NAIST-IS-DD0861209

Doctoral Dissertation

Studies on F-Scan: A Design for Testability Method for Functional RTL Circuits

Marie Engelene J. Obien

February 3, 2011

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Marie Engelene J. Obien

Thesis Committee:

Professor Hideo Fujiwara	(Supervisor)
Professor Yasuhiko Nakashima	(Co-supervisor)
Associate Professor Michiko Inoue	(Co-supervisor)
Assistant Professor Satoshi Ohtake	(Co-supervisor)

Studies on F-Scan: A Design for Testability Method for Functional RTL Circuits*

Marie Engelene J. Obien

Abstract

As the complexity and the number of transistors in digital chips increase, ensuring quality becomes more difficult. In order to make circuits easily testable, design for testability (DFT) is the most popular approach. Full scan design is a mainstream DFT method that effectively addresses the complexity of test pattern generation. The trade-offs of gate-level full scan, however, impact test costs in terms of area overhead and test application time. Moreover, scan-based DFT may change the circuit states during test mode that can be possibly different from that in functional mode. This means that automatic test pattern generation (ATPG) tools may generate patterns that are illegal during functional mode, hence result in over-testing and yield loss.

In this thesis, we propose F-scan, a new DFT technique applicable to functional register-transfer-level circuits. F- scan organizes every register in the circuit in an F-scan-path by maximizing the use of available functional logic and paths for testing purposes. Hence, it effectively reduces the hardware overhead due to test without compromising fault coverage. The creation of F-scan-paths also ensure short test application time. To complete the DFT method, we propose the following ATPG techniques for F-scan: (a) constrained ATPG for efficient test pattern generation; (b) hybrid model for delay fault testing; and (c) F-scan test

*Doctoral Dissertation, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD0861209, February 3, 2011.

generation model using standard full scan delay fault ATPG for test generation time improvement.

Constrained ATPG generates only the applicable test vectors to the F-scan-paths. This is because the constraints are based on the F-scan-path information. Since the F-scan-paths utilize available functional paths in the circuit, there is a possibility of generating test patterns that conform closely to the functionally reachable state of the circuit. Thus, over-testing can be reduced. This constrained ATPG technique is applicable for stuck-at and delay fault models. The constrained ATPG for stuck-at faults is straightforward but for delay faults, a model for two-pattern test is needed.

In full scan, the two conventional approaches to delay fault testing are skewed-load and broadside. Each of these methods has different disadvantages. For skewed-load, fast scan-enable timing requirement is costly and takes longer design time. On the other hand, broadside has lower fault coverage compared to skewed-load. Thus, we have extended our constrained ATPG to a hybrid model that allows both skewed-load and broadside approaches for delay fault testing. This is done by copying the combinational part of the F-scannable circuit to two time frames during testing and connecting these to the constraint modules. This hybrid model produces high fault coverage without the problem of scan-enable timing, which is best for circuits that require high quality. If the fault coverage is not a priority and high yield is expected, the hybrid model can be set to broadside mode only in order to reduce over-testing instead. Moreover, in order to improve test generation time for F-scan delay fault testing, we propose a new test generation model for F-scan that uses the standard full scan delay fault ATPG. This allows easy integration of F-scan to currently available ATPG tools.

For all the methods related to F-scan proposed, we conducted experiments thoroughly on benchmark circuits and evaluated the results to prove the effectiveness of these approaches against the performance of conventional gate-level full scan design.

Keywords:

scan-based DFT, functional RTL circuits, automatic test pattern generation, high-level testing, delay fault testing, assignment decision diagrams

List of Publications

Journal Papers

1. Marie Engelene J. Obien, Satoshi Ohtake, and Hideo Fujiwara, "F-Scan: A DFT method for functional scan at RTL," IEICE Trans. on Information and Systems, Vol. E94-D, No. 1, pp. 104-113, Jan. 2011.
2. Katsuya Fujiwara, Hideo Fujiwara, Marie Engelene J. Obien and Hideo Tamamoto, "Enumeration and synthesis of shift register equivalents for secure scan design," IEICE Trans. on Information and Systems, Vol. J93-D, No. 11, pp. 2426-2436, Nov. 2010. (In Japanese)

International Conferences (Reviewed)

1. Marie Engelene J. Obien and Hideo Fujiwara, "F-scan: an approach to functional RTL scan for assignment decision diagrams," IEEE 8th International Conference on ASIC (ASICON2009), pp.589-592, Changsha, China, Oct. 2009.
2. Marie Engelene J. Obien and Hideo Fujiwara, "A DFT method for functional scan at RTL," 10th IEEE Workshop on RTL and High Level Testing (WRTLT'09), pp.6-15, Hong Kong, Nov. 2009.
3. Hideo Fujiwara and Marie Engelene J. Obien, "Secure and testable scan design using extended de bruijn graphs," 15th Asia and South Pacific Design Automation Conference (ASP-DAC 2010), pp.413-418, Taipei, Taiwan, Jan. 2010.
4. Katsuya Fujiwara, Hideo Fujiwara, Marie Engelene J. Obien and Hideo Tamamoto, "SREEP: shift register equivalents enumeration and synthesis program for secure scan design,"13th IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS 2010), pp.193-196, Vienna, Austria, Apr. 2010.
5. Marie Engelene J. Obien, Satoshi Ohtake, and Hideo Fujiwara, "Delay fault ATPG for F-scannable RTL circuits," 2010 International Symposium on Communications and Information Technologies (ISCIT 2010), pp.717-722, Tokyo, Japan, Oct. 2010.

6. Marie Engelene J. Obien, Satoshi Ohtake and Hideo Fujiwara,"Constrained ATPG for functional RTL circuits using F-scan," 2010 IEEE International Test Conference (ITC 2010), Paper 21.1, pp.1-10, Austin, Texas, Nov. 2010.
7. Marie Engelene J. Obien, Satoshi Ohtake and Hideo Fujiwara,"F-scan test generation model for delay fault testing at RTL using standard full scan ATPG," 16th IEEE European Test Symposium (ETS 2011), Trondheim, Norway, May 2011 (To Appear).

Technical Reports

1. Hideo Fujiwara and Marie Engelene J. Obien, "A secure scan design approach using extended de bruijn graph," IEICE Technical Report, Vol. 108, No. 431, pp.61-66, Feb. 2009.

Awards

1. IEEE ASICON 2009 Excellent Student Paper Award, Oct. 2009.
2. 2010 IEEE Kansai Section Student Paper Award, Feb. 2011.

To Nico

"Here's to the crazy ones.
The misfits. The rebels. The troublemakers.
The round pegs in the square holes.
The ones who see things differently.
They're not fond of rules.
And they have no respect for the status quo.
You can quote them, disagree with them, glorify or vilify them.
About the only thing you can't do is ignore them.
Because they change things.
They push the human race forward.
And while some may see them as the crazy ones, we see genius.
Because the people who are crazy enough to think they can change the world,
are the ones who do."
- Think Different by Apple, 1997

Contents

List of Figures

List of Tables

Chapter 1

Introduction

THE MAIN CONCERN of this thesis is to deal with the problems of hardware testing while focusing on the early stage of the design process. The problems of testing at lower abstraction levels have long been discussed and researched by many before. However, with the Moore's Law still continuing the trend in very large scale integrated (VLSI) circuits, escalating size and complexity in digital chips have pushed the development of new techniques for higher levels of abstraction. Circuits are now being developed at high abstraction levels, but the tools for testing and design for testability (DFT) have remained lagging behind. Hence, testing complex hardware today is still a problem, moreso in terms of quality and cost.

Testing digital circuits is done to detect faults introduced during or after production. This is different from *hardware verification*, which aims to detect design errors. The aim of the work described in this thesis is to provide a new methodology on making digital circuits *easily testable* by DFT at functional register-transfer level (RTL). Utilizing the available functional elements and paths in the circuit for testing purposes, the DFT method proposed considers reduction of hardware overhead and test application time.

It is also important to improve the test pattern generation for the new DFT method proposed in order to make it useful and integrable to current technologies. In this thesis, a technique for automatic test pattern generation (ATPG) that ensures high fault coverage is defined. An extension of this technique is done to make it applicable for delay fault testing. Further enhancements are also proposed for faster test generation.

To better illustrate how the proposed method can be beneficial in the development of VLSI circuits, it is essential to understand the design flow and the test flow practices, which are outlined in Sections 1.1 and 1.2, respectively. To quantify the effectiveness of different test methodologies, Sections 1.3 and 1.4 discuss the concepts of test cost and quality. The contributions of this thesis are presented in Section 1.5 and the thesis overview is given in Section 1.6.

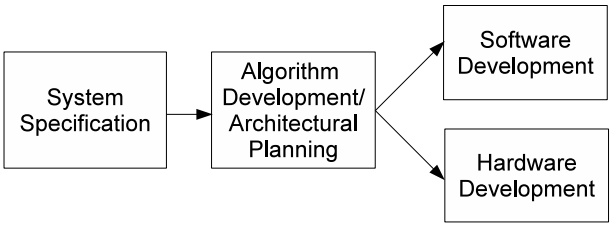


Figure 1.1. System specification refinement and hardware/software partitioning.

1.1 Digital Circuit Models and Design Flow

The development of VLSI circuits and systems begin with specification where the functionality of the circuit is described, as shown in Figure 1.1. The specification is then translated to algorithmic descriptions, which are used for architectural planning. The design flow at this stage is then divided into two paths: *software development* and *hardware development*. This thesis focuses on the hardware development process only.

In order to reduce size and complexity of hardware design specification, *circuit models* are developed. A circuit model has relevant features at different levels of abstraction, wherein the details vary across the levels. Circuit models can be classified in terms of representation or view and levels of abstraction. Figure 1.2 shows the three-axes Y-chart proposed by Gajski [2], which represents the three different model views: *behavioral*, *structural*, and *physical*. The different views of a circuit model describe different types of information of each component in a design. For example, when modelling a circuit that computes the maximum and minimum, the behavioral view specifies the mathematical relationship between the input and the expected outputs of the design. On the other hand, the structural view shows the components (e.g., arithmetic logic units or ALUs) and their interconnects needed to implement the circuit. Lastly, the physical view gives the details of the physical information (e.g., location, dimension) of every component that is used in the structural view. The circuit model views vary in different levels of abstraction. The four main levels of abstraction are: *architectural*, *register-transfer*, *gate*, and *transistor*.

