$	*min*	$g=4$	$g=8$	*min*	$g=4$	$g=8$	$g=16$
s13207	163333	164964	163100	164032	171488	–	–	–	–
s15850	57434	57528	57434	57904	57904	–	–	–	–
s35932	21216	22656	21156	21696	22656	21216	21216	21696	22656
s38417	113288	114784	113288	113696	114784	113152	113152	113152	115328
s38584	161700	172920	161040	164120	173360	161700	162800	170720	186560

Table D.1. ATE pin-group granularity constrained WSC partitioning for ISACS89 benchmarks circuits [132]

Core	$w=8$		$w=16$		
	min	$g=4$	*min*	$g=4$	$g=8$
SOC p22810					
Module1	953775	1039340	–	–	–
Module21	574275	574740	–	–	–
Module26	2148289	2169104	2193358	2215440	2215440
SOC p34392					
Module2	4636794	4675344	4636794	4669176	4675344
Module10	2206440	2206440	2591886	2827512	2923760
Module18	5055570	5152420	–	–	–
SOC p93791					
Module6	5292604	5408144	5292604	5408144	5408144
Module20	3186560	3186560	3186560	3186560	3354624
Module27	2865248	2879904	2865248	2912880	2923872

Table D.2. ATE pin-group granularity constrained WSC partitioning for ITC02 benchmarks circuits [135]

are tabulated in Tables D.1 and D.2 respectively. The tables list, for each core, the minimum memory requirements (*min*) obtained using **mUMA** for $np = 2$, and the memory requirements obtained employing **mUMA** for $np = 2$ and considering different granularities ($g = 4, 8$ and 16) for test bus widths (w) of 8, 16 and 32. In the case of ISCAS89 benchmark circuits, cores s5378 and s9234 reach minimum TAT when using a test bus width of 6 and 5 respectively (see Table 5.1 Page 106). Therefore, they were not considered in this experiment. Also, a "–", in the tables, indicates that the minimum TAT was obtained for a smaller test bus width than the one given by the column header. Therefore, the corresponding test bus widths were not considered in the experiments. Analysing the results, it can be observed that the influence on the memory requirements is small. However, it tends to increase when the ATE pin-group granularity is increased. For example, in the case of core s15850 in Table D.1, for a test bus of $w = 8$, considering a granularity of $g = 4$ will increase the memory requirements with only 0.16% when compared to the minimum memory requirements obtained for this test bus width with **mUMA** for $np = 2$

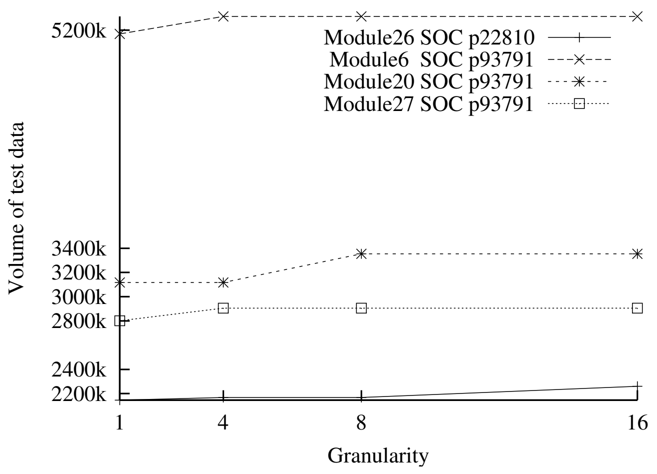


Figure D.3. Volume of test data variation with granularity for $w = 32$ for ITC02 benchmark circuits [135]

(columns 2 and 3 in the table). Similarly, in the case of core s38417 considering a test bus of $w = 16$ and a granularity of $g = 8$, the memory requirements are increased with 1.32% when compared to the memory requirements obtained using **mUMA** for $np = 2$ with no granularity constraint. It is interesting to note that for core s38417 for a test bus of $w = 32$, there is no increase in memory requirements when $g = 4$ or $g = 8$ are considered. A similar behaviour is observed for the ITC02 systems. For example, for core Module10 from SOC p34392 (see Table D.2), considering a granularity of $g = 4$ for $w = 8$ does not increase the memory requirements for this core. In general it can be observed that with the increase in granularity, the volume of test data tends to increase. However, this increase is usually small. This is best illustrated in Figure D.3, where the volume of test data variation with the granularity for a test bus width of 32 is given. For Module27 from SOC p93791, for example, for $w = 32$, when the granularity is $g = 4, 8$ or 16, the increase in memory requirements is 3.57% when compared to the minimum memory requirements for this test bus width obtained with **mUMA** for $np = 2$ (reported in the figure for a granularity of 1). Hence, considering the ATE pin-group granularity constraint in the core wrapper design has small or even no influence on the memory requirements.

Appendix E

Extended distribution architecture's implementation

Chapter 6 proposed an integrated test solution for core based SOC. This appendix provides additional useful explications and experimental results. These include scan power dissipation details in Section E.1; the implementation details of the *XOR-network* architecture and addition results for the multiple scan chains (MSCs) comparison with the proposed *add-on-xDistr*, in Section E.2; and implementation details and gate level timing diagrams for the *xDistr* in Section E.3.

E.1 Power dissipation during scan

Power dissipation in digital CMOS circuits is caused by four sources: the leakage current (determined by the fabrication technology); the standby current, which is the DC current drawn continuously from V_{dd} to ground; the short-circuit (rush-through) current, which is due to the DC path between the supply rails during output transitions; and the capacitance current, which flows to charge and discharge capacitive loads during logic changes [159]. The dominant source of power dissipation CMOS circuits is the charging and discharging of the node capacitances and it is given by [159]:

$$P = 0.5 \cdot C_L \cdot V_{dd}^2 \cdot E(sw) \cdot f_{chip} \tag{E.1}$$

where C_L is the physical capacitance at the output of the node, Vdd is the supply voltage, $E(sw)$ (referred to as the switching activity) is the average number of output transitions per $1/f_{chip}$ time, and f_{chip} is the clock frequency. The product of $E(sw)$ and f_{chip}, which is the number of transitions per second, is referred to as the transition density. In the following it will be considered that $\lambda = 0.5 \cdot C_L \cdot V_{dd}^2$ for convenience.

As illustrated in Appendix A (see Page 151), with respect to scanning of test data, the switching activity is due to shifting values into the scan chain. Therefore, the number of switches within the scan chain can be correlated to the power dissipation during scan as follows. If *ffs* represents the length of the scan chain, then at any given moment, there can be maximum *ffs* scan cells switching due to the loading and unloading of scan data. Hence, the maximum scan power is given by:

$$P_i = \lambda \cdot \text{ffs} \cdot f_{chip} \tag{E.2}$$

xDistr power reduction over single scan chain TDC In the following the reduction in power dissipation obtained with the *xDistr* architecture over a single scan chain test data compression (TDC) method, when both approaches drive the scan chain with the same frequency is given.

As noted in Section 6.2 (see Page 124), in the worst case scenario there are maximum two wrapper scan chains (WSCs) active. Considering the total length of the scan chain l, the length of the first partition l_1 and the length of the second one l_2, the scan power reduction is given by:

$$P_r \;=\; \frac{P_{ssc}}{P_{xDistr}} = \frac{\lambda \cdot l \cdot f_{chip}}{\lambda \cdot (l_1 + l_2) \cdot f_{chip}} = \frac{l}{l_1 + l_2} \tag{E.3}$$

Assuming that there are w WSCs, all of length m, the power reduction is computed as $P_r = \frac{m \cdot w}{m+m} = \frac{w}{2}$.

xDistr power reduction over XOR-network TDC To illustrate the reduction attainable using the proposed *xDistr* architecture over the *XOR-network* architecture, consider that the frequency ratio between the on-chip test frequency and the ATE operating frequency is $\alpha = \frac{f_{chip}}{f_{ate}}$, and that both architectures drive a core wrapper having w WSCs of length m. As also emphasised in Section 6.1.2, the *XOR-network* architecture uses the ATE clock (f_{ate}) to drive the core's WSCs. Using the same reasoning as above, the reduction in power dissipation is given by:

$$P_r = \frac{P_{XOR-network}}{P_{xDistr}} = \frac{\lambda \cdot w \cdot m \cdot f_{ate}}{\lambda \cdot (m+m) \cdot f_{chip}} = \frac{w}{2 \cdot \alpha} \quad \text{(E.4)}$$

E.2 XOR-network based architecture

The XOR-network architecture based on the method proposed in [114] is illustrated in Figure E.1 for a core with 4 WSCs. The main idea behind the method is to use one data signal to stream data into *SR*, which will then justify through the XOR-network the care bits into the WSCs. However, the architecture can run into temporal pattern lockout, i.e., the inability to justify the WSC care bits at a given moment – hence one control signal to halt the WSC load temporarily is needed. Therefore, the architecture requires two ATE channels. In addition, care must be taken to avoid structural pattern lockout, i.e., when there are no *SR* values such that the XOR-network outputs will justify the correct care bits into the WSCs – since this can affect fault coverage. To account for this problem and to tune the method for the chosen core based SOC environment, the XOR-network has been associated to a square matrix $A(w,w)$ with $det(A) \neq 0$. In addition in the implemented approach a shift register (*SR* in the figure) has been used, of size w. The two combined, guarantee that any pattern can be justified to the inputs of the WSCs regardless of the care bit density. Hence, the implemented architecture can also be used from the system integrator's perspective at the system integration level in IP based SOCs. It is important to note that the architecture implemented as illustrated above does not replicate the method as proposed in [114], but rather it adapts its main idea and tunes it for core based SOC test. Next the test data compression process implemented for the XOR-network based architecture is detailed.

Firstly, to ensure that structural pattern lockouts are avoided, a prim polynomial of degree w has been chosen, and the attached matrix (M) [38] has been constructed. Secondly, the matrix has been raised to the power of w. The obtained matrix is now the XOR-network attached matrix ($A = M^w$). The values to be shifted in the *SR* and the time

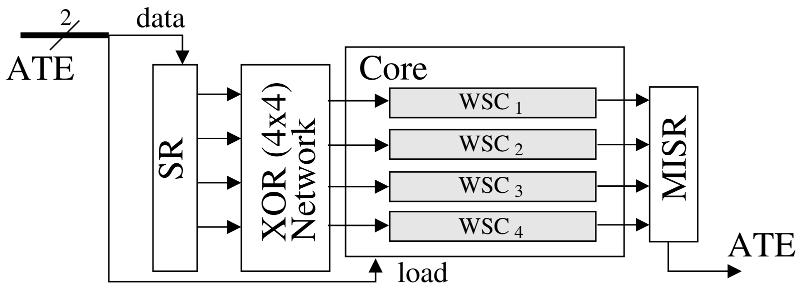


Figure E.1. XOR-network based architecture [114]

intervals when the *load* signal needs to halt the load of data into the WSCs has been determined by solving the linear equations using Gauss-Jordan elimination when the *SR* values represent unknowns, i.e., $SR \times A = W$, where W represents the values to be loaded into the WSCs. This process is illustrated in the following example.