- *Architectural level*: This level contains the main components that are used for the design. These components usually include processors, buses, and

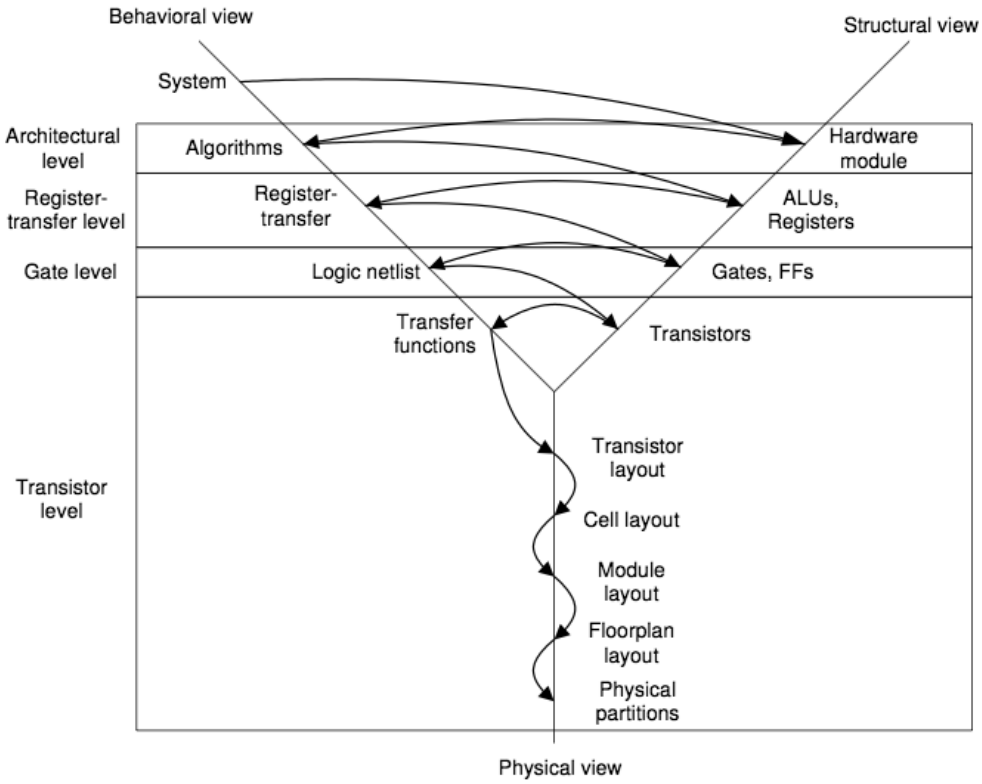


Figure 1.2. Y-chart representation of circuit model.

memories. However, only their behaviors are specified in this level. Thus, the components are treated as black boxes and their actual implementation is unknown.

- *Register-transfer level*: This level (RTL) consists of a set of transfer functions described by registers and functional units, such as ALUs. Thus, the RTL representation gives a better understanding on how the hardware components at the architectural design level will be actually implemented.
- *Gate level*: At this level, the transfer functions of a circuit at RTL description are transformed to logic equations. These equations are evaluated by a set of primitives (logic gates and flip-flops) that are obtained from a targeted technology library.
- *Transistor level*: This level describes the circuit using transistors, which resemble the exact implementation of the primitives from the gate level

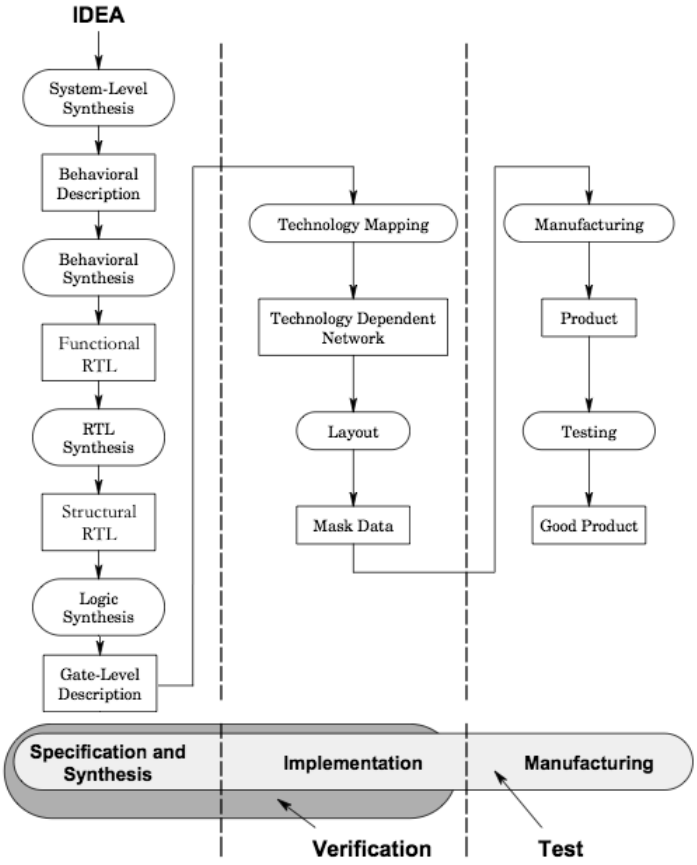


Figure 1.3. Design and production flow.

description on the silicon die. Thus, this level usually represents designs with increased complexity and with a large amount of information.

The descriptions of the functionality of the circuit at different levels of abstraction are provided usually with the use VHDL, Verilog or any other hardware description language (HDL) [3]. The transformation between different abstraction levels are commonly performed by synthesis algorithms. *Synthesis* is done to convert a less detailed model at a higher level of design abstraction to a more refined and detailed model at a lower level of abstraction [4]. The synthesis flow is shown on the left side of Figure 1.3 . Typically, the following synthesis steps are described from highest abstraction level downwards [5]:

1. *System-level synthesis*: At the highest level of abstraction, the specification

of a system is usually given by its functionality and a set of implementation constraints. In this step, the main task is to decompose the system into several subsystems and to give a behavioral description for each of then. The output of this is to be used as the input for behavioral synthesis.

2. *Behavioral synthesis*: This begins with a description specifying the computational solution of the problem, that is, in terms of operations on inputs in order to produce the desired outputs. In such descriptions, the basic elements are similar to those in programming languages, which include control structures and variables with operations applied to them. There are three major subtasks:
   - Resource allocation - selection of appropriate functional units,
   - Scheduling - assignment of operations to time slots,
   - Resource assignment - mapping of operations to functional units.

   After behavioral synthesis, a description at RTL, which consists of a datapath and a control part, is the output. The datapath usually consists of functional units, storage, and interconnected hardware and it performs operations on the input data in order to produce the required output. The control part, on the other hand, is typically represented as a state-transition table and it controls the type and sequence of data manipulations.
3. *RTL synthesis*: This process takes the RTL description output of the behavioral synthesis. In this step, resource allocation and assignment in the datapath can be improved and generation of appropriate controller architecture from the input consisting of states and state transitions for the control part is done.
4. *Logic synthesis*: A technology dependent description of the system serves as the input, specified by blocks of combinational logic and storage elements such as flip-flops. In this step, optimization and logic minimization are dealt with.

In the development of VLSI systems, specification and synthesis are followed by implementation, and then manufacturing. After that, production tests are performed to detect production errors. Testing the system may also be done during operation and maintenance. Hardware testing can be used to detect design

errors, but testing for all possible errors requires a lot of effort. To minimize the effort in testing and maximize the test coverage, there is a need to consider the test problems during the design process.

In order to ease the complex problem of test pattern generation and improve the manufacturing yield, different techniques have been considered such as inserting various DFT structures into the circuit while maintaining the original functionality of the design. The next section will discuss these DFT structures and the test flow.

1.2 Testing and Design for Testability

Reliable electronic systems are not only needed in the areas where failures can lead to catastrophic events, but also increasingly required in all application domains. A key requirement for obtaining reliable electronic systems is the ability to determine that the systems are error-free [6].

Testing verifies that the manufactured digital circuit works according to its intended functionality. It does not verify the correctness of the design, instead, its purpose is to verify the correctness of the manufacturing process. Thus, it can be said that *manufacturing test* is the verification of circuit fabrication [4]. When a circuit passes manufacturing test, it is expected to be reliable.

1.2.1 Failures and Fault Models

A failure is defined as an incorrect response in the behavior of the circuit. According to [7], there are two views of failures:

1. Physical or design domain: defects
    - On the device level: gate oxide shorts, metal-to-polysilicon shorts, cracks, seal leaks, dielectric breakdown, impurities, bent-broken leads, solder shorts, and bonding.
    - One the board level: missing component, wrong component, miss-oriented component, broken track, shorted tracks, and open circuit.
    - Incorrect design or functional defect.
    - Wearout or environmental failures: temperature related, high humidity, vibration, electrical stress, crosstalk, and radiation such as alpha

particles or neutron bombardment

2. Logical domain: structural faults. A fault is a model that represents the effect of a failure by means of the change that is produced in the system signal.
    - Stuck-at faults: single, multiple.
    - Bridging faults: AND, OR, non-feedback and feedback.
    - Delay faults: gate and interconnect.

Fabrication anomalies of digital circuits in the manufacturing process may cause some circuits to behave erroneously. Manufacturing test helps to detect the physical defects that lead to faulty behaviors of the fabricated chips. These defects can be detected by *parametric tests for chip pins* and *tests for functional blocks* [8]. Parametric tests include *DC tests* and *AC tests*. DC parametric tests are used for detecting shorts, opens, maximum current, leakage, output drive current and threshold levels. AC parametric tests are for testing, setup and hold times, functional speed, access time, refresh and pause time, and rise and fall time. These tests are usually technology-dependent and can be performed without any regard to the chip functionality. On the other hand, the tests for functional blocks check for the proper operation of a manufactured circuit by testing the internal chip nodes using input vectors. The corresponding circuit responses are compared to the expected responses for pass/fail analysis. These technology independent tests for functional blocks can be further divided into *functional tests* and *structural tests*.

- *Functional tests*

  Functional tests verify the functionality of each component in the circuit. To completely exercise the circuit functions, a complete set of test patterns is needed. For a circuit with n inputs, the number of input vectors will be 2^n. For instance, a 64-bit ripple-carry adder will have 2^{129} input vectors. To apply the complete test set to the *circuit-under-test* (CUT) using an *automated test equipment* (ATE), it would take $2.158x10^{22}$ years, assuming that the tester and circuit can operate at $1GHz$ [8]. Due to the exhaustive nature of complete functional tests, testing time is prohibitively large for logic blocks, which makes them not feasible for testing complex digital integrated circuits.

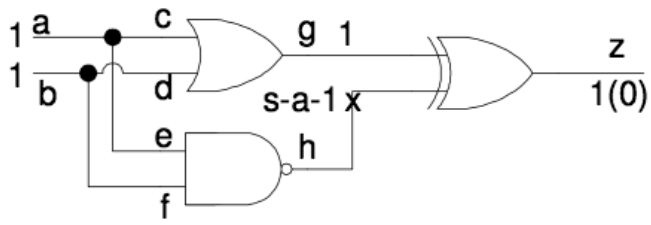


Figure 1.4. Single stuck-at fault example.

- *Structural tests*
  On the other hand, structural tests depend on the netlist structure of a design. Depending on the logic and timing behavior of electrical defects, different *fault models* are introduced to allow automatic algorithms to be developed for test generation, test application, and test evaluation. The most commonly used fault model is the *single stuck-at fault model*. It assumes a single line of the logic network to be stuck at a logic 0 (s-a-0) or logic 1 (s-a-1). An illustration of the stuck-at model is shown in Figure 1.4. In this example, the targeted fault is (s-a-1) at node h, which can be sensitized by the input vector $1, 1$ from inputs a, b. The correct response for this circuit at output z is 1. The fault response is therefore 0. When using the *single stuck-at fault model* for the 64-bit ripple-carry adder, only 1728 stuck-at faults would need to be excited with 1728 test patterns in the worst case scenario [8]. Another fault model that is gaining attention is the *delay fault model*, which will be detailed in Section 1.4.

1.2.2 Test Pattern Generation

After a model has been selected for structural test, the next step is to generate a set of test patterns. The outcome of test generation is a set of input vectors, which are applied to the circuit inputs to sensitize targeted faults, and a set of expected output responses, which are used for comparison with the actual circuit responses. Test generation is done by *automatic test pattern generation* (ATPG). There are two types of ATPG algorithms: *combinational ATPG* and *sequential ATPG*.

- *Combinational ATPG*

Combinational ATPG is one of the most important steps in the test flow. It is proven to be NP-Complete [8], which makes it prohibitively expensive in terms of CPU run time and volume of test data when applied to complex VLSI circuits [9]. Due to its complexity, many heuristics have been investigated [8] and all of them are based on four main operations: *excitation*, *sensitization*, *justification*, and *implication*. To generate a pattern for a stuck-at fault on a line (or wire), the fault is first excited, the response would then be sensitized to an observation point (e.g. primary output), and the logic values required on the input lines are justified. At the same time, the implications of logic values on other gates will be determined. Figure 1.4 can be used to illustrate the four operations. To excite the stuck-at-1 fault at node h, the value of that wire has to be set to 0. The effect of the fault is sensitized to the primary output z. In order to excite the fault at node h with a value of 0, the values of the primary inputs a, b are justified to be $1, 1$. This input combination implies the value of node g to be 1. By iteratively applying the four operations to all the faults in the circuit, a complete set of test patterns can be generated.

- *Sequential ATPG*

  If the internal state elements are not controllable, *sequential ATPG* is needed. There are several reasons why test pattern generation for sequential circuits is more difficult than for combinational circuits. One of the reason is that, the output response of the circuit depend not only on the input patterns, but also on the internal states of the circuit. These internal states may be synchronous or asynchronous. Another reason is, sensitizing a fault to a primary output requires the circuit to be driven to a known state. This sensitization process alone might require more than one pattern, and the order in which the test patterns are applied is critical. Furthermore, propagating the effect of the fault to an observable output may take several clock cycles. Also, multiple clock domains further complicate test pattern generation for sequential circuits, because the relations between clock domains must be followed to avoid any unpredictable behavior [10].

The difficulty in controlling and observing internal states makes sequential ATPG not applicable to large circuits. The enhancement of circuit testability

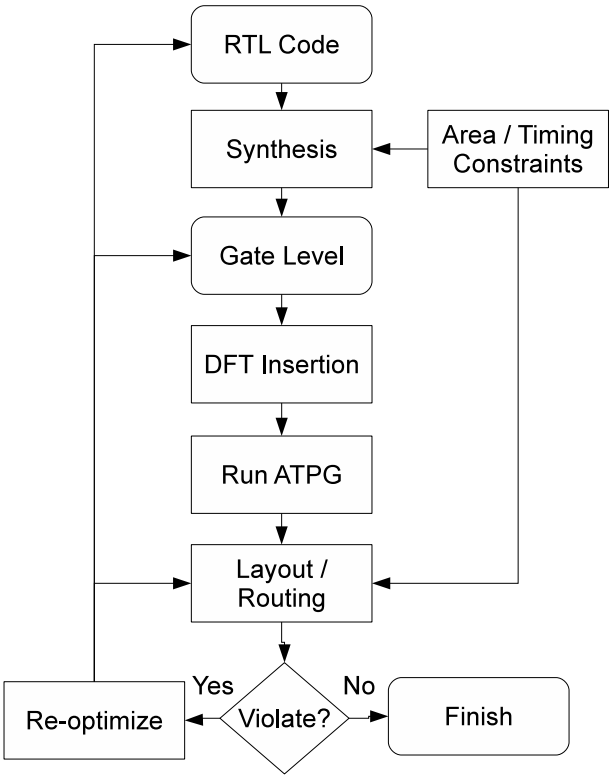


Figure 1.5. Design flow with DFT.

will allow ATPG to generate test patterns for complex VLSI designs in a more efficient way (i.e., tractable given the resources at hand). Thus, techniques to improve the controllability and observability of a design are needed.

Traditionally, a circuit testability was considered as an after though. Efforts were only done at the end of the cycle. However, this approach often led to low fault coverage or rising production costs due to the unforeseen increase in cycle time as size and complexity of VLSI circuits grow. As a result, design-for-testability (DFT) was introduced to account for testability within the design cycle [11]. Figure 1.5 shows the modified design flow when considering testability within the design cycle. In this scenario, DFT structures are inserted after the structural netlist at the gate design abstraction level is obtained from logic synthesis tools. Although considering testability within the design cycle may increase the development cost, it can be offset by the decrease in production cost and improved manufacturing yield [12]. The common DFT structures that are

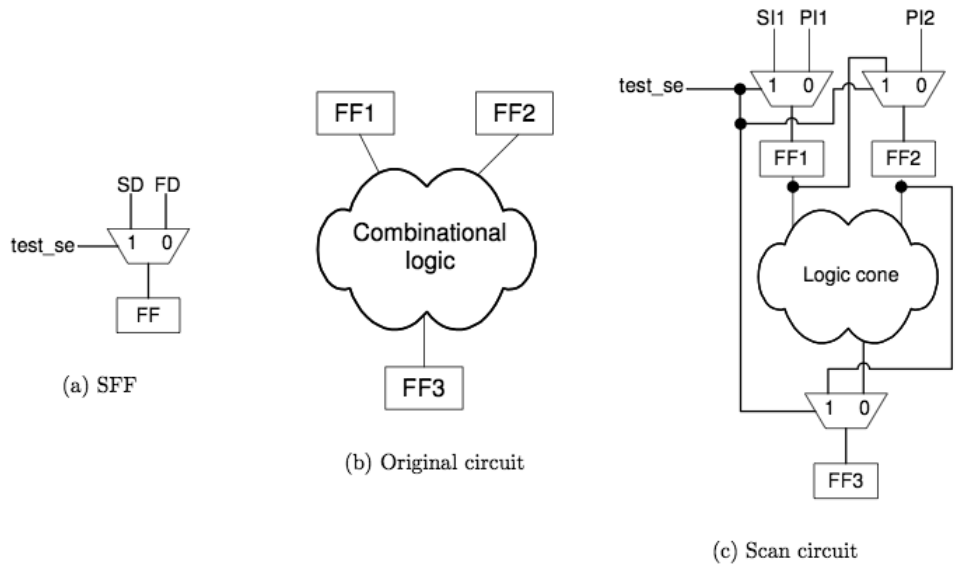


Figure 1.6. Full scan design.

used to enhance testability of VLSI circuits will be discussed in detail in the following section.

1.2.3 Design for Testability Structures

The most popular DFT structures are *scan design* and *built-in self-test*. These DFT techniques are explained in this section.

Scan Design It is common for VLSI designs today to have internal state signals which cannot be easily controlled from primary inputs or observed at primary outputs. This prohibits sequential ATPG to be tractable to complex VLSI designs, which may contain thousands (or even millions) of state elements. In order to enhance controllability and observability of large sequential circuits, the *scan method* is used to transform sequential circuits into combinational circuits from the test generation standpoint. Hence, the more tractable combinational ATPG algorithms can be used [10].

The scan method attempts to control and observe the sequential elements (i.e., FFs) inside a circuit by inserting a test mode such that, when the circuit

is in this mode, all the FFs are connected together to form one or multiple shift registers. These shift registers, also known as *scan chains* (SCs), are connected to primary inputs and primary outputs, which are called *scan inputs* (SIs) and *scan outputs* (SOs) respectively. By serially shifting arbitrary values into the SCs from SIs (called scan in), all the internal FFs can be set to desired states. Similarly, the internal FFs can be observed by scanning out their values in the SCs through SOs. As a result, the circuit becomes fully controllable and observable [8]. The complete controllability and observability of a scan design eliminates the need for sequential ATPG. Instead, the scan-flip-flops in the circuit are treated as *pseudo-primary-inputs* and *pseudo-primary- outputs*. Thus, from the ATPG standpoint, the sequential circuit is transformed into a combinational circuit.

In order to construct the SCs, original FFs in the design will have to be replaced with special *scan-flip-flops* (SFFs). A SFF has an additional 2-input multiplexor (MUX) that is connected to the input of the FF. The hardware structure of an SFF is illustrated in Figure 1.6(a). In the normal functional mode, the SFF reads the value from the functional data input of the MUX, thus retaining the original functionally of the design. Conversely, in the test mode, the SFF takes its value from the scan data (SD) input of the MUX, which is connected to another FF in the SC. Figure 1.6(b) shows a circuit without scan. In this circuit, the input FFs $FF1, FF2$ connect to a combinational logic block, which feeds the output FF $FF3$. By replacing the three FFs in Figure 1.6(b) with SFFs, the scan design is shown in Figure 1.6(c). In this scan design, the signal *test_se* indicates whether the circuit is operating in the normal mode or in the test mode. In the normal mode, the scan circuit has the same functionality as the original circuit. In the test mode, The FFs are connected to form an SC with the following order: $FF1, FF2, FF3$. This SC can be used to shift in test vectors through the scan input $SI1$.

Built-in Self-test With the help of the scan method, test patterns can be generated from ATPG and test application can be done next. The longest internal scan chain determines the test application time. The number of internal scan chains that can be directly driven by the tester depends on the constrained test access to the I/O pins. Thus, for circuits with many flip-flops but with a small

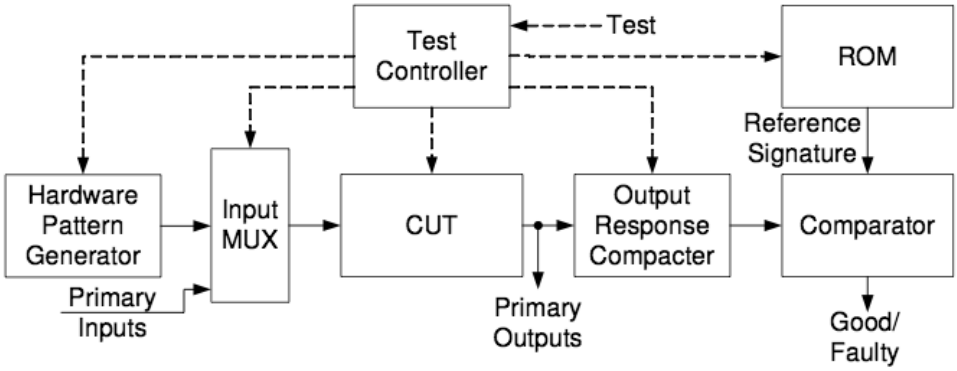
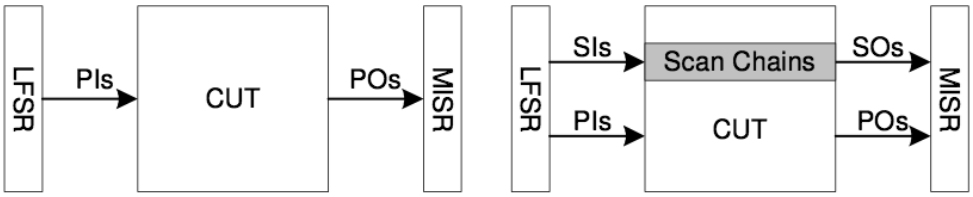


Figure 1.7. General BIST architecture.

number of test pins, the internal scan chain may be very long and in effect, the time spent by the circuit in external testers may be prohibitively large. Also, the huge amount of test data introduces another problem when external testers are employed. This is because storing the test data will require either reloading of buffers or the use of expensive testers with gigantic buffers. Hence, a new approach for test application called *built-in self-test* (BIST) has emerged [13]. Instead of feeding test patterns and observing circuit responses of the CUT from an external tester, on-chip test pattern generators and response analyzers controlled by a test controller are used in BIST. Figure 1.7 illustrates a general BIST architecture. There are two types of BIST schemes for applying test patterns to the CUT. They are *test-per-clock* and *test-per-scan*.