Example E.1 Consider a 4-stage LFSR with the characteristic polynomial given by $h(x) = x^4 + x^3 + 1$. The corresponding attached matrix will be

$$M = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Raising the matrix to the power of 4 will give

$$A = M^4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

Assume $W^1 = [10xx]$ and $W^2 = [00x0]$, the values to be loaded into the WSCs into two consecutive load cycle, where x stands for don't care. The corresponding systems of equations are then:

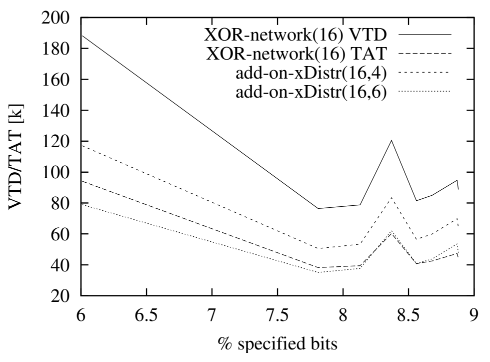
$$\begin{bmatrix} SR_1^1 & SR_2^1 & SR_3^1 & SR_4^1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x & x \end{bmatrix}$$

$$\begin{bmatrix} SR_1^2 & SR_2^2 & SR_3^2 & SR_4^2 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & x & 0 \end{bmatrix}$$

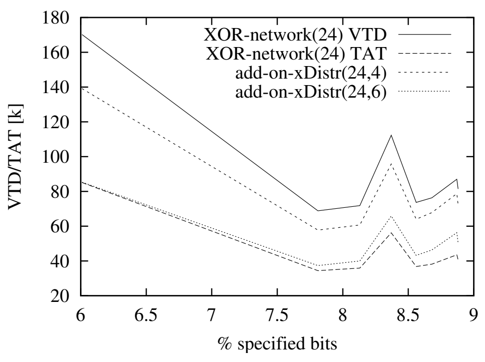
Solving the first system it can be found that there are multiple solutions. Let us consider $SR^1 = [1100]$. Starting from these initial values, after one shift the content of $SR = [SR_1^2\ 110]$. When the SR_i values are inserted in the second equation it can be observed that regardless of the value of SR_1^2 the system does not have a solution. Hence, one more shift is required, $SR = [SR_1^2\ SR_2^2\ 11]$. When the new SR_i values are inserted in the second equation the solution is $SR^2 = [1011]$. Hence, starting from SR^1 two shifts are required to justify WS^2 through the *XOR-network*.

To account for the above, a simple two step greedy heuristic has been implemented. Firstly, the equations have been solved using Gauss-Jordan elimination. Secondly, the smallest number of shifts from SR^1 to SR^2 has been determined such that the care bits in W^2 are justified through the XOR-network. Using Gauss-Jordan elimination will give an upper bound on number of shifts required to justify W^2 starting from any SR^1, hence reducing the computational time for this step. This is explained next. After the reduced matrix corresponding to the second equation is obtained, the maximum bit position in SR^2 used in the reduced Gauss-Jordan matrix represents the maximum number of shifts required to justify W^2 through the XOR-network. For the example given above, it can be derived that, the maximum number of shifts from any SR^1 to SR^2 is 4. The actual number of shifts was determined incrementally starting from SR^1, as shown in the example.

It should be noted that raising the attached matrix to the power of w has been performed to reduce the compressed test set size. While experiments have been performed for different powers of M, it has been found that choosing w leads on average to $w/2$ inputs controlling one output. This allows for increased control of the XOR-network outputs hence reducing the VTD. For example, for a test set with 16.35% care bits for core s38584 when $w = 16$, using attached matrices M^8 and M^{16} yielded compressed test sets of sizes 239076 bits and 143106 bits respectively. Hence, considering M^{16} as the XOR-network matrix leads to clearly smaller test sets.



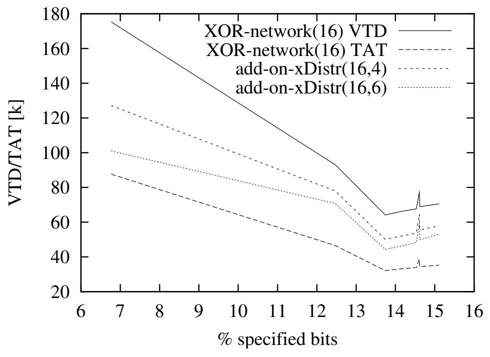
(a) $w = 16$



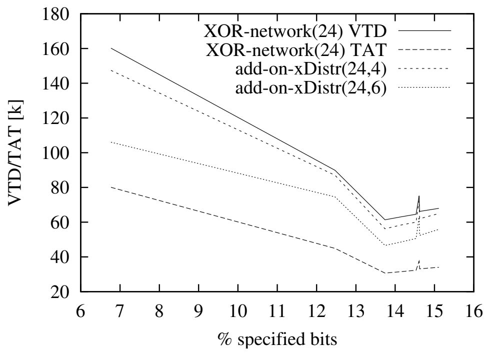
(b) $w = 24$

Figure E.2. VTD and TAT for s13207

Additional experimental results In the following additional experimental results for the comparison between *XOR-network* architecture and the proposed *add-on-xDistr* are presented. Figures E.2, E.3, E.4, E.5 illustrate the variation in VTD and TAT when the care bit density (%cb) increases. Analysing the figures, it can be observed that in almost all the cases the VTD values obtained with the proposed architecture are below the ones of the *XOR-network*'s ones. The TAT's however, can be also greater then the ones obtained by the *XOR-network* architecture. It should be noted though that the *XOR-network* architecture uses two ATE channels while *add-on-xDistr* uses one ATE channel. Hence, the TAT per channel is comparable and even smaller than the one obtained by the *XOR-network* architecture. Also note that the spikes in the plotted data at the end of the considered %cb interval is due to the heuristic used for controlled compaction. It seems that for certain cores the obtained test sets are larger in size but with a greater amount of specified bits, when controlled compaction for high values of B is performed (see Section 6.3.2 on Page 139).

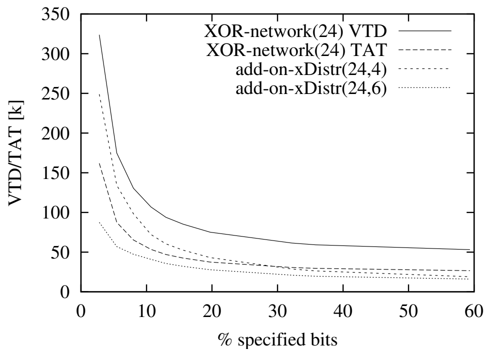


(a) $w = 16$

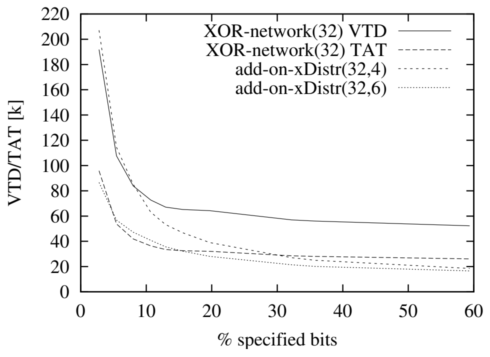


(b) $w = 24$

Figure E.3. VTD and TAT for s15850

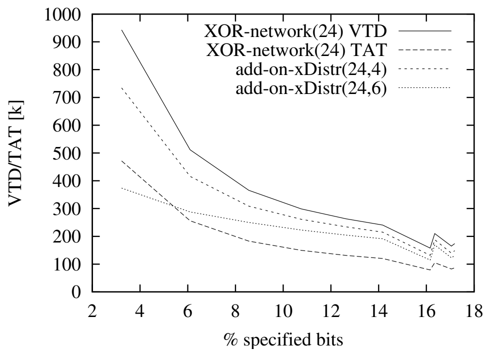


(a) $w = 24$

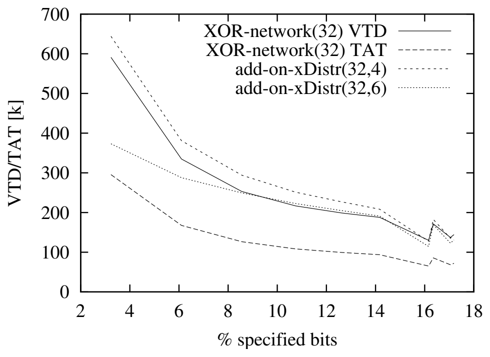


(b) $w = 32$

Figure E.4. VTD and TAT for s35932



(a) $w = 24$



(b) $w = 32$

Figure E.5. VTD and TAT for s38584

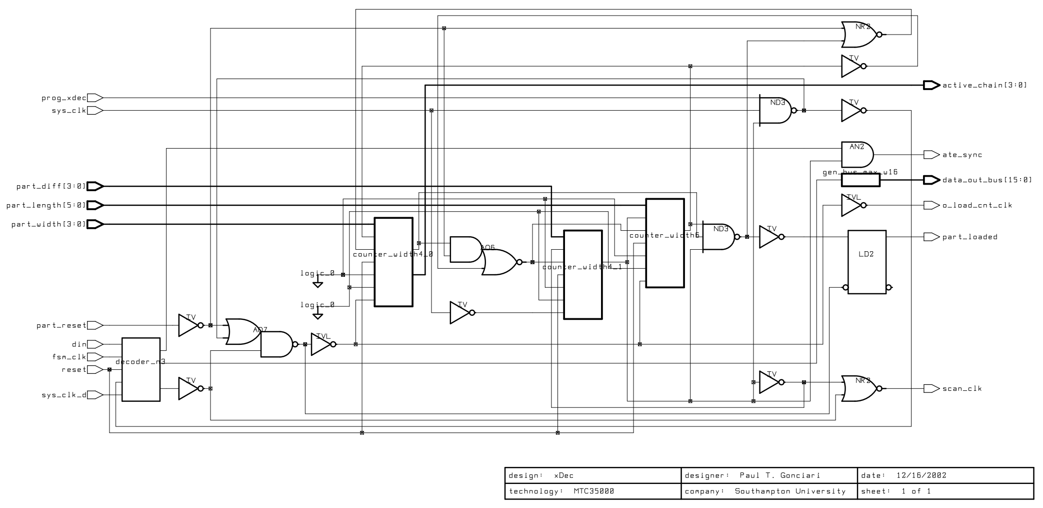


Figure E.6. The extended decoder (xDec)

E.3 Gate level implementation of add-on-xDistr

Similar to Appendixes B and C, the *add-on-xDistr* architecture has been synthesised using Synopsys Design Compiler [133] and the Alcatel MTC35000 technology library, and the gate level design simulated using ModelSim [158]. The schematic for the *xDec* is given in Figure E.6, and the schematic for the *add-on-xDistr* is given in Figure E.7. The implementation illustrated in this section is for Example 6.3 (see Page 127). The proposed *add-on-xDistr*, in Chapter 6, is programmable. Hence, it will require a programming step. This is illustrated with the gate level timing diagram in Figure E.8. The signal's notation is consistent with the one given in Appendix B. To provide this programmability feature shadow shift registers have been used. These are loaded while the *do_prog* signal is high. As also noted with the distribution architecture illustrated in Appendix C, the decoders required initialisations. This is ensured with the *init_decoders*. The functionality of the proposed *add-on-xDistr* after this programming step is illustrated with the gate level diagram in Figure E.9. For the example considered in Example 6.3 (see Page 127), the first partition had to account for the difference between the outputs and the inputs. This can be noted in the figure with the *xdecoder_1/part_diff* row, where for the first 200 *ns*, the corresponding counter decrements to 0. Also to note that the corresponding *ate_sync(1)* signal is low, hence the second partition is active.