- *Test-per-clock*
  The architecture for the test-per-clock BIST system is shown in Figure 1.8(a). In this architecture, the PIs of the CUT are driven by the *linear feedback shift register* (LFSR), and the POs are connected to the *multiple input signature register* (MISR) to generate a response signature for the circuit. By generating and applying a new test pattern to the CUT continuously from the LFSR, a new set of faults are tested every clock period [8]. However, the BIST controller for the test-per-clock BIST system can be quite complex, which may result in high area overhead. Moreover, the use of large test registers (e.g., LFSRs, MISRs) can significantly impact both area and performance of the original design.



(a) Test-per-clock scheme (b) Test-per-scan scheme

Figure 1.8. BIST test application schemes.

- *Test-per scan*
  The test-per-scan BIST system is shown in Figure 1.8(b). In this system, the concept of test-per-clock is combined with the scan method. As a result, a two-step process is required for each new set of faults. In addition to the single clock period for conducting the test, a series of shifts for the SC are needed to initialize the circuit and read out all the test results. Therefore, the test-per-scan system will take several clock cycles per pattern. However, the test control and test hardware is non-intrusive since it reuses the available scan structure for test application. This leads to lower area and performance overhead when compared with the test-per-clock system. In addition, it fits easily into any designs which already have scan structures in place.

In addition to the above, details of other DFT structures can be found in [8, 10]. In this thesis, we restrict the discussion to the scan method due to its applicability to large circuits and different fault models, its suitability for both BIST and ATE-based test application, and its ease of integration in the VLSI design flow. To evaluate the effectiveness of different scan architectures, different parameters are introduced to quantify the benefits or drawbacks of each architecture. These parameters will be discussed in the next section.

1.3 Test Cost

Scan structures can significantly improve the testability of complex VLSI designs. However, the enhancement does not come for free. In order to quantify the added test cost of different scan structures, a number of parameters are used. The

parameters that are considered in this thesis are: area overhead, test application time, volume of test data and test generation time.

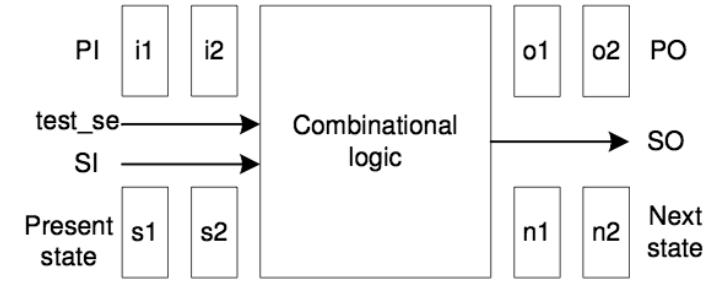
- *Area overhead*

  It is obvious that the insertion of scan structures for testability improvement introduces area overhead to the design [8]. In the case of the scan method, the increase in silicon area is due to the complexity of the scan device, FF, or latch. For instance, the gate overhead for the SFF in Figure 1.6(a) will be the input multiplexer, which is equivalent to four logic gates. In addition to the gate overhead, the scan method may require a significant amount of routing, which can impact the chip area. In a scan design, the test enable (test se) signal is routed to all FFs, and the output of each FF is routed to the SD input of the subsequent FFs in an SC. To reduce area occupied by the interconnect wires from the scan design, one can re-order the sequential elements in the chain. However, this effort to diminish routing overhead can only be performed at the layout generation or routing step in the design flow.

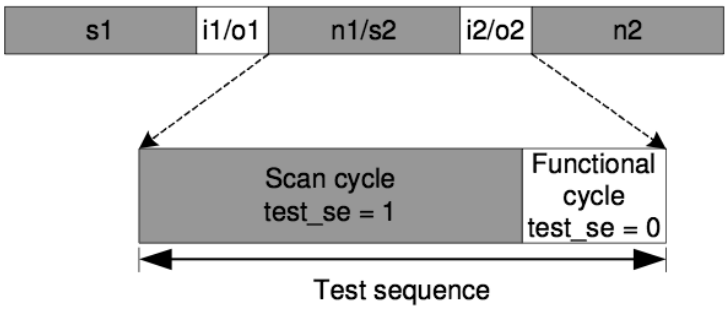
  In order to quantify the area overhead of our proposed method, we synthesize the circuit to gate level and generate the optimized area size result. The increase in area from the original circuit is due to augmented circuitry, which is expected to be lower than that of full scan design.

- *Test application time*

  Testing of scan circuits targeting faults in the combinational logic is a multi-step process. It involves shifting the test patterns generated by combinational ATPG into the SFFs, applying the shifted test patterns to the circuit, and shifting test responses out of the chip. The time it takes to complete this three-step process is the *test application time*. Figure 1.9 demonstrates the entire test application procedure for a circuit with one SC. The test patterns for the combinational logic are shown in Figure 1.9(a). There are two sets of test patterns in this example. $i1, i2$ are the parts of the test vectors applied at the primary inputs. $s1, s2$ are the parts of test vectors applied through internal FFs. $o1, o2$ and $n1, n2$ are the circuit responses available at the primary outputs and from the internal FFs respectively. Figure 1.9(b) illustrates the test sequences. For each sequence, the follow-



(a) Combinational test vectors



(b) Scan test sequences

Figure 1.9. Scan test sequences for single-clock designs.

ing steps are performed. Firstly, an input test vector for internal FFs is shifted into the chip by setting the test control signal (*test_se*) to 1 in the scan cycle. After that, an input test vector for primary inputs will be applied when *test_se* is set to 0 in the functional cycle. This allows the internal states to be updated and the circuit responses to be propagated to the primary outputs. Finally, the updated states are shifted out of the chip in the following sequence. They will then be used together with the expected responses for pass/fail analysis. As we can see from the example, the test application time is dominated by the scan time of internal FFs. To reduce test application time, one can divide FFs into multiple SCs which are driven simultaneously. Figure 1.10(a) shows the structure of a single SC. For this structure, the scan time of each test pattern for a test sequence will be n clock cycles. On the other hand, Figure 1.10(b) divides the SC into k segments. Each of these SCs has its own dedicated scan input and scan

output. The scan time for each pattern is then reduced to $\lceil \frac{n}{k} \rceil$ clock cycles. However, additional pins or a more complex pin-multiplexing scheme will be required for this structure.

- *Volume of test data*

  The volume of test data (VTD) represents the amount of data a circuit needs to achieve a desirable fault coverage for a targeted fault model. For a scan design, the VTD can be calculated using Equation 1.1.

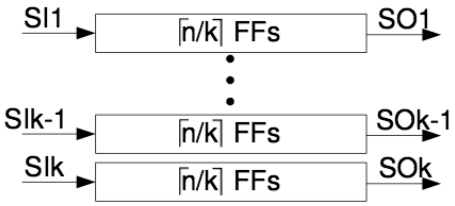
$$VTD = Num_{patterns} \times (2 \times Num_{SFF} + Num_{PIs} + Num_{POs}) \qquad (1.1)$$

  In this equation, $Num_{patterns}$ represents the number of test patterns for the design to achieve a desirable fault coverage. Num_{SFF} indicates the number of scan-flip-flops in the design. Num_{PIs} and Num_{POs} denote the number of primary inputs and primary outputs respectively. For example, if a circuit with 5,000 SFFs, 128 primary inputs and 128 primary outputs requires 2,000 test patterns to achieve a single stuck-at fault coverage of over 99%, the VTD will be around 20 Mbits. As size and complexity of VLSI designs increase, the VTD grows rapidly. To supply this massive VTD during test application, the size of ATE buffers will have to be large enough to store all the data in one test session. Otherwise, reloading of buffers will be required. In both cases, the cost of test will be increased [14].

- *Test generation time*

  The time it takes for ATPG to generate the test patterns that can possibly achieve high fault coverage is called *test generation time*. Usually, this is measured by the ATPG tool used. There are several algorithmic methods developed to address combinational and sequential circuits. In this work, only combinational ATPG is used because the proposed DFT method already reduces the ATPG problem of the sequential circuit into a combinational one. Usually, the process begins by identifying faults to test using fault models. Then, the test vector is generated for each fault. Fault simulation is done to calculate the fault coverage of the generated vectors. The time it takes to finish these steps is the test generation time. The speed of generating test vectors and fault simulation is dependent on the ATPG tool. Hence, improvements to reduce test generation time can

(a) Scan design with one scan chain



(b) Scan design with k scan chains

Figure 1.10. Scan designs with different number of scan chains.

be targeted on the improvement of the ATPG tool or by making the circuit as easily testable as possible for the available ATPG tool to handle.

Other test cost parameters that are considered in this thesis, but not quantified are:

- *Performance penalty*
  Performance penalty is mainly caused by the extra hardware from the inserted scan structures. The replacement of every FF with SFF shown in Figure 1.6(a) brings an additional MUX to the circuit. Each additional MUX located on the critical path of the design adds performance penalty equivalent to two gate-delays. It is obvious that as the number of FFs in the critical path increases, the performance penalty grows proportionally. Moreover, the extra wires used for the creation of scan paths raise the capacitive loading on the FF outputs, which may also increase the propagation delays. In general, the propagation delays in scan design increase around 5% [8].
- *Test development time*
  The time it takes to transform an original design into a testable design that meets all the environmental and/or timing constraints is called *test development time*. The test development process includes the insertion and optimization of scan structures. For example, the scan insertion step will allocate each FF in one of the multiple scan chains and the optimization

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
V_1	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8
V_2	γ_1	γ_2	γ_3	γ_4	γ_5	γ_6	γ_7	γ_8

Table 1.1. Example of a two-pattern test.

step may be required for ordering the FFs in each scan chain to reduce the routing overhead. If the optimization step cannot reduce the routing overhead sufficiently, some FFs may need to be reassigned to different SCs and the test development process proceeds iteratively. If, after a predefined number of iterations, the use of scan still violates the constraints, the original design may have to be changed to compensate for the added penalty of the scan structures. As a result, the prolonged test development time can directly affect the cost of the design.