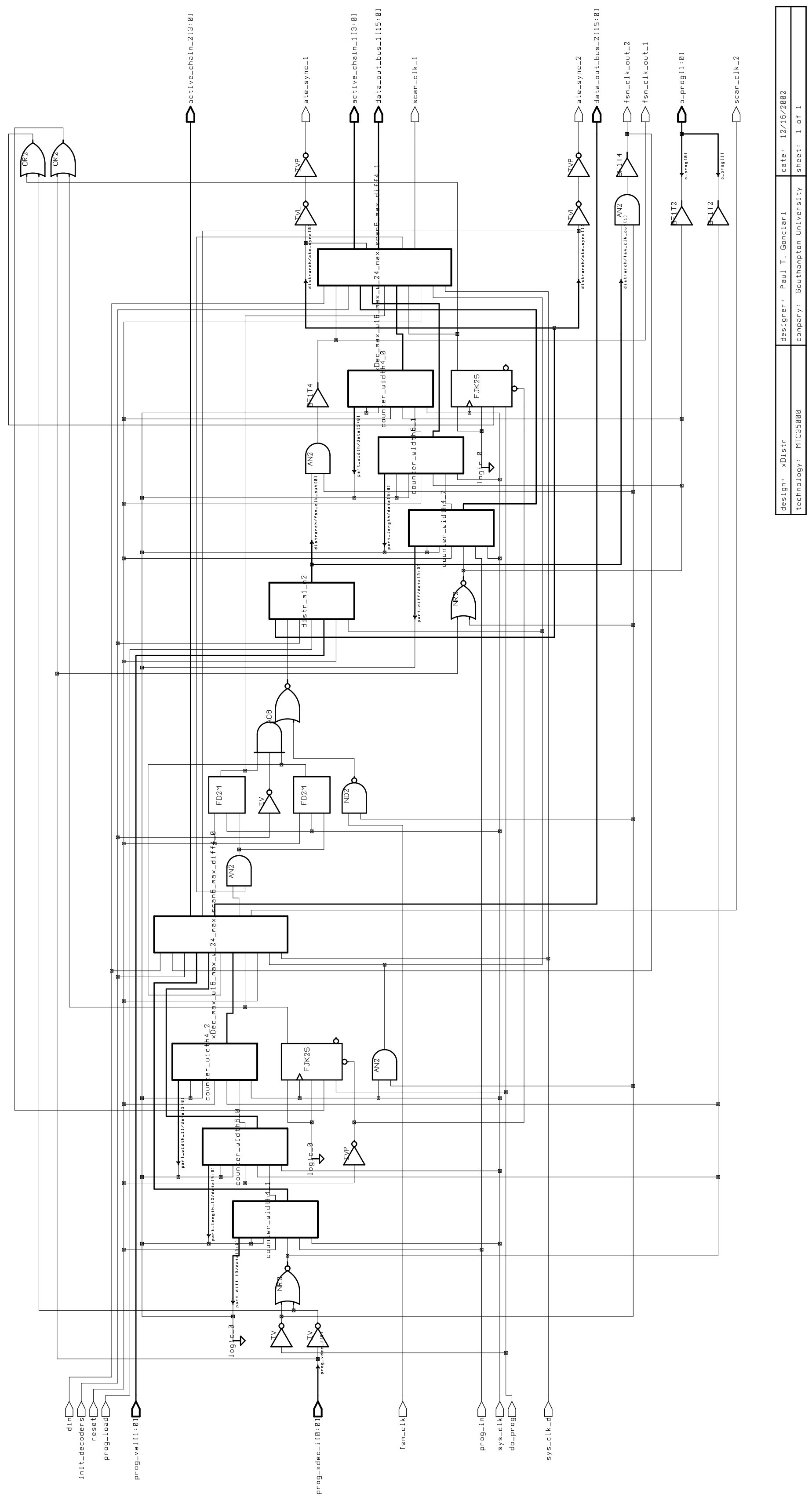


Figure E.7. The extended distribution architecture (add-on-xDistr)

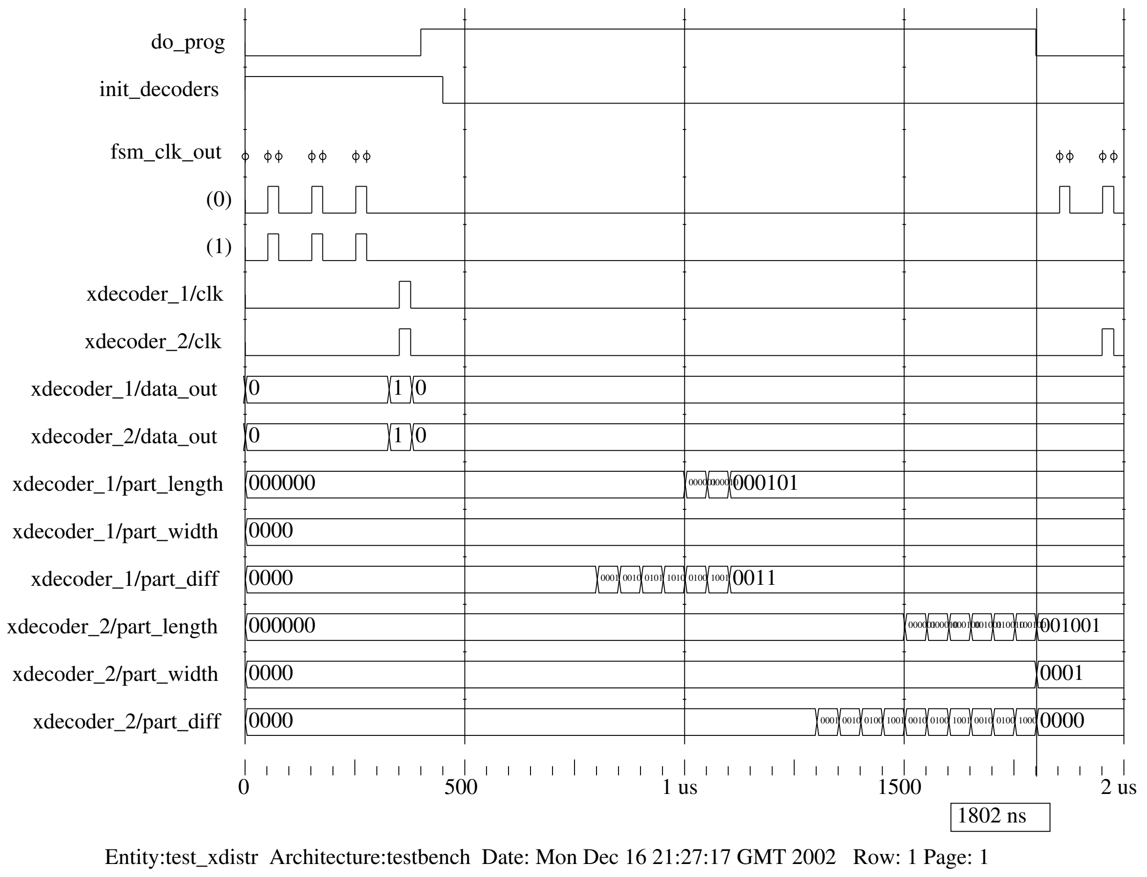
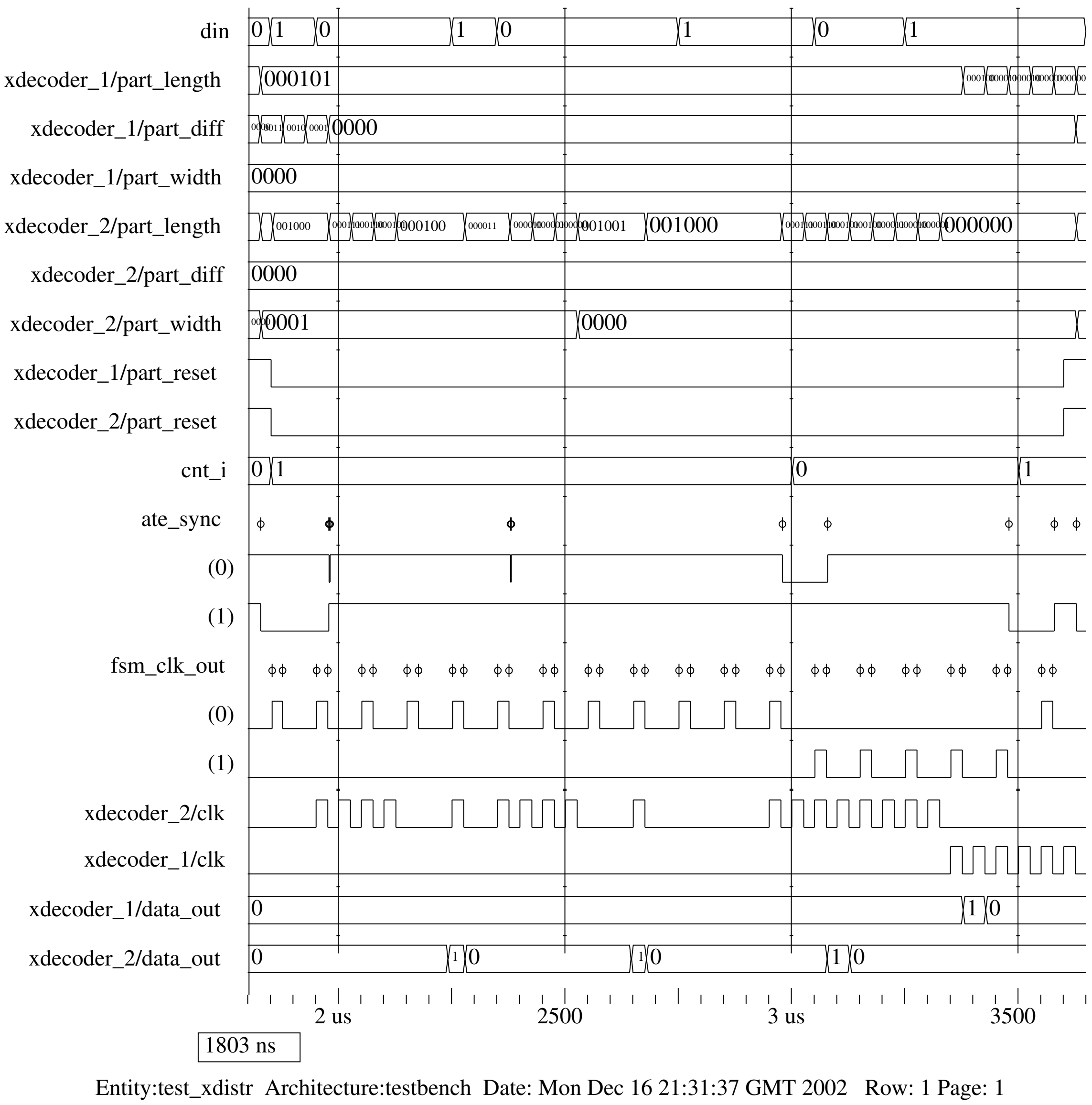


Figure E.8. Gate level timing diagram for programming the add-on-xDistr for the example given in Figure 6.9



Entity:test_xdistr Architecture:testbench Date: Mon Dec 16 21:31:37 GMT 2002 Row: 1 Page: 1

Figure E.9. Gate level timing diagram for the example given in Figure 6.9

Appendix F

Tools and benchmarks

This appendix describes tools and benchmarks used to generate the results in Chapters 3, 4, 5 and 6. The appendix is organised as follows: in Section F.1 the benchmark circuits are briefly described, and in Sections F.2 to F.5, flowcharts illustrating the implementation of the in-house tools, and their dependencies are given.

F.1 Benchmark circuits

Throughout the dissertation two types of benchmark circuits have been used: ISCAS89 [132] and ITC02 [135]. ISCAS89 are academic benchmarks, while ITC02 contain representation of both academic and industrial circuits. The specifications of the benchmarks are given in Tables F.1 and F.2 respectively. In Table F.1, for each of the ISCAS89 circuits the number of inputs (n), outputs (m), scan chains (s), the total number of flip flops (FFs) and the minimum and maximum scan chain length (min_{sc}/max_{sc}) are given. In addition, the table also lists the length of the MinTest [131] dynamically compacted and fully compacted test sets. The dynamically compacted test set has been used as input to the compression algorithm, while the size of the fully compacted test set has been used as reference, for the improvement in test application time (TAT) and volume of test data (VTD) as shown in Chapter 3 (see Section 3.4).

The ITC02 benchmarks circuits are briefly specified in Table F.2. The table lists the number of inputs (n), outputs (m), scan chains (s) and the total number of flip flops (FFs); the minimum and maximum scan chain length (min_{sc}/max_{sc}); the number of test vectors

Core	n/m	s	FFs	min_{sc}/max_{sc}	MinTest[131]	
					dyn. comp	fully comp
s5378	35/49	4	179	44/45	23754	20758
s9234	36/39	4	211	52/53	39273	25935
s13207	62/152	16	638	39/40	165200	163100
s15850	77/150	16	534	33/34	76986	57434
s35932	35/320	32	1728	54/54	28208	19393
s38417	28/106	32	1636	32/33	164736	113152
s38584	38/304	32	1426	44/45	199104	104111

Table F.1. Core specification for ISCAS89 [132] benchmarks

Core	$n/m/q$	s	FFs	min_{sc}/max_{sc}	n_v	mem
SOC p22810						
Module1	28/56/32	10	1122	110/113	785	927870
Module21	115/76/64	10	1054	93/186	465	578925
Module26	66/33/98	31	11485	198/400	181	2108469
SOC p34392						
Module2	165/263/0	29	8856	8/570	514	4636794
Module10	129/207/0	19	4731	16/519	454	2206440
Module18	175/212/0	14	6555	198/729	745	5013850
SOC p93791						
Module6	417/324/72	46	23789	500/521	218	5292604
Module20	136/12/72	44	7450	132/181	210	3185728
Module27	30/7/72	46	3026	50/68	916	2865248

Table F.2. Core specification for ITC02 [135] benchmarks

(n_v) used for testing and the amount of memory required (*mem*). These informations have been derived from the ITC02 benchmark suite, which in addition to the three SOCs noted in Table F.2 (p22810, p34392 and p93791 which represent Philips SOCs), comprises a number of academic and industrial SOCs. Since limited information about the cores are available, these benchmarks have been used only in Chapter 5.

The detailed description of the p22810 SOC is given in the listing given on the next page. As it can be observed, the SOC has 29 cores (*Modules*). Each module is described by the level (i.e., how deep is the core in the hierarchy, level 0 representing the SOC), the number of inputs (*Inputs*), outputs (*Outputs*), bidirectional pins (*Bidirs*), and scan chains (*ScanChains*), and the scan chains lengths. In addition, for each module the number of tests, the TAM and the patterns used for each test are also detailed. Based on these informations, the specifications given in Table F.2 have been determined. For a complete description of the ITC02 benchmarks the reader is referred to [135].

```
SocName p22810
TotalModules 29
Options Power 0 XY 0

Module 0 Level 0 Inputs 10 Outputs 67 Bidirs 96 ScanChains 0
Module 0 TotalTests 2
Module 0 Test 1 ScanUse 1 TamUse 1 Patterns 10
Module 0 Test 2 ScanUse 1 TamUse 1 Patterns 89

Module 1 Level 1 Inputs 28 Outputs 56 Bidirs 32 ScanChains 10 : 130 111 111 110 110 110 110 110 110 110

Module 1 TotalTests 1
Module 1 Test 1 ScanUse 1 TamUse 1 Patterns 785

Module 2 Level 2 Inputs 47 Outputs 33 Bidirs 0 ScanChains 0
Module 2 TotalTests 1
Module 2 Test 1 ScanUse 1 TamUse 1 Patterns 12324

Module 3 Level 2 Inputs 38 Outputs 26 Bidirs 0 ScanChains 0
Module 3 TotalTests 1
Module 3 Test 1 ScanUse 1 TamUse 1 Patterns 3108

Module 4 Level 2 Inputs 48 Outputs 64 Bidirs 0 ScanChains 0
Module 4 TotalTests 1
Module 4 Test 1 ScanUse 1 TamUse 1 Patterns 222

Module 5 Level 1 Inputs 90 Outputs 112 Bidirs 32 ScanChains 29 : 214 106 106 105 105 103 102 101 101 101
100 93 92 84 84 75 75 73 73 73 73 27 27 27 27 27 27 27 27

Module 5 TotalTests 1
Module 5 Test 1 ScanUse 1 TamUse 1 Patterns 202

Module 6 Level 2 Inputs 80 Outputs 64 Bidirs 0 ScanChains 0
Module 6 TotalTests 1
Module 6 Test 1 ScanUse 1 TamUse 1 Patterns 712

Module 7 Level 2 Inputs 84 Outputs 64 Bidirs 0 ScanChains 0
Module 7 TotalTests 1
Module 7 Test 1 ScanUse 1 TamUse 1 Patterns 2632

Module 8 Level 2 Inputs 36 Outputs 16 Bidirs 0 ScanChains 0
Module 8 TotalTests 1
Module 8 Test 1 ScanUse 1 TamUse 1 Patterns 2608

Module 9 Level 1 Inputs 116 Outputs 123 Bidirs 32 ScanChains 24 : 122 105 100 100 100 100 100 100 100 100
100 99 96 96 95 95 95 95 95 95 95 95 32 24

Module 9 TotalTests 1
Module 9 Test 1 ScanUse 1 TamUse 1 Patterns 175

Module 10 Level 1 Inputs 50 Outputs 30 Bidirs 0 ScanChains 4 : 99 98 10 2
Module 10 TotalTests 1
Module 10 Test 1 ScanUse 1 TamUse 1 Patterns 38

Module 11 Level 1 Inputs 56 Outputs 23 Bidirs 71 ScanChains 8 : 88 87 86 84 69 69 68 38

Module 11 TotalTests 1
Module 11 Test 1 ScanUse 1 TamUse 1 Patterns 94

Module 12 Level 1 Inputs 40 Outputs 23 Bidirs 71 ScanChains 11 : 82 82 64 64 64 64 64 63 63 62 42

Module 12 TotalTests 1
Module 12 Test 1 ScanUse 1 TamUse 1 Patterns 93

Module 13 Level 1 Inputs 68 Outputs 149 Bidirs 0 ScanChains 4 : 104 96 48 32
Module 13 TotalTests 1
Module 13 Test 1 ScanUse 1 TamUse 1 Patterns 1

Module 14 Level 1 Inputs 22 Outputs 15 Bidirs 0 ScanChains 3 : 73 4 1
Module 14 TotalTests 1
Module 14 Test 1 ScanUse 1 TamUse 1 Patterns 108

Module 15 Level 1 Inputs 84 Outputs 42 Bidirs 32 ScanChains 6 : 80 80 80 73 73 36
Module 15 TotalTests 1
Module 15 Test 1 ScanUse 1 TamUse 1 Patterns 37

Module 16 Level 1 Inputs 13 Outputs 43 Bidirs 72 ScanChains 1 : 109
Module 16 TotalTests 1
Module 16 Test 1 ScanUse 1 TamUse 1 Patterns 8

Module 17 Level 1 Inputs 223 Outputs 69 Bidirs 32 ScanChains 4 : 89 19 6 4
Module 17 TotalTests 1 ScanUse 1 TamUse 1 Patterns 25
Module 17 TotalTests 1 ScanUse 1 TamUse 1 Patterns 25

Module 18 Level 1 Inputs 53 Outputs 11 Bidirs 32 ScanChains 5 : 68 68 67 56 56
Module 18 TotalTests 1
Module 18 Test 1 ScanUse 1 TamUse 1 Patterns 644

Module 19 Level 1 Inputs 38 Outputs 29 Bidirs 0 ScanChains 3 : 43 40 17
Module 19 TotalTests 1
Module 19 Test 1 ScanUse 1 TamUse 1 Patterns 58

Module 20 Level 1 Inputs 45 Outputs 40 Bidirs 2 ScanChains 4 : 77 77 76 1
Module 20 TotalTests 1
Module 20 Test 1 ScanUse 1 TamUse 1 Patterns 124

Module 21 Level 1 Inputs 115 Outputs 76 Bidirs 64 ScanChains 10 : 186
108 104 94 94 94 94 94 93 93

Module 21 TotalTests 1
Module 21 Test 1 ScanUse 1 TamUse 1 Patterns 465

Module 22 Level 1 Inputs 54 Outputs 40 Bidirs 0 ScanChains 3 : 77 76 13
Module 22 TotalTests 1
Module 22 Test 1 ScanUse 1 TamUse 1 Patterns 59

Module 23 Level 1 Inputs 31 Outputs 8 Bidirs 35 ScanChains 7 : 115 92 18 16 16 16 16
Module 23 TotalTests 1
Module 23 Test 1 ScanUse 1 TamUse 1 Patterns 40

Module 24 Level 1 Inputs 73 Outputs 23 Bidirs 35 ScanChains 5 : 101 69 6 2 2
Module 24 TotalTests 1
Module 24 Test 1 ScanUse 1 TamUse 1 Patterns 27

Module 25 Level 1 Inputs 58 Outputs 46 Bidirs 86 ScanChains 18 : 181
139 139 139 139 138 138 138 138 138 138 109 108 108 108 108 108 108

Module 25 TotalTests 1
Module 25 Test 1 ScanUse 1 TamUse 1 Patterns 215

Module 26 Level 1 Inputs 66 Outputs 33 Bidirs 98 ScanChains 31 : 400
400 400 400 400 400 399 399 399 399 399 399 399 399 399 399 399 399
399 399 399 399 334 334 333 333 333 279 279 278 198

Module 26 TotalTests 1
Module 26 Test 1 ScanUse 1 TamUse 1 Patterns 181

Module 27 Level 1 Inputs 285 Outputs 94 Bidirs 0 ScanChains 1 : 34
Module 27 TotalTests 1
Module 27 Test 1 ScanUse 1 TamUse 1 Patterns 2

Module 28 Level 1 Inputs 48 Outputs 43 Bidirs 0 ScanChains 5 : 100 99
99 79 40

Module 28 TotalTests 1
Module 28 Test 1 ScanUse 1 TamUse 1 Patterns 26
```