1.4 Test Quality

Although it was proven that the single stuck-at fault model can cover a large spectrum of physical defects, new issues arise when circuits are implemented in nanometer technologies [15]. For example, to compensate for the decreasing effectiveness of quiescent current-based (I_{DDQ}) testing for circuits manufactured in smaller process geometries, the *delay fault model* is essential to screen the process variations that may affect only the circuit timing. The objective of testing for delay faults is to detect defects that adversely affect the timing behavior of a circuit without changing its logical operation under static conditions. A delay fault is detected when the amount of time a desired transition takes to propagate through the circuit from an initialization point to an observation point exceeds the period allowed for it [16]. To achieve delay fault detection, at-speed test application of two consecutive patterns is required, which imposes added constraints on scan development, as discussed next.

To detect a delay fault in a scan circuit, the primary inputs and internal flip-flops are used as initialization points, while the observation points include both primary outputs and flip-flops [17]. This scan-based delay fault test consists of two test patterns, V_1 and V_2. The first pattern V_1 is called the initialization pat-

tern and it must first be applied to the circuit to initialize the logic into a known state. The second pattern V_2, called the excitation pattern, is then applied in the successive clock pulse to trigger the desired transition. This process is called the *two-pattern test methodology*. One way to apply two-pattern tests is the *broad-side test application strategy*. In this strategy, the initialization pattern is first scanned into the SC and applied to the circuit to drive all the memory elements to known states. The excitation pattern is then derived as the combinational circuits response to the initialization pattern. One major disadvantage with this strategy is that it complicates the test pattern generation problem [16]. *Skewed-load* (or last-shift launch) test application strategy for two-pattern delay fault test methodology eliminates the need for sequential ATPG. It can also reuse the available DFT infrastructure provided for stuck-at fault testing. In this strategy, the pattern V_1 is loaded into the SCs prior to test application, with V_2 as the shifted version of V_1. This correlation between the pattern pair imposes some restrictions on the possible patterns that can be applied due to the SC order [17]. To demonstrate this restriction, Table 1.1 shows the application of the pattern pair V_1, V_2 in the FF set $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8$. We assume that the SC is connected such that a single shift of the SC causes each bit to move one position to the right. The value α is the bit data in the initialization pattern V_1. The value γ is the new value that is shifted into the SC while V_1 is shifted once to obtain the excitation pattern V_2. Because scan chain order will determine the correlation between the test patterns and hence it will influence the detectability of delay faults, it is essential to investigate new ways to insert scan structures that account for skewed-load delay fault testing.

1.5 Contributions of this Thesis

The main contributions of this thesis are as follows:

- **A new DFT technique for functional RTL circuits.** In this thesis, we propose F-scan, a DFT method applicable to functional register-transfer level circuits. The main idea is to utilize available functional elements and paths in the circuit as much as possible. In order to make the circuit *F-scannable*, every register must be included in an *F-scan-path*. We define the concepts involved in F-scan in this work, including the heuristic algorithm

in creating F-scan-paths. The motivation of this work is to improve gate-level full scan design in terms of area overhead and test application time. We perform experiments to show that F-scan is superior against gate-level full scan in these parameters.

- **A new constrained ATPG model for F-scannable circuits.** We propose a model for automatic test pattern generation that includes the constraints according to F-scan-path information in the F-scannable circuit in order to achieve high fault coverage and reduce over-testing as much as possible. The approach of constrained ATPG includes both the F-scan-paths and the combinational part of the circuit during combinational ATPG. In contrast, for gate-level full scan, only the combinational part of the circuit is included during ATPG. Experiments are done to show the effectiveness of constrained ATPG in achieving high fault coverage for stuck-at fault model. For delay faults, we also perform experiments to show reduction in over-testing using the broadside approach.
- **A new hybrid model for F-scan delay fault testing.** We extend the constrained ATPG model into a hybrid model for F-scan delay fault testing in order to achieve high delay fault coverage, whenever needed. In this model, the combinational ATPG can choose whether to use skewed-load or broadside in testing a certain fault. We perform experiments to present advantages of the hybrid F-scan model against the conventional skewed-load and broadside schemes for gate-level full scan. We also propose a model that allows testing delay faults for F-scannable circuits using full scan delay fault ATPG. This is done to improve the test generation time, which is a concern in the hybrid model due to long fault simulation time. Experiments are also done to show that the generated test vectors in this manner can be successfully used for detecting delay faults and that it allows significant reduction of the test generation effort, while keeping the same test quality.

1.6 Thesis Organization

This thesis presents a new design for testability method for circuits at the functional RTL. The remainder of the thesis is organized as follows. Chapter 2 intro-

duces the new DFT method called F-scan. First, the advantages of doing DFT at a higher level of abstraction are discussed. A review of related literature follows, which includes the different schemes studied in order to improve full scan design. F-scan is then explained by discussing preliminary concepts on assignment decision diagrams and the definition of functional scan, F-paths, and F-scan-paths. The DFT selection method and the algorithm specifics are also included in this chapter. In order to test F-scannable circuits, a general approach to F-scan test environment generation is discussed. Experimental results are provided and the chapter ends with a conclusion.

Chapter 3 details the description of constrained automatic test pattern generation for F-scan. This chapter introduces the F-scan constraint module, which is used to include the F-scan-paths during combinational ATPG. Discussion on the detection of redundant faults and testing F-scan-paths are also included in this chapter. We also present the experimental results comparing F-scan constrained ATPG and gate-level full scan. The chapter ends with an brief overview on how this method can be applicable to industrial designs and a conclusion.

In order to use constrained ATPG for delay fault testing that achieves high fault coverage, Chapter 4 introduces a new hybrid model for F-scan delay fault testing. It starts with the background on scan-based delay test techniques. The details of the new hybrid model is then discussed followed by the experimental results. Using the new hybrid model for F-scan delay fault testing, test generation time is compromised. Hence, discussion on how to improve it is also included in this chapter. An improved test generation model for F-scan delay fault testing using full scan delay fault ATPG for faster test generation is also introduced in this chapter. New test generation time results are provided and then, the chapter is concluded.

Finally, the thesis conclusion and suggestions for further work are given in Chapter 5.

Chapter 2

F-Scan DFT Method

To reduce the exponential complexity of sequential automatic test pattern generation (ATPG), various design for testability (DFT) approaches have been proposed. Using DFT schemes, the circuit structure and functionality change during test mode to allow easier testing. The most popular approach is scan design, which increases the testability of sequential circuits considerably [18]. Full Scan Design is widely used because it effectively reduces the sequential circuit ATPG problem into a combinational one.

Despite the increase in fault coverage, there are some disadvantages using scan techniques:

- increase in silicon area,
- larger number of pins needed for multiple scan chains,
- increased power consumption,
- increase in test application time, especially for single scan chain,
- decreased clock frequency.

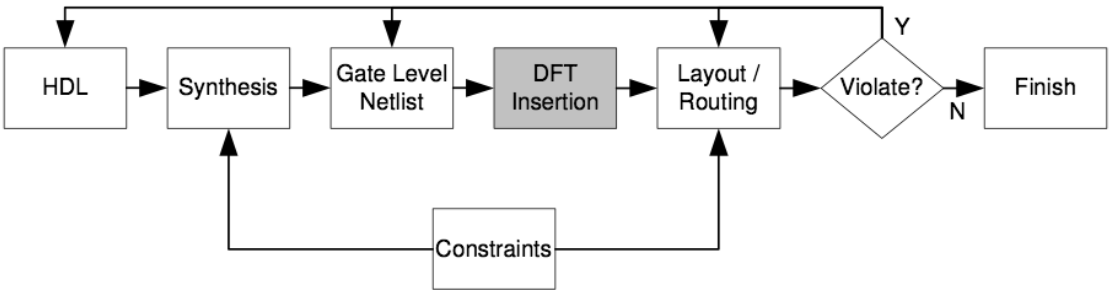
These penalties prove DFT, particularly full scan, to be very costly, especially for high-volume, low-cost applications. While the disadvantages of DFT hold true, our proposed DFT technique reduces chip area overhead and test application time as much as possible so that for high-density circuits, such overhead can be negligible. Also, with our proposed DFT method, the number of additional pins needed is minimal. Moreover, we apply DFT to register-transfer level (RTL) circuits wherein the number of primitive elements in the circuit is reduced.

The rest of the chapter is as follows. The comparison between doing DFT at gate level against RTL is explained in Section 2.1. A review of related literature is given in Section 2.2. In Section 2.3, F-scan and other preliminary concepts such as ADD are introduced. We describe the details of F-scan design methodology in Section 2.4 . We also explain the procedure for test environment generation in Section 2.6. The experimental results are provided in Section 2.8 and the conclusion of the chapter in Section 2.9.

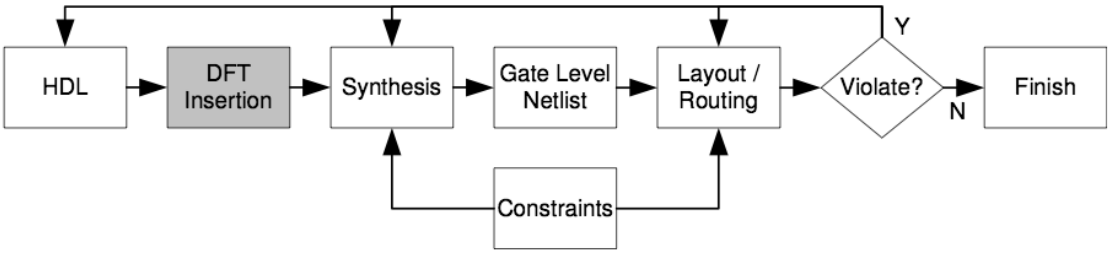
2.1 Gate vs. Register-Transfer Abstraction Level

It has been discussed that the scan chain architecture directly affects the cost of test for scan designs in terms of area. Nonetheless, constructing scan chains at different levels of design abstraction can also impact other cost parameters. For example, test application time can be influenced by introducing DFT structures at the RTL. Due to inherent parallel scan paths and the more natural flow of test vectors at RTL, close to that during normal function mode, not only is the TAT reduced but also over-testing. Moreover, there can also be an impact to test development time. This is because considering DFT early in the design cycle can produce a different design flow that allows synthesis tools to better meet design constraints by optimizing test logic and functional logic concurrently, which is not possible when DFT structures are inserted at the gate level [19]. However, due to the maturity of today's tools for gate-level testing, high level test generation is still not yet at par in terms of speed.