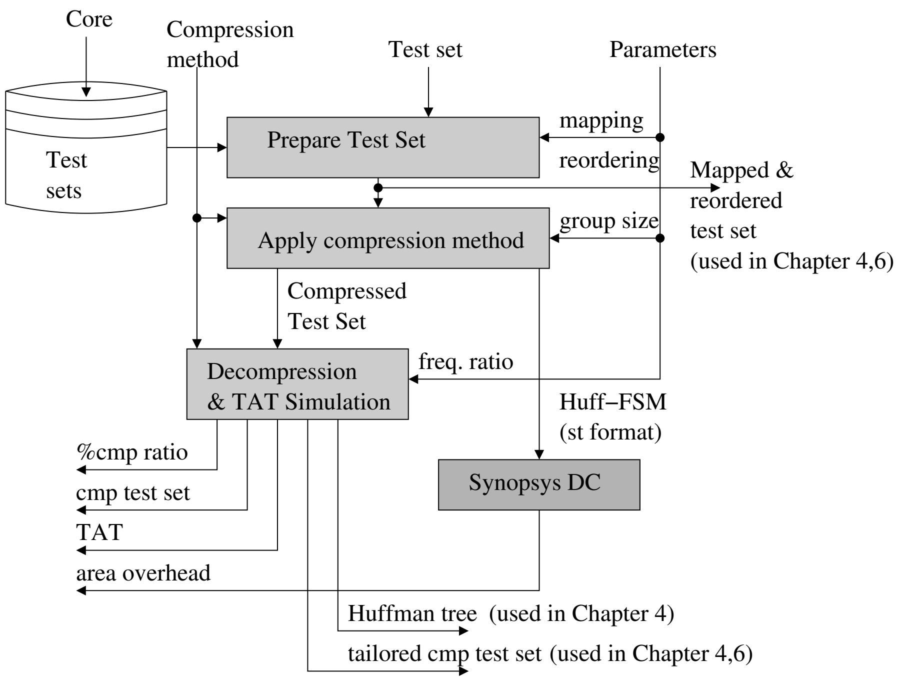


Figure F.1. Tool flow for test data compression

F.2 Test data compression

The results provided in Chapter 3 have been determined using the flow illustrated in Figure F.1. The inputs to the flow are the core, the compression method, and the parameters (whether mapping and reordering should be performed, which is the group size and the frequency ratio). Additionally, a test set can be also fed to the flow. Two in-house tools have been developed to implement the operations of this flow. Firstly, the test set (either determined based on the core's information - as in the case of Chapter 3, or received as input) is prepared for compression. As also noted in Chapter 3 this is performed through mapping and reordering of the test set. Secondly, based on the group size, the mapped and reordered test set is compressed. Further on, the flow follows a decompression and simulation step. This step has been added to determine the TAT for a given frequency ratio. For details about how the TAT is computed the reader is referred to Section 3.3.2 (see Page 52). In order to determine the area overhead imposed by the decoder, the Huffman FSM is generated during compression in the Synopsys state table format. This is, then, fed to the Synopsys design compiler together with a customisable VHDL implementation of the control and generation unit (see Section 3.3 on Page 48). For details regarding the VHDL implementation the reader is referred to Appendix B. Using the Synopsys design compiler the area overhead has been determined. Additionally, the decompres-

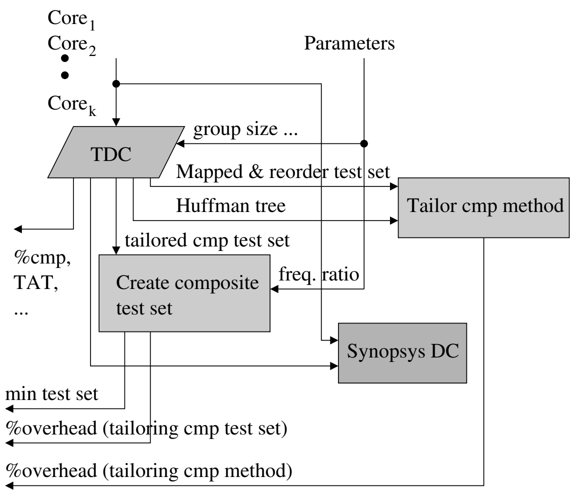


Figure F.2. Tool flow for computing synchronisation overhead

sion and simulation tool is also capable of generating the Huffman tree and the tailored compressed test set. These are used in Chapters 4 and 6.

F.3 Synchronisation overhead

The results provided in Chapter 4 have been determined using the flow illustrated in Figure F.2. The inputs to this flow are the cores fed to the distribution architecture and the parameters used for compression (e.g., the group size, the frequency ratio). Two issues are analysed in Chapter 4: the difference between tailoring the compressed test set and tailoring the compression method, and the distribution architecture. An in-house tool has been developed to perform these experiments. Firstly, the core passes through the test data compression (TDC) flow (see Section F.2) and, then, the tailored compressed test set is used to determine the composite test set for the distribution architecture containing k cores. To also facilitate the comparison between tailoring the compression method and tailoring the compressed test set, the Huffman tree generated by the TDC flow has been tailored separately (tailor cmp. method in the figure), and the overhead with respect to the minimum size test set (min. test set) has been determined (see Section 4.1.2 for further information regarding tailoring of the compression method). The decoders obtained from the TDC flow and the number of cores have been fed to a customisable VHDL implementation of the distribution architecture which has been then synthesised using Synopsys design compiler (DC). This step has been performed manually. For details regarding the VHDL implementation the reader is referred to Appendix C.

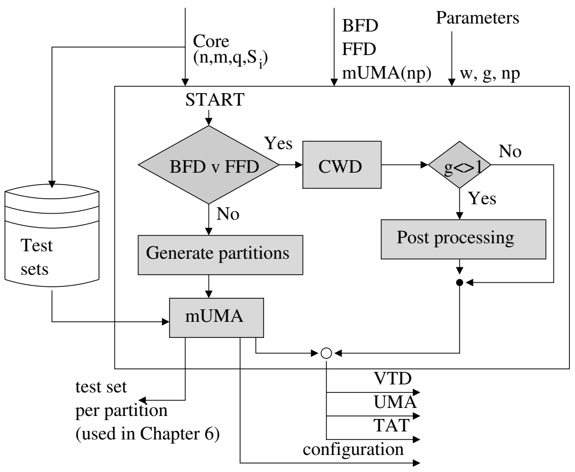


Figure F.3. Tool flow for determining UMA

F.4 Core wrapper design

The tool flow used to determine the results provided in Chapter 5 is illustrated in Figure F.3. The inputs to this flow are the specifications of the core, the type of core wrapper design, and parameters (e.g., the test bus width (w), the granularity (g), the number of partitions (np)). Two in-house tools have been developed for the BFD and FFD heuristics and the **mUMA** heuristic respectively. If the core wrapper design method was either FFD or BFD, these methods have been used, and depending on the group granularity, the post processing step has been applied or not. When the **mUMA** algorithm was used, firstly, the partitions have been generated as outlined in Chapter 5 (see Section 5.3.1 on Page 95), and, then, for the generated partitions the **mUMA** algorithm has been used. Additionally, a test set can be also fed into the algorithm, case in which the partition's test sets can be determined. These will be used in Chapter 6. The configuration generated by this tool flow contains the number of partitions, the length and cardinality of the partitions, and the corresponding amount of UMA. It should be noted that, as detailed in Section 5.3.1, when $g > 1$, the partitions have been generated such that each partition's cardinality is divisible with g.

F.5 Integrated SOC test solution

As illustrated with the design flow given in Section 6.2.2, the proposed integrated test solution requires core wrapper design (CWD) and test access mechanism (TAM) design

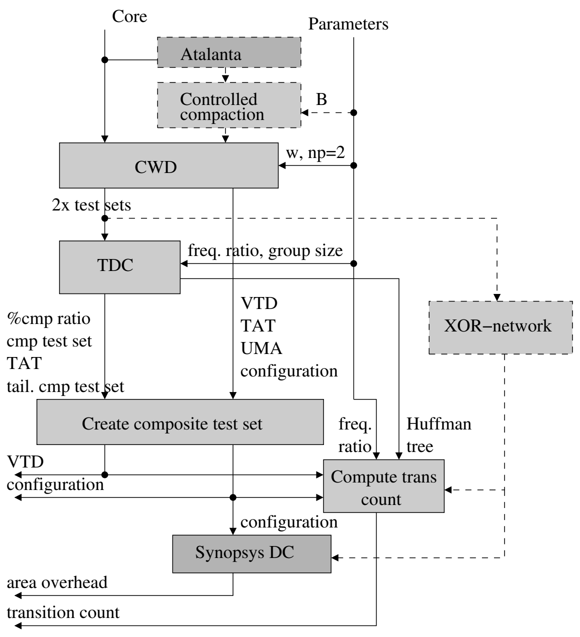


Figure F.4. Tools flow for integrated SOC test solution

to be performed before entering the TDC extension. In Figure F.4, the tool flow which implements the TDC extension to the design flow is illustrated. The inputs to the tool flow are the core and the parameters (e.g., the frequency ratio, the group size, the test bus width (w), there are only two partitions ($np = 2$)). Firstly, the core follows the CWD flow illustrated in Section F.4, then with the test sets associated to the two partitions (see Section 6.2 on Page 124), the test data compression (TDC) flow is entered. Here, the compression ratio (% cmp. ratio in the figure), the tailored compressed test set (tail. cmp. test set in the figure), the test application time (TAT) are determined. Using the tailored compressed test sets for the two partitions, the next tool will generate the composite test set for the extended distribution architecture as detailed in Section 6.2. If the transition count for the proposed architecture is also required, then the compute transition count tool is used, which using the frequency ratio, the Huffman tree, the core wrapper configuration and the composite test set determines the average number of transitions in the scan chains. Finally, when area overhead should be computed the core wrapper configuration and the Huffman state table format file are fed into Synopsys design compiler. For details about the VHDL implementation of the extended distribution architecture, the reader is referred to Appendix E.

As noted in Chapter 6.3, two experiments have been performed. The first one follows the above flow, while the second one required two more extensions (marked with dashed lines in Figure F.2). Firstly, to emulate different care bit densities, the Atalanta [152] ATPG tool has been used. Atalanta has been applied on the ISCAS89 benchmarks as detailed in Section 6.3.2 (see Page 139). After the Atalanta test set has been obtained, controlled static compaction has been applied using the control value *B*. In order to perform the compression using the XOR-network method, a tool has been implemented using the algorithm detailed in Section E.2. Similarly to the extended distribution architecture, the transition count and the area overhead have been determined using an in-house tool and Synopsys design compiler respectively.