- Gate level DFT insertion
  Figure 2.1(a) illustrates the state-of-the-art design flow, where DFT structures are inserted after the RTL circuit description has been synthesized into the gate level structural netlist. In this case, the impact of DFT structures in terms of test cost will not be realized until late in the design flow. If design constraints are violated due to the addition of DFT structures, either the DFT structures or the original design will have to be modified to compensate the cost of test. This iterative process of circuit re-optimization may translate into lengthy development time, which in turn increases the cost of the design. Moreover, even though after reiteration of the design process, after gate-level DFT insertion, the circuit including test elements may still not be optimized for timing, power, and performance.
  Aside from such concerns, over-testing is a growing problem in circuits with gate-level DFT. This is because there may be test vectors generated that will put the circuit in such state that is not legal during functional mode. If a fault is detected using these test vectors, the circuit will be discarded. This may cause unnecessary loss of yield, which can be translated to wasted cost.
- RTL DFT insertion



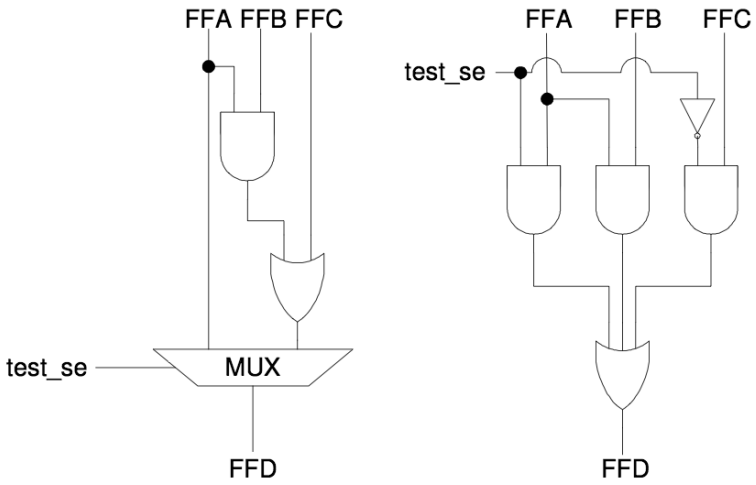
(a) Gate level DFT Insertion



(b) RTL DFT Insertion

Figure 2.1. Design flow with gate level and RTL DFT insertion. [20]

On the other hand, Figure 2.1(b) shows a different design flow, where DFT structures are inserted at the RTL. By introducing DFT at the RTL, synthesis tools can have greater flexibility to optimize the testable circuit to meet the design constraints by considering the test logic and functional logic simultaneously during synthesis. This is illustrated in Figures 2.2(a) and 2.2(b). Figure 2.2(a) shows the implementation of gate level scan, where an extra MUX is inserted to create an SP. In this case, the performance penalty for the critical path from FFA to FFD is 4 gate-delays. Conversely, by inserting scan at the RTL, synthesis tools can optimize the scan logic together with the original circuit, producing the optimized design in Figure 2.2(b). In this case, the delay in the critical path is reduced to only 2 gate-delays. The problem of wire delay can also be addressed by embedding test logic into functional logic to eliminate the need for long



(a) Gate Level scan (b) RTL scan

Figure 2.2. Optimization of scan logic.

wires in the construction of scan paths (SPs). Furthermore, this flexibility of synthesis tools may be able to lessen the area overhead of scan designs. This is because the synthesis tool can now better optimize the testable circuit to reduce the logic per FF ratio. In addition, introducing scan at the RTL can eliminate the need to re-order scan cells at later stages of the design flow (i.e., layout and routing stage) to reduce routing overhead from the scan structure based on two reasons. First of all, extra wires are not needed when SPs are created by reusing existing functional paths in a design. Secondly, in the case when additional wires are needed for SP creation, the inserted wires will not affect the timing closure of the synthesized design. This is because during RTL/logic synthesis, the scan structure is already taken into consideration by the synthesis tools. Thus, the resulted netlist will not contain wires that may influence the timing closure of the design after the creation of SPs. Moreover, constructing scan at the RTL (prior to logic and physical synthesis) can facilitate the embedding of additional constraints, which will tune the generated scan structure for different objectives, without affecting the timing closure of the scan circuit.

2.2 Review of Related Works

Due to the popularity of gate-level full scan design, there have been several DFT techniques proposed to improve it, both scan and non-scan based. Historically, Gupta et al. [21] introduced an approach to RTL DFT, which is a structured partial scan design that converts only the selected flip-flops into scan flip-flops. Partial scan can reduce hardware overhead, however, full-scan-based approaches, which include all FFs in scan chains, ensure stronger testability of circuits. Cost-free scan design [22] was first proposed for gate-level circuits to improve the area overhead of full scan design. H-Scan [23, 24] is a full-scan-based technique that utilizes paths between registers, but only through multiplexers. Although it can achieve the same fault coverage as full scan, further area overhead reduction can still be achieved by utilizing other available paths in the circuit. An improvement is orthogonal scan [25], which uses datapath flow as scan path. This method, though, requires multiple test configurations because it uses hold functions through load enable. *Hold function* is a logic that causes a register to hold the same value when the function is activated. This is necessary when a functional logic is shared by two scan paths because it allows scanning-in and -out of vectors from these paths one at a time, thus allowing the shared element to be used for testing. Our method does not employ this kind of function (with the exception of handling some state registers) because of the disadvantages of adding extra pins for controlling multiple paths during test and the expected longer test application time because simultaneous scan-in and -out cannot take place. Although we use some sort of initialization to scan-in the state register value first, which is a kind of hold function, we do not use *hold* whenever a functional operation is shared by candidate F-scan-paths. Moreover, our method is not only applicable to the datapath but the entire circuit.

Further improvements were done on the previous methods mentioned. Huang et al. [20] proposed the arrangement of registers in scan chains through cost rules to ensure the lowest possible area overhead for the circuit. Though this method tries to exploit available functional logic as much as possible without the use of hold functions, mask function is not considered. A *mask function* can be applied to operation logic, wherein the value from one input can be passed through the output by masking the other inputs. This function further reduces

area overhead, which is a DFT element widely used by our method. D-scan [26], on the other hand, uses *thru functions* (logic that allow values to pass through hardware modules) with predetermined control signals for the scan paths in the circuit. This work, however, utilizes hold functions to handle multiple paths that share the same thru function.

Techniques that utilize available circuity for test were also proposed in non-scan DFT techniques [27, 28, 29, 30, 31]. However, these approaches require a test controller and a means to isolate the controller part from the datapath part, thus increasing area overhead. Moreover, these methods are applicable to structural description of circuits, while our method handles functional RTL. Most of the designers are increasingly using functional description of circuits, which makes our proposed technique more relevant. Furthermore, our method deals with the circuit in assignment decision diagrams (ADD), which represent both the controller and datapath parts similarly. Thus, the application of both the DFT method and test is consistent for the entire circuit. The use of ADD also allows for easier manipulation of the circuit for DFT.

Our new approach to functional scan, F-scan, improves all of the mentioned previous works in terms of area overhead. F-scan organizes every register in the circuit in an F-scan-path by maximizing the use of available functional logic and paths to be used during scan, hence keeping hardware overhead due to test at the minimum. F-scans approach improves the other previously proposed methods (e.g., H-Scan, non-scan DFT) due to the following characteristics:

- F-path instead of identity path or I-path. *I-paths* or identity-paths are paths that are enabled by switching some pins such that data can be transmitted through them unchanged. The concept of I-paths has been defined in [32]. F-scan has the ability to use paths between registers with operations that can be masked, not just paths with multiplexers.
- F-scan is applied on functional RTL circuits, unlike the others that are done on structural RTL circuits. F-scan has the ability to be applied on the entire circuit uniformly, unlike the different approaches proposed before for datapath and controller parts.

F-scan is also a scan approach that can run-under-test using system clock, similar to at-speed testing of non-scan based methods. Furthermore, F-scan pri-

oritizes candidates to create F-scan-paths with the least possible scan time. Single F-scan-paths automatically allow parallel and simultaneous scan (dependent on the bit width), thus test application time is minimized. For further reduction, we also prioritize the use of multiple F-scan-paths, whenever readily available (dependent on the available primary inputs and outputs). The new concepts and methodology to create *F-scannable* circuits are provided in the next section.

2.3 F-Scan

In order to define *functional scan*, we first give a brief introduction about assignment decision diagrams and other preliminary concepts.

2.3.1 ADD and the Nine Symbol Algebra

Assignment Decision Diagram or ADD shown in Figure 2.3 is a representation developed for high-level synthesis that is complete, efficient, and partially unique. It can be used to describe functional RTL circuits in which the controller and the datapath parts are consistently represented.

ADD consists of four types of nodes: a) read nodes and b) write nodes (primary inputs or PI and outputs or PO, registers, or constants), c) operation nodes (arithmetic and logic), and d) assignment decision nodes or ADN (multiplexers) [33].

The concept of functional scan uses the following nine-symbol algebra used by Ghosh [34] for automatic test pattern generation (ATPG) of ADD circuits.

1. Cg (general controllability) of a register means it can be controlled to any arbitrary value.
2. Cq (controllability to a constant) of a register means it is controllable to any fixed constant value. This subsumes $C0$ (controllability to zero), $C1$ (controllability to one), and $Ca1$ (controllability to all one).
3. O (observability) of an RTL variable is the ability to observe fault at a variable.
4. Cs (controllability to a state) is similar to Cq but is applied to state registers to control to a particular state.

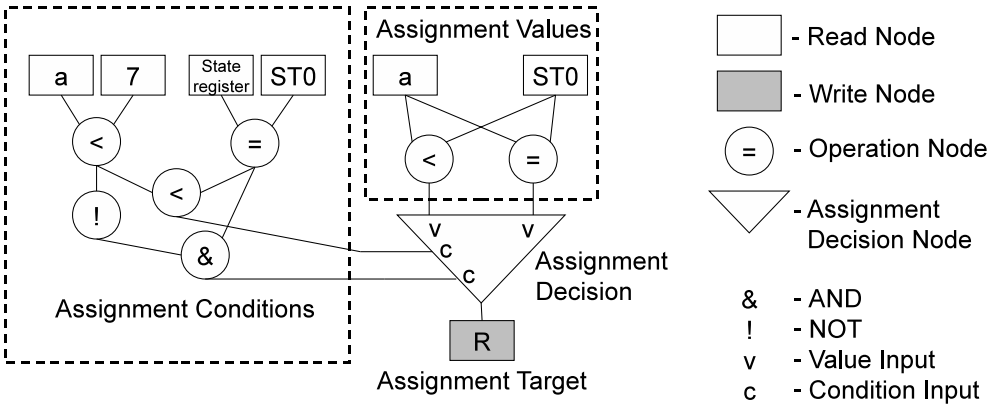


Figure 2.3. The assignment decision diagram.

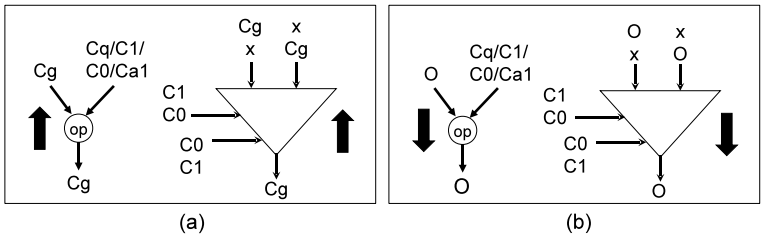


Figure 2.4. a) General controllability and b) observability of operation nodes and ADNs.