Appendix G

Publications

- P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, “Improving Compression Ratio, Area overhead, and Test Application Time in System-on-a-Chip Test Data Compression/ Decompression,” in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 604–611, IEEE Computer Society Press, Mar. 2002.

- P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, “Useless Memory Allocation: Problems and Solutions,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 423–430, IEEE Computer Society Press, Apr. 2002.

- P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, “Reducing Synchronization Overhead in Test Data Compression Environments,” in *Digest of Papers of IEEE European Test Workshop (ETW)*, pp. 147–152, IEEE Computer Society Press, May 2002.

- P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, “Integrated Test Data Decompression and Core Wrapper Design for Low-Cost System-on-a-Chip Testing,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 64–73, IEEE Computer Society Press, Oct. 2002.

- P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, “Test Data Compression: The System Integrator’s Perspective,” in *Proceedings Design, Automation, and Test in Europe (DATE)*, IEEE Computer Society Press, Mar. 2003. (To appear).

- P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, “Variable-length Input Huffman

Coding for System-on-a-Chip Test," *IEEE Transactions on Computer-Aided Design*. Accepted for publication.

- P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Addressing Useless Test Data in Core Based System-on-a-Chip," *IEEE Transactions on Computer-Aided Design*. 2nd revision.

- P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Addressing Synchronization Overhead for High Speed Test Using Low-Cost Testers," *IEEE Design & Test of Computers*. 1st revision.

References

[1] R. K. Gupta and Y. Zorian, “Introducing Core-Based System Design,” *IEEE Design & Test of Computers*, vol. 14, pp. 15–25, Dec. 1997.

[2] Y. Zorian, E. J. Marinissen, and S. Dey, “Testing Embedded-Core-Based System Chips,” *IEEE Computer*, vol. 32, pp. 52–60, June 1999.

[3] E. J. Marinissen and Y. Zorian, “Challenges in Testing Core-Based System ICs,” *IEEE Communications Magazine*, vol. 37, pp. 104–109, June 1999.

[4] Y. Zorian, S. Dey, and M. Rodgers, “Test of Future System-on-Chips,” in *Proceedings International Conference on Computer-Aided Design (ICCAD)*, pp. 392–398, Nov. 2000.

[5] B. T. Murray and J. P. Hayes, “Testing ics: Getting to the core of the problem,” *IEEE Computer*, vol. 29, pp. 32–38, Nov. 1996.

[6] A. M. Rincon, C. Cherichetti, J. A. Monzel, D. R. Stauffer, and M. T. Trick, “Core Design and System-on-a-Chip Integration,” *IEEE Design & Test of Computers*, vol. 14, pp. 26–35, Dec. 1997.

[7] L. Whetsel, “Core Test Connectivity, Communication, & Control,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 303–312, IEEE Computer Society Press, Oct. 1998.

[8] L. Whetsel, “Optaddressable Test Ports: An Approach to Testing Embedded Cores,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 1055–1064, IEEE Computer Society Press, Sept. 1999.

[9] J. Rajski, “DFT for High-Quality Low Cost Manufacturing Test,” in *Proceedings of the Asian Test Symposium (ATS)*, pp. 3–8, IEEE Computer Society Press, Nov. 2001.

[10] H. Vranken, T. Waayers, H. Fleury, and D. Lelouvier, “Enhanced Reduced Pin-Count Test for Full-Scan Design,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 738–747, IEEE Computer Society Press, Oct. 2001.

[11] R. Kapur, R. Chandramouli, and T. Williams, “Strategies for Low-Cost Test,” *IEEE Design & Test of Computers*, vol. 18, pp. 47–54, Dec. 2001.

[12] J. Bedsole, R. Raina, A. Crouch, and M. S. Abadir, “Very Low Cost Testers: Opportunities and Challenges,” *IEEE Design & Test of Computers*, vol. 18, pp. 60–69, Sept. 2001.

[13] ITRS, “The International Technology Roadmap for Semiconductors, 2001 Edition.” http://public.itrs.net/.

[14] B. Bottoms, “The third millennium’s test dilemma,” *IEEE Design & Test of Computers*, vol. 15, pp. 7–11, Oct. 1998.

[15] A. Khoche and J. Rivoir, “I/O Bandwidth Bottleneck for Test: Is it Real ?,” in *Proceedings of Test Resource Partitioning Workshop*, pp. 2.3–1–2.3–6, IEEE Computer Society Press, Nov. 2000.

[16] N. Touba and B. Pouya, “Testing Embedded Cores Using Partial Isolation Rings,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 10–16, IEEE Computer Society Press, Apr. 1997.

[17] B. Pouya and N. Touba, “Modifying User-Defined Logic for Test Access to Embedded Cores,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 60–68, IEEE Computer Society Press, Nov. 1997.

[18] N. Touba and B. Pouya, “Using Partial Isolation Rings to Test Core-Based Designs,” *IEEE Design & Test of Computers*, vol. 14, pp. 52–59, Dec. 1997.

[19] S. Gerez, *Algorithms for VLSI Design Automation*. John Wiley & Sons, 1999.

[20] Z. Navabi, *VHDL: Analysis and Modeling of Digital Systems*. McGraw-Hill, 1997.

[21] D. Ott and T. Wilderotter, *A Designer’s Guide to VHDL Synthesis*. Kluwer Academic Publishers, 1994.

[22] A. Rushton, *VHDL for Logic Synthesis*. John Wiley & Sons, second ed., 1998.

[23] M. Zwoliński, *Digital System Design with VHDL*. Prentice-Hall, 2000.

[24] Z. Navabi, *Verilog Digital System Design (Professional Engineering)*. McGraw-Hill, 1999.

[25] D. Verkest, J. Kunkel, and F. Schrirrmeister, “System level design using C++,” in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 74–81, IEEE Computer Society Press, Mar. 2000.

[26] R. Pasko, S. Vernalde, and P. Schaumont, “Techniques to evolve a C++ based system design language,” in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 302–309, IEEE Computer Society Press, Mar. 2002.

[27] S. Liao, “Towards a new standard for system-level design,” in *Proceedings of the International Workshop on Hardware/Software Codesign*, pp. 2–6, May 2000.

[28] G. Arnout, “SystemC standard,” in *Proceedings of the Asia and South Pacific Design Automation Conference*, pp. 573–577, Jan. 2000.

[29] P. R. Panda, "SystemC a modeling platform supporting multiple design abstractions," in *International Symposium on System Synthesis*, pp. 75–80, Sept. 2001.

[30] S. Devadas, A. Ghosh, and K. Keutzer, *Logic Synthesis*. McGraw-Hill, 1994.

[31] C. Kern and M. Greenstreet, "Formal verification in hardware design: A survey," *ACM Transactions on Design Automation of Electronic Systems*, vol. 4, pp. 123–193, Apr. 1999.

[32] S. Tasiran and K. Keutzer, "Coverage Metrics for Functional Validation of Hardware Designs," *IEEE Design & Test of Computers*, vol. 18, pp. 36–45, July 2001.

[33] F. V. of Commercial Integrated Circuits, "Carl Pixley," *IEEE Design & Test of Computers*, vol. 18, pp. 4–5, July 2001.

[34] H. Foster, "Applied Boolean Equivalence Verification and RTL Static Sign-Off," *IEEE Design & Test of Computers*, vol. 18, pp. 6–15, July 2001.

[35] H. Chang, L. Cooke, M. Hunt, G. Martin, A. McNelly, and L. Todd, *Surviving the SOC revolution. A Guide to Platform-Based design*. Kluwer Academic Publishers, 1999.

[36] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*. IEEE Press, 1990.

[37] R. C. Aitken, "Nanometer Technology Effects on Fault Models for IC Testing," *IEEE Computer*, vol. 32, pp. 46–51, Nov. 1999.

[38] M. L. Bushnell and V. D. Agrawal, *Esentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Kluwer Academic Publishers, 2000.

[39] K. D. Wagner and S. Dey, "High-Level Synthesis for Testability: A Survey and Perspective," in *Proceedings ACM/IEEE Design Automation Conference (DAC)*, pp. 131–136, Association for Computing Machinery, Inc., June 1996.

[40] S. Dey, A. Raghunathan, and K. D. Wagner, "Design for testability techniques at the behavioral and register-transfer levels," *Journal of Electronic Testing: Theory and Applications*, vol. 13, no. 2, pp. 79–91, 1998.

[41] I. Ghosh, N. K. Jha, and S. Bhawmik, "A BIST scheme for rtl circuits based on symbolic testability analysis," *IEEE Transactions on Computer-Aided Design*, vol. 19, pp. 111–128, Jan. 2000.

[42] M. Schulz, E. Trishler, and T. Sarfert, "Socrates: A highly efficient automatic test pattern generation system," *IEEE Transactions on Computer-Aided Design*, pp. 126–137, Jan. 1988.

[43] T. M. Niermann and J. H. Patel, "HITEC: A test generation package for sequential circuits," in *Proceedings European Conference on Design Automation (EDAC)*, pp. 214–218, Feb. 1991.

[44] K.-T. Cheng and A. Krstic, "Current Directions in Automatic Test-Pattern Generation," *IEEE Computer*, vol. 32, pp. 58–64, Nov. 1999.

[45] S. Ma, P. Franco, and E. McCluskey, "An experimen-tal chip to evaluate test techniques experimental results," in *Proceedings IEEE International Test Conference (ITC)*, pp. 663–672, Apr. 1995.

[46] E. J. McCluskey and C.-W. Tseng, "Stuck-Fault Tests vs. Actual Defects," in *Proceedings IEEE International Test Conference (ITC)*, pp. 336–343, IEEE Computer Society Press, Oct. 2000.

[47] A. L. Crouch, *Design-for-test for Digital IC's and Embedded Core Systems*. Prentice-Hall, 1999.

[48] Y. Zorian and E. J. Marinissen, "System Chip Test: How Will It Impact Your Design?," in *Proceedings ACM/IEEE Design Automation Conference (DAC)*, pp. 136–142, Association for Computing Machinery, Inc., June 2000.

[49] Y. Zorian, "Test Requirements for Embedded Core-Based Systems and IEEE P1500," in *Proceedings IEEE International Test Conference (ITC)*, pp. 191–199, IEEE Computer Society Press, Nov. 1997.

[50] R. Kapur *et al.*, "P1500-CTL: Towards a Standard Core Test Language," in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 489–490, IEEE Computer Society Press, Apr. 1999.

[51] E. J. Marinissen, R. Kapur, and Y. Zorian, "On Using IEEE P1500 SECT for Test Plug-n-Play," in *Proceedings IEEE International Test Conference (ITC)*, pp. 770–777, IEEE Computer Society Press, Oct. 2000.

[52] "IEEE P1500 Web Site." http://grouper.ieee.org/groups/1500/.

[53] E. J. Marinissen, Y. Zorian, R. Kapur, T. Taylor, and L. Whetsel, "Towards a Standard for Embedded Core Test: An Example," in *Proceedings IEEE International Test Conference (ITC)*, pp. 616–627, IEEE Computer Society Press, Sept. 1999.