5. Other symbols are Cz (controllability to the Z value) and O' (complement observability), but these are not used for our study.

In Figure 2.4, controllability and observability in functional scan are illustrated with the use of these symbols. In Figure 2.4(a), we see that a value can be passed through an operation node as long as the other inputs to the node (side inputs) are constants such as Cq, $C0$, $C1$, and $Ca1$. Any arbitrary value can also pass through an available ADN by manipulating its control inputs to $C0$ and $C1$. Similarly, we can observe through operation nodes and ADNs as shown in Figure 2.4(b).

2.3.2 Functional Scan

We introduce the new concepts of functional scan by describing the means of justification and propagation in an ADD circuit. In Figure 2.4(a), we see that any arbitrary value can pass through available operation nodes and ADN, given by the symbol Cg, as long as the other inputs (side inputs) to the nodes are constants. Similarly, we can observe through operation nodes and ADNs as shown in Figure 2.4(b). Given these, we have the following definitions. Refer to Figures 2.6 and 2.7 for clarification of variables used in these definitions.

Assume that $p(X, Y)$ is a path from a read node X to a write node Y such that the side inputs of operational nodes and control inputs of ADNs along the path p can be controlled to fixed constants in an ADD circuit A.

Definition 1 (Essential value set.) *$EV(Y)$ is the **essential value set** of Y such that it is a set of values that can be essentially assigned to Y, which means the set of all values assignable to Y according to the functionality of A.*

Definition 2 (Essential error set.) *$EE(X)$ is the **essential error set** of X such that it is a set of errors that can be essentially detected from X, which means the set of all errors detectable from X. An error can be detected from the difference between a faulty and a fault-free value.*

Definition 3 (Essential Value Justification (EVJ) for $p(X, Y)$.) *Any value in $EV(Y)$ can be justified at Y by $p(X, Y)$ provided that any value in $EV(X)$ is justified at X.*

Definition 4 (Essential Error Propagation (EEP) for $p(X, Y)$.) *Any error in $EE(X)$ can be propagated to Y by $p(X, Y)$.*

Figure 2.5 illustrates how available functional logic is exploited for testing. Each register node can be both *essentially justifiable* and *essentially propagable* by controlling the side inputs along the involved path to $Cq/C0/C1/Ca1$. This means that for operation nodes in path p, since we know the constant value of the other input(s), we can compute for the value of the input (X) such that any arbitrary value within $EV(Y)$ can be passed to Y to make it *essentially justifiable*. On the other hand, in order for an error in $EE(X)$ to *essentially*

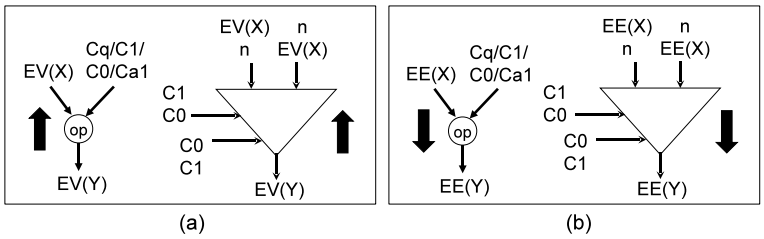


Figure 2.5. a) Essential value justification and b) Essential error propagation.

propagate through an operation node in path p, the difference between faulty and fault-free values should be detectable from X through Y. For ADNs in path p, any value/error can be retrieved from the ADN by controlling which input of the ADN connected to an essentially justifiable/propagable read node will pass its value/error to the essentially justifiable/propagable write node.

Definition 5 (Functional scan) *(abbrev. F-scan) is satisfied when all registers are made essentially justifiable and essentially propagable to be used for F-scan function. F-scan is a concept that uses available functional elements and paths to create scan chains for testing.*

The value ranges are obtained from the description of the ADD circuit and the error set depends on the fault model. This means that the ADD circuit may be augmented differently for various fault models for F-scan to be satisfied. To handle all errors, *complete error propagation* may be used instead. Similarly, if it is difficult to augment the circuit in order to obtain EVJ, *complete value justification* will be considered.

Definition 6 (Complete Value Justification (CVJ) **for** $p(X,Y)$**.)** *Any value can be justified at* Y *by* $p(X,Y)$ *provided that any value is justified at* X*.*

Definition 7 (Complete Error Propagation (CEP) **for** $p(X,Y)$**.)** *Any error at* X *can be propagated to* Y *by* $p(X,Y)$*.*

Both CVJ and CEP are strong conditions that are equivalent to justification and propagation conditions in full scan design.

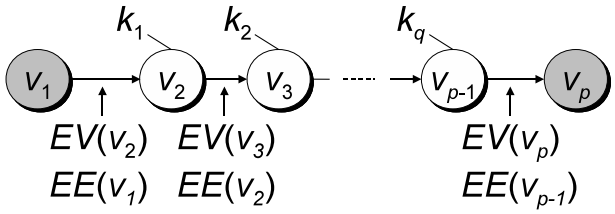


Figure 2.6. General representation of F-path.

2.3.3 F-paths and F-scan-paths

The main difference between gate-level full scan and F-scan is the method of building scan paths. Gate-level full scan arranges all flip-flops in single or multiple chains to shift test vectors while F-scan includes all registers of an RTL circuit in one or more scan chains called F-scan-paths, wherein the least possible scan time is achieved. While full scan augments multiplexers to connect flip-flops, F-scan exploits available functional elements and paths. F-scan-paths also allow scan-in and -out test vectors simultaneously, thus, similar to full scan, only one test pin is needed to activate scan to handle all registers. Another test pin will be needed to handle the state register whenever necessary, which is further discussed in Section 2.4.

We defined *F-path* in [35, 36], which represents the topology of a path in an ADD circuit from a read node to a write node as shown in Figure 2.6. Between the read node (PI or register), v_1, and the write node (PO or register), v_p, there may be operation nodes (v_2 to v_{p-1}) and ADN where value or error can *pass* through the path. Side inputs along the path should be made constant (k_1 to k_q).

Considering the path $p(X, Y)$ in the previous subsection and by referring to Figure 2.7, we have the following definitions.

Definition 8 *$p(X, Y)$ is an **Essential F-path** if:*

- *Case 1. X and Y are both registers: $p(X, Y)$ satisfies both essential value justification and essential error propagation for $p(X, Y)$.*
- *Case 2. X is PI: $p(X, Y)$ satisfies essential value justification for $p(X, Y)$.*
- *Case 3. Y is PO: $p(X, Y)$ satisfies essential error propagation for $p(X, Y)$.*

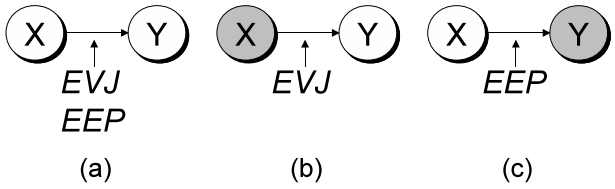


Figure 2.7. Essential F-path Illustrated. (a) Case 1, (b) Case 2, and (c) Case 3.

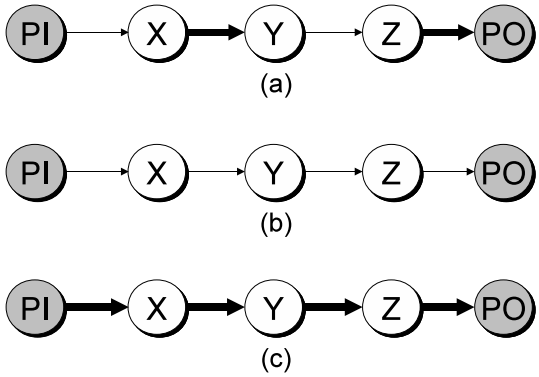


Figure 2.8. F-scan-path Illustrated. (a) Single F-scan-path, (b) Essential F-scan-path, and (c) Complete F-scan-path.

Definition 9 *$p(X, Y)$ is a **Complete F-path** if $p(X, Y)$ satisfies both complete value justification and complete error propagation for $p(X, Y)$.*

Definition 10 (Single F-scan-path.) *A concatenation of F-paths wherein the head is a PI and the tail is a PO. There are two special cases of F-scan-path.*

1. ***Essential F-scan path**. A concatenation of all essential F-paths.*
2. ***Complete F-scan path**. A concatenation of all complete F-paths.*

Definition 11 (Multiple F-scan-path) *is a set of mutually compatible (disjoint) F-scan-paths.*

Definition 12 *An ADD circuit is said to be an **F-scannable circuit** if every register in the circuit is included in an F-scan-path, wherein it appears once and only once.*

2.4 DFT Selection Method

We introduce a new functional RTL scan approach called *F-Scan design*, which makes any ADD circuit F-scannable. The preliminary concepts and the overview of the DFT algorithm are presented in this section.

2.4.1 Problem Formulation

In order to test an ADD circuit, we control and observe all read and write nodes by organizing all registers in F-scan-paths. Whenever there is no direct connection from a read node to a write node, the functional logic and path in between can be utilized by augmenting DFT elements that will allow these functional elements to be used for scan. A direct connection may also be augmented.

Definition 13 *The DFT for F-scannable ADD circuits is formalized as the following optimization problem.*

- ***Input:*** *an ADD circuit*
- ***Output:*** *an F-scannable ADD circuit such that there are m F-scan-paths defined as*

$$m = min\{n_i, n_o\} \tag{2.2}$$

where n_i is the number of PIs and n_o is the number of POs. We can also solve the scan-length using m. Given the number of register nodes, k, we have:

$$scan\text{-}length = \left\lceil \frac{k}{m} \right\rceil . \tag{2.3}$$

Shown in Figure 2.9 is an illustration of an ADD circuit with more POs than PIs. We choose m, which is the number of F-scan-paths $F_1, ..., F_m$, to be the minimum between the number of PIs (excluding reset and clock pins) and POs.

- ***Optimization:*** *Minimize area overhead (i.e., hardware of augmented DFT elements)*

  After determining the fixed number of F-scan-paths, we organize the registers to fit the computed scan-length per F-scan-path. There will be cases wherein the connection of registers in the circuit cannot satisfy the creation of F-scan-paths with length k/m. This means that other F-scan-paths may

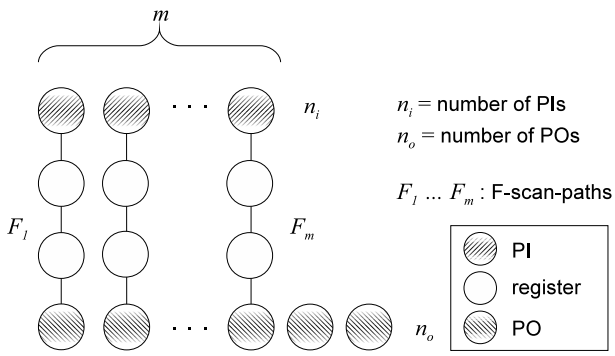


Figure 2.9. Determining the number of F-scan-paths.

be longer. Since scan time is a condition, even if a longer F-scan-path can potentially reduce area overhead further, F-scan-path slicing is still considered.