[54] R. Kapur, M. Lousberg, T. Taylor, B. Keller, P. Reuter, and D. Kay, "CTL, The Language for Describing Core-Based Test," in *Proceedings IEEE International Test Conference (ITC)*, IEEE Computer Society Press, Oct. 2001.

[55] E. J. Marinissen, R. Kapur, M. Lousberg, T. McLaurin, M. Ricchetti, and Y. Zorian, "On IEEE P1500's Standard for Embedded Core Test," *Journal of Electronic Testing: Theory and Applications*, vol. 18, pp. 365–383, Aug. 2002.

[56] E. J. Marinissen, S. K. Goel, and M. Lousberg, "Wrapper Design for Embedded Core Test," in *Proceedings IEEE International Test Conference (ITC)*, IEEE Computer Society Press, Oct. 2000.

[57] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, “Co-Optimization of Test Wrapper and Test Access Architecture for Embedded Cores,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 1023–1032, IEEE Computer Society Press, Oct. 2001.

[58] M. Garey and D. Johnson, *Computers and Intractability*. W.H.Freeman and Company, 1979.

[59] J. Hromkovic, *Algorithmics for Hard Problems: Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics*. Springer-Verlag, 2002.

[60] K. Chakrabarty, “Design of System-on-a-Chip Test Access Architectures Under Place-and-Route and Power Constraints,” in *Proceedings ACM/IEEE Design Automation Conference (DAC)*, pp. 432–437, Association for Computing Machinery, Inc., June 2000.

[61] K. Chakrabarty, “Design of System-on-a-Chip Test Access Architectures Using Integer Linear Programming,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 127–134, Apr. 2000.

[62] P. K. Nag, A. Gattiker, S. Wei, R. Blanton, and W. Maly, “Modeling the Economics of Testing: A DFT Perspective,” *IEEE Design & Test of Computers*, vol. 19, pp. 29–41, Jan. 2002.

[63] T. Warwick, “What a Device Interface Board Really Costs: An Evalyation of Technical Considerations for Testing Products Operating in the Gigabit Region,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 555–564, IEEE Computer Society Press, Oct. 2002.

[64] D. McFeely, “The Process and Chanllenges of a High-Speed DUT Board Project,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 565–573, IEEE Computer Society Press, Oct. 2002.

[65] D. Heidel, S. Dhong, P. Hofstee, M. Immediato, K. Nowka, J. Silberman, and K. Stawiasz, “High-speed serialiazing/deserializing design-for-test methods for evaluating a 1 ghz microprocessor,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 234–238, IEEE Computer Society Press, Apr. 1998.

[66] J. Blyler, “DFT Drives Quality, Cost, And Time to Market,” pp. 91–93, Wireless System Design, Jan. 2001.

[67] R. Aitken and F. Muradali, “Trends in SLI deign and their effect on test,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 628–637, IEEE Computer Society Press, Sept. 1999.

[68] K. Chakrabarty, V. Iyengar, and A. Chandra, *Test Resource Partitioning for System-on-a-Chip*. Kluwer Academic Publishers, 2002.

[69] P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Improving Compression Ratio, Area overhead, and Test Application Time in System-on-a-Chip Test Data Compression/Decompression," in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 604–611, IEEE Computer Society Press, Mar. 2002.

[70] P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Reducing Synchronization Overhead in Test Data Compression Environments," in *Digest of Papers of IEEE European Test Workshop (ETW)*, pp. 147–152, IEEE Computer Society Press, May 2002.

[71] P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Useless Memory Allocation: Problems and Solutions," in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 423–430, IEEE Computer Society Press, Apr. 2002.

[72] P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Integrated Test Data Decompression and Core Wrapper Design for Low-Cost System-on-a-Chip Testing," in *Proceedings IEEE International Test Conference (ITC)*, pp. 64–73, IEEE Computer Society Press, Oct. 2002.

[73] P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Test Data Compression: The System Integrator's Perspective," in *Proceedings Design, Automation, and Test in Europe (DATE)*, IEEE Computer Society Press, Mar. 2003. (To appear).

[74] P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Variable-length Input Huffman Coding for System-on-a-Chip Test," *IEEE Transactions on Computer-Aided Design*. Accepted for publication.

[75] P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Addressing Useless Test Data in Core Based System-on-a-Chip," *IEEE Transactions on Computer-Aided Design*. 2nd revision.

[76] P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Addressing Synchronization Overhead for High Speed Test Using Low-Cost Testers," *IEEE Design & Test of Computers*. 1st revision.

[77] V. Immaneni and S. Raman, "Direct Access Test Scheme - Design of Block and Core Cells for Embedded ASICs," in *Proceedings IEEE International Test Conference (ITC)*, pp. 488–492, IEEE Computer Society Press, Sept. 1990.

[78] P. Varma and S. Bhatia, "A Structured Test Re-Use Methodology for Core-Based System Chips," in *Proceedings IEEE International Test Conference (ITC)*, pp. 294–302, IEEE Computer Society Press, Oct. 1998.

[79] I. Ghosh, S. Dey, and N. K. Jha, "A Fast and Low Cost Testing Technique for Core-based System-on-Chip," in *Proceedings ACM/IEEE Design Automation Conference (DAC)*, pp. 542–547, Association for Computing Machinery, Inc., June 1998.

[80] M. Nourani and C. Papachristou, "Structural Fault Testing of Embedded Cores Using Pipelining," *Journal of Electronic Testing: Theory and Applications*, vol. 15, no. 1, p. 129, 1999.

[81] L. Whetsel, "An IEEE 1149.1 Based Test Access Architecture for ICs with Embedded Cores," in *Proceedings IEEE International Test Conference (ITC),* pp. 69–78, IEEE Computer Society Press, Nov. 1997.

[82] M. Sugihara, H. Date, and H. Yasuura, "A Novel Test Methodology for Core-Based System LSIs and a Testing Time Minimization Problem," in *Proceedings IEEE International Test Conference (ITC),* pp. 465–472, IEEE Computer Society Press, Oct. 1998.

[83] K. Chakrabarty, "Test scheduling for core-based systems using mixed-integer linear programming," *IEEE Transactions on Computer-Aided Design,* vol. 19, pp. 1163–1174, Oct. 2000.

[84] E. J. Marinissen, R. Arendsen, G. Bos, H. Dingemanse, M. Lousberg, and C. Wouters, "A Structured And Scalable Mechanism for Test Access to Embedded Reusable Cores," in *Proceedings IEEE International Test Conference (ITC),* pp. 284–293, IEEE Computer Society Press, Oct. 1998.

[85] J. Aerts and E. J. Marinissen, "Scan Chain Design for Test Time Reduction in Core-Based ICs," in *Proceedings IEEE International Test Conference (ITC),* pp. 448–457, IEEE Computer Society Press, Oct. 1998.

[86] S. K. Goel and E. J. Marinissen, "Cluster-Based Test Architecture Design for System-on-Chip," in *Proceedings IEEE VLSI Test Symposium (VTS),* pp. 259–260, IEEE Computer Society Press, Apr. 2002.

[87] S. K. Goel and E. J. Marinissen, "Effective and Efficient Test Architecture Design for SOCs," in *Proceedings IEEE International Test Conference (ITC),* pp. 529–538, IEEE Computer Society Press, Oct. 2002.

[88] M. Nourani and C. Papachristou, "An ILP Formulation to Optimize Test Access Mechanism in System-on-Chip Testing," in *Proceedings IEEE International Test Conference (ITC),* pp. 902–910, IEEE Computer Society Press, Oct. 2000.

[89] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "On Using Rectangle Packing for SOC Wrapper/TAM Co-Optimization," in *Proceedings IEEE VLSI Test Symposium (VTS),* pp. 253–258, IEEE Computer Society Press, Apr. 2002.

[90] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Integrated wrapper/TAM co-optimization, constraint-driven test scheduling, and tester data volume reduction for SOCs," in *Proceedings ACM/IEEE Design Automation Conference (DAC),* pp. 686–690, IEEE Computer Society Press, June 2002.

[91] V. Iyengar, S. K. Goel, E. J. Marinissen, and K. Chakrabarty, "Test Resource Optimization for Multi-Site Testing of SOCs Under ATE Memory Depth Constraints," in *Proceedings IEEE International Test Conference (ITC),* pp. 1159–1169, IEEE Computer Society Press, Oct. 2002.

[92] Y. Huang, S. Reddy, W.-T. Cheng, P. Reuter, N. Mukherjee, C.-C. Tsai, O. Samman, and Y. Zaidan, “Optimal Core Wrapper Width Selection and SOC Test Scheduling Based on 3-D Bin Packing Algorithm,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 74–82, IEEE Computer Society Press, Oct. 2002.

[93] S. Koranne and V. Iyengar, “On the Use of *k* tuples for SoC Test Schedule Representation,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 539–548, IEEE Computer Society Press, Oct. 2002.

[94] I. Hamzaoglu and J. H. Patel, “Test set compaction algorithms for combinational circuits,” in *Proceedings International Conference on Computer-Aided Design (ICCAD)*, pp. 283–289, Nov. 1998.

[95] I. Pomeranz, L. Reddy, and S. Reddy, “COMPACTEST: A method to generate compact test set for combinational circuits,” *IEEE Transactions on Computer-Aided Design*, vol. 12, pp. 1040–1049, July 1993.

[96] T. Yamaguchi, M. Tilgner, M. Ishida, and D. S. Ha, “An Efficient Method for Compressing Test Data to Reduce the Test Data Download Time,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 79–88, IEEE Computer Society Press, 1997.

[97] M. Ishida, D. S. Ha, and T. Yamaguchi, “COMPACT: A Hybrid Method for Compressing Tets Data,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 62–69, IEEE Computer Society Press, Apr. 1998.

[98] G. Hetherington, T. Fryars, N. Tamarapalli, M. Kassab, A. Hassan, and J. Rajski, “Logic BIST for Large Industrial Designs: Real Issues and Case Studies,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 358–367, IEEE Computer Society Press, Sept. 1999.

[99] B. Koenemann, “LFSR-Coded Test Patterns for Scan Designs,” in *Proceedings IEEE European Test Conference (ETC)*, pp. 237–242, IEEE Computer Society Press, Mar. 1991.

[100] H.-J. Wunderlich and G. Kiefer, “Bit-Flipping BIST,” in *Proceedings International Conference on Computer-Aided Design (ICCAD)*, Nov. 1996.

[101] N. A. Touba and E. J. McCluskey, “Altering a Pseudorandom Bit Sequence for Scan-Based BIST,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 167–175, IEEE Computer Society Press, Oct. 1996.

[102] G. Kiefer and H.-J. Wunderlich, “Deterministic BIST with Multiple Scan Chains,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 1057–1064, IEEE Computer Society Press, Oct. 1998.

[103] J. Rajski, J. Tyszer, and N. Zacharia, “Test data decompression for multiple scan designs with boundary scan,” *IEEE Transactions on Computers*, vol. 47, pp. 1188–1200, Nov. 1998.

[104] G. Kiefer, H. Vranken, E. J. Marinissen, and H.-J. Wunderlich, “Application of Deterministic Logic BIST on Industrial Circuits,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 105–114, IEEE Computer Society Press, Oct. 2000.

[105] L. Jin-Fu and W. Cheng-Wen, “Memory fault diagnosis by syndrome compression,” in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 97–101, IEEE Computer Society Press, Mar. 2001.

[106] J. T. Chen, , J. Rajski, J. Khare, O. Kebichi, and W. Maly, “Enabling embedded memory diagnosis via test response compression,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 292–298, IEEE Computer Society Press, May 2001.

[107] J. Rajski, J. Tyszer, M. Kassab, N. Mukherjee, R. Thompson, K.-H. Tsai, A. Hertwig, N. Tamarapalli, G. Mrugalski, G. Eide, and J. Qian, “Embedded Deterministic Test for Low Cost Manufacturing Test,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 301–310, IEEE Computer Society Press, Oct. 2002.

[108] S. Mitra and K. S. Kim, “X-Compact an Efficient Response Compaction Technique for Test Cost Reduction,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 311–320, IEEE Computer Society Press, Oct. 2002.

[109] D. Das and N. A. Touba, “Reducing Test Data Volume Using External/LBIST Hybrid Test Patterns,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 115–122, IEEE Computer Society Press, Oct. 2000.

[110] C. Krishna, A. Jas, and N. A. Touba, “Test Vector Encoding Using Partial LFSR Reseeding,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 885–893, IEEE Computer Society Press, Oct. 2001.

[111] A. Jas, C. Krishna, and N. A. Touba, “Hybrid BIST Based on Weighted Pseudo-Random Testing: A New Test Resource Partitioning Scheme,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 2–8, IEEE Computer Society Press, May 2001.

[112] D. Kay and S. Mourad, “Compression technique for interactive BIST application,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 9–14, IEEE Computer Society Press, May 2001.

[113] D. Kay, S. Chung, and S. Mourad, “Embedded Test Control Schemes for Compression in SOCs,” in *Proceedings ACM/IEEE Design Automation Conference (DAC)*, pp. 679–684, IEEE Computer Society Press, June 2002.

[114] B. Koenemann, C. Barnhart, B. Keller, T. Snethen, O. Farnsworth, and D. Wheater, “A SmartBIST Variat with Guaranteed Encoding,” in *Proceedings of the Asian Test Symposium (ATS)*, pp. 325–330, IEEE Computer Society Press, Nov. 2001.

[115] R. Dorsch and H.-J. Wunderlich, “Tailoring ATPG for Embedded Testing,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 530–537, IEEE Computer Society Press, Oct. 2001.

[116] I. Bayraktaroglu and A. Orailoglu, “Test Volume and Application Time Reduction Through Scan Chain Concealment,” in *Proceedings ACM/IEEE Design Automation Conference (DAC)*, vol. 38, pp. 151–155, June 2001.

[117] C. Krishna and N. A. Touba, “Reducing Test Data Volume Using LFSR Reseeding with Seed Compression,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 311–320, IEEE Computer Society Press, Oct. 2002.

[118] V. Iyengar, K. Chakrabarty, and B. Murray, “Deterministic built-in pattern generation for sequential circuits,” *Journal of Electronic Testing: Theory and Applications*, vol. 15, pp. 97–114, August/October 1999.

[119] A. Jas and N. Touba, “Test Vector Decompression Via Cyclical Scan Chains and Its Application to Testing Core-Based Designs,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 458–464, IEEE Computer Society Press, Oct. 1998.

[120] A. Jas and N. Touba, “Using an Embedded Processor for Efficient Deterministic Testing of Systems-on-a-Chip,” in *Proceedings International Conference on Computer Design (ICCD)*, pp. 418–423, Oct. 1999.

[121] A. Jas, J. Ghosh-Dastidar, and N. A. Touba, “Scan Vector Compression/Decompression Using Statistical Coding,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 114–121, IEEE Computer Society Press, Apr. 1999.

[122] A. Chandra and K. Chakrabarty, “System-on-a-Chip Test Data Compression and Decompression Architectures Based on Golomb Codes,” *IEEE Transactions on Computer-Aided Design*, vol. 20, pp. 113–120, Mar. 2001.

[123] A. Chandra and K. Chakrabarty, “Frequency-Directed Run-Length (FDR) Codes with Application to System-on-a-Chip Test Data Compression,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 114–121, IEEE Computer Society Press, Apr. 2001.

[124] A. El-Maleh, S. al Zahir, and E. Khan, “A Geometric-Primitives-Based Compression Scheme for Testing Systems-on-Chip,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 114–121, IEEE Computer Society Press, Apr. 2001.

[125] S. Reda and A. Orailoglu, “Reducing Test Application Time Through Test Data Mutation Encoding,” in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 387–393, IEEE Computer Society Press, Mar. 2002.

[126] F. G. Wolff and C. Papachristou, “Multiscan-based Test Compression and Hardware Decompression Using LZ77,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 321–330, IEEE Computer Society Press, Oct. 2002.

[127] T. Cover and J. Thomas, *Elements of Information Theory*. John Wiley & Sons, 1991.

[128] V. D. Agrawal and T. J. Chakraborty, “High-Performance Circuit Testing with Slow-Speed Testers,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 302–310, IEEE Computer Society Press, Apr. 1995.

[129] H. Ichihara, A. Ogava, T. Inoue, and A. Tamura, “Dynamic Test Compression Using Statistical Coding,” in *Proceedings of the Asian Test Symposium (ATS)*, pp. 143–148, IEEE Computer Society Press, Nov. 2001.

[130] H. Ichihara, K. Kinoshita, I. Pomeranz, and S. Reddy, “Test Transformation to Improve Compaction by Statistical Encoding,” in *Proceedings International Conference on VLSI Design*, pp. 294–299, IEEE Computer Society Press, Jan. 2000.

[131] The University of Illinois. www.crhc.uiuc.edu/IGATE.

[132] F. Brglez, D. Bryan, and K. Kozminski, “Combinational profiles of sequential benchmark circuits,” in *Proceedings International Symposium on Circuits and Systems (ISCAS)*, pp. 1929–1934, May 1989.

[133] Synopsys Inc., “Design Compiler Reference Manual.” http://www.synopsys.com/, 2002.

[134] A. Chandra and K. Chakrabarty, “Test Resource Partitioning for SOCs,” *IEEE Design & Test of Computers*, vol. 18, pp. 80–91, Sept. 2001.

[135] E. J. Marinissen, V. Iyengar, and K. Chakrabarty, “ITC’02 SOC Test Benchmarks Web Site.” http://www.extra.research.philips.com/itc02socbench/.

[136] B. West and T. Napier, “Sequencer Per Pin ^{TM} Test System Architecture,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 355–361, IEEE Computer Society Press, Sept. 1990.

[137] F. F. Hsu, K. M. Butler, and J. H. Patel, “A Case Study on the Implementation of the Illinois Scan Architecture,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 538–547, IEEE Computer Society Press, Oct. 2001.

[138] C. Barnhart, V. Brunkhorst, F. Distler, O. Farnsworth, B. Keller, B. Keller, and B. Koenemann, “OPMISR: The Fundation for Compressed ATPG Vectors,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 738–747, IEEE Computer Society Press, Oct. 2001.

[139] I. Goulden and D. Jackson, *Combinational Enumeration*. John Wiley & Sons, 1983.

[140] A. Sivaram, “Split Timing Mode (STM) - Answer to Dual Frequency Domain Testing,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 738–747, IEEE Computer Society Press, Oct. 2001.

[141] Y. Zorian, “A distributed BIST control scheme for complex VLSI devices,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 4–9, IEEE Computer Society Press, 1993.

[142] S. Wang and S. Gupta, “ATPG for heat dissipation minimization during test application,” *IEEE Computer*, vol. 47, pp. 256–262, Feb. 1998.

[143] L. Whetsel, "Adapting Scan Architectures for Low Power Operation," in *Proceedings IEEE International Test Conference (ITC)*, pp. 863–872, IEEE Computer Society Press, Oct. 2000.

[144] N. Nicolici, *Power Minimisation Techniques for Testing Low Power VLSI Circuits*. PhD thesis, University Southampton, Southampton, UK, Oct. 2000.

[145] A. Chandra and K. Chakrabarty, "Combining Low-Power Scan Testing and Test Data Compression for System-on-a-chip," in *Proceedings ACM/IEEE Design Automation Conference (DAC)*, vol. 38, pp. 113–120, June 2001.

[146] P. M. Rosinger, P. T. Gonciari, B. A. Hashimi, and N. Nicolici, "Simultaneos reduction in volume of test data and power dissipation for system-on-a-chip," *IEE Electronics Letters*, vol. 37, pp. 1434–1436, Nov. 2001.

[147] R. Sankaralingam, R. R. Oruganti, , and N. A. Touba, "Static Compaction Techniques to Control Scan Vector Power Dissipation," in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 35–40, Apr. 2000.

[148] P. M. Rosinger, P. T. Gonciari, B. Al-Hashimi, and N. Nicoli, "Analysing trade-offs in scan power and test data compression for system-on-a-chip," *IEE Proceedings, Computers and Digital Techniques*, vol. 149, pp. 188–196, July 2002.

[149] N. Nicolici and B. M. Al-Hashimi, "Multiple Scan Chains for Power Minimization during Test Application in Sequential Circuits," *IEEE Computer*, vol. 51, pp. 721–734, June 2002.

[150] P. Rosinger, B. Al-Hashimi, and N. Nicolici, "Scan architecture for shift and capture power reduction," in *Proceedings of the IEEE Symposium on Defect and Fault Tolerance (DFT)*, pp. 129–137, Nov. 2002.

[151] A. Chandra and K. Chakrabarty, "Test Resource Partitioning and Reduced Pin-Count Testing Based on Test Data Compression," in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 598–603, IEEE Computer Society Press, Mar. 2002. (To appear).

[152] "Virginia Polytechnic Institute and State University." http://www.ee.vt.edu/˜ha/cadtools/cadtools.html.

[153] S. Morris, "The DFT-Focused Tester." http://www.teseda.com/pdfs/DFT-FocusedTesters.pdf, Aug. 2002.

[154] B. G. West, "At-Speed Structural Test," in *Proceedings IEEE International Test Conference (ITC)*, pp. 795–800, IEEE Computer Society Press, Sept. 1999.

[155] G. Kiefer and H.-J. Wunderlich, "Using BIST Control for Pattern Generation," in *Proceedings IEEE International Test Conference (ITC)*, pp. 347–355, IEEE Computer Society Press, Nov. 1997.

[156] S. Hellebrand, H.-G. Liang, and H.-J. Wunderlich, “A Mixed Mode BIST Scheme Based on Reseeding of Folding Counters,” in *Proceedings IEEE International Test Conference (ITC)*, pp. 778–784, IEEE Computer Society Press, Oct. 2000.

[157] M. Chatterjee and D. K. Pradhan, “A novel pattern generator for near-perfect fault-coverage,” in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 417–425, IEEE Computer Society Press, 1995.

[158] Model Technology Inc., “ModelSim, HDL Simulator.” http://www.model.com/, 2002.

[159] M. Pedram, “Power minimization in IC design: principles and applications,” *ACM Transactions on Design Automation of Electronic Systems*, vol. 1, pp. 3 – 56, Jan. 1996.