To assure that the least scan time is achieved without adding extra PI and PO as much as possible, we consider the condition of having m F-scan-paths. However, there may be situations when the bit widths of the registers in the circuit do not match the bit widths of the available PIs and POs. For different cases, we do the following:

1. If a circuit has no PI and/or PO for data transfer and the registers have the same bit width, we augment PI or PO with bit width equal to that of the registers in the ADD circuit. If the registers do not have the same bit width, we determine which bit width is common to most number of registers and consider that for the bit width of PI and PO to be augmented. If there are other registers with higher bit width to that, we slice those registers.
2. If a circuit has a one-bit PI and PO, we do not augment any PI or PO even if the registers in the circuit have higher bit widths. In this case, one test cycle of F-scan will be equivalent to that of full scan design, but the available functional elements and paths will be utilized.
3. If a circuit has a one-bit PI (resp. PO) and PO (resp. PI) with higher bit width, and the registers in the circuit have bit widths equal or less than that of the PO (resp. PI), we augment PI (resp. PO) such that the bit width will be equal to that of the register with the highest bitwidth. If

there are registers with higher bit width compared with the PO (resp. PI), then these registers will be sliced.

4. If the bit widths of the PI and PO in the circuit do not match, we choose the bit width of the F-scan-path according to the bit width that is common to most of the registers in the circuit. We then augment PI or PO or both to handle the F-scan-path during scan. For registers that have higher bit widths, slicing is done. For registers that have lower bit widths, we combine them to produce a group of registers with the same bit width as that of the F-scan-path.

Slicing is done by dividing a register according to the desired bitwidth and then, by connecting them in parallel through multiplexers. For example, if there is a 16-bit register to be sliced to two, the first eight bits will be connected to the other eight bits of the same register in parallel such that it will take two clock cycles to scan-in/out test vectors to the whole register. On the other hand, combining is done by scheduling the registers along the F-scan-path at the same time frame. This scheduling is explained more by the *test environment*, which will be discussed in Section 2.6.

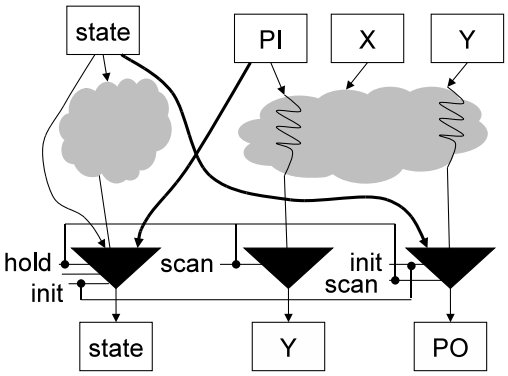
There is also a possible impact of the proposed DFT on logic synthesis. F-scan may introduce extra data flow at the ADD level. Some operations in the circuit may be involved in the extra data flow, which may prevent sharing of one operational module (at structural RTL) with other several operations (at functional RTL). If this happens, the resulting gate-level circuit may be large because the reduction of area during synthesis is not maximized. This is evaluated through our experiments.

2.4.2 Overview of the DFT Algorithm

The DFT algorithm consists of the following stages.

- **Stage 1.** Create a weighted connectivity graph (WCG) based on the information given by the ADD circuit. Here, all possible F-paths between each read/write node are exhaustively determined.
- **Stage 2.** Construct the F-scan-paths to make the circuit F- scannable.

Considering the number of possibilities, determining the F-scan-paths that are disjoint for an ADD circuit is regarded as an NP-hard problem. Thus, we



hold/scan	initialize	Mode
0	0	Normal
0	1	Initialize
1	0	Hold/Scan
1	1	Forbidden combination

Figure 2.10. Augmentation to handle state registers.

employ a heuristic algorithm to simplify it. The details are described in the next section.

2.5 DFT Algorithm Specifics

This section details the DFT algorithm. In this section, we describe how to handle state registers and what new ADD elements are needed to do masking for F-scan. The weighted connectivity graph and the local heuristic approach are also explained.

2.5.1 Handling State Registers

The state register is not readily accessible from PIs and POs and usually has a different bit width with the other registers, hence it cannot be readily included in the F-scan-paths. We augment the circuit to handle the state register as shown in Figure 2.10. The bold lines indicate the added connection from PI to state registers and from state registers to PO. To test the state registers, we first initialize by scanning-in state value from the PI to the state register. Then, we

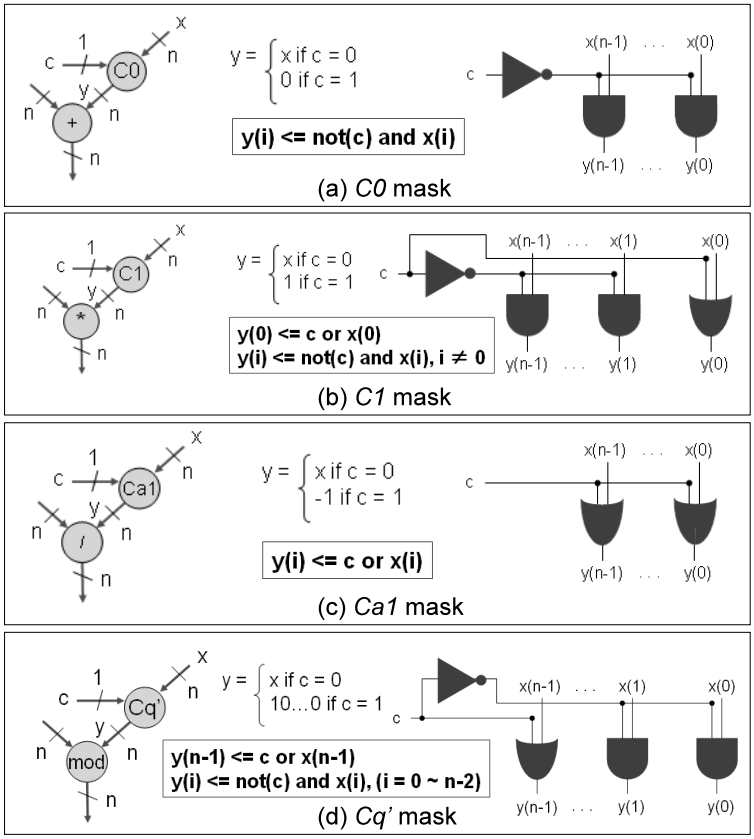


Figure 2.11. New mask functions for ADD illustrated.

set the test control inputs in scan mode for the entire circuit, while the state value is being held. When normal mode is done, new state value is scanned out during initialize, then we hold this value again to scan out the register values. Simultaneously, scan-in can occur while scanning-out. However, if there is a PI/PO pair available in the circuit that is not used by any F-scan-path or if there is an available F-scan-path that can include the state register, there is no need for the hold function.

2.5.2 New ADD Elements for Masking

Since there is no available ADD node that describes the ***mask function*** to keep an input to an operation node constant during scan, we have proposed the following new ADD elements in [35]. These elements are used as DFT elements

for F-scan. Figure 2.11 illustrates the new ADD elements and their corresponding gate-level representation, which are saved in the library.

Definition 14 (C0 mask.) *This mask is used for addition and subtraction operation nodes when the side input is not readily a constant. When the scan pin is set to* 0*, the output of this element is equal to the normal value of the line. If the scan pin is set to* 1*, the output of this element is* 0*.*

Definition 15 (C1 mask.) *This is used for multiplication and division operation nodes for them to pass any value from one input to the output without any changes. The output of this node is equal to the normal value of the line when the scan pin is set to* 0*. If it is set to* 1*, the output of this element is* 1*.*

Definition 16 (Ca1 mask.) *This is an alternative to* $C1$ *masking applicable to multiplication and division operation nodes as well. When the scan pin is set to* 0*, normal value of the line applies. If it is set to* 1*, all bits become* 1*.*

Definition 17 (Cq' masking for modulo.) *Since this constant is specific for modulo masking, we indicate it as* Cq'*. Being the highest* $2n$ *value within the range of the line, bitwise, the highest bit is* 1 *while the rest are zeros. This value* (10...0) *is the output of this node if the scan pin is set to* 1*. If it is set to* 0*, normal values of the line apply. This type of mask limits the range of a line, which is why using it is subject to the requirements of the essential ranges.*

2.5.3 Weighted Connectivity Graph

The *weighted connectivity graph* (WGC) represents the topology of an ADD circuit, which includes the read/write nodes and the cost information derived from F-path candidates. The cost rules are defined in [35], which depend on the amount of circuitry to be augmented to realize the F-path.

Determining all possible paths from a read node to a write node is a problem that grows exponentially with the circuit size. Thus, essential F-path candidates for each read-write node pair are limited to a number of possible paths in the circuit, which is chosen by the designer depending on the size of the circuit. If incompatibilities exist for all the current F-path candidates, another path is

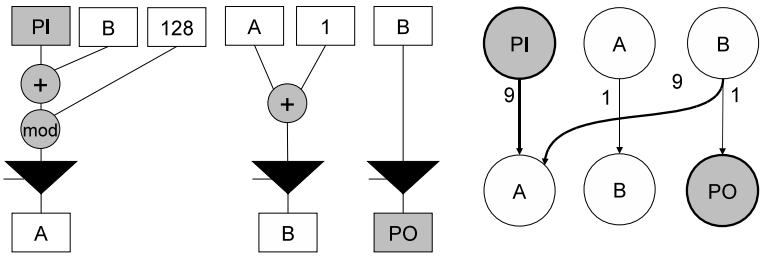


Figure 2.12. Sample WCG extracted from an ADD Circuit.

determined and the cost is compared with the full scan cost. The path with the least cost is to be chosen. Another candidate is a complete F-path, of which the cost is similar to full scan. This is chosen only when no available essential F-path is compatible to be used in the F-scan-path or if there is no essential F-path obtained at all.

In Figure 2.12, there are two candidate essential F-paths involving the read node B. The weights are indicated for each essential F-path. From A to B, for example, the cost 1 corresponds to the gate needed to augment the control input to activate this path during scan. Such information can be represented in WCG, where all paths merely indicate connectivity and weights.

2.5.4 Local Optimum Heuristic Approach

This ensures that in every local location (i.e., read-write node pairs in one scan time frame) the least area overhead due to test possible is achieved by choosing the candidate essential (or for the worst case, complete) F-paths that has the least cost. In the following, we only use the term F-path, which can be *either essential or complete*, depending on which has the least cost and is usable to create the F-scan-path.

1. ***PI/PO Priority.*** Once the number of F-scan-paths is determined, the primary inputs having F-paths to write nodes (registers) with the least cost are chosen. These F-paths are automatically the first in the F-scan-paths. Once chosen, backtrack is not applicable to change these F-paths (locked). Similarly, the F-paths with the least cost that connect read nodes

(register) to primary outputs are chosen and locked in the F-scan-paths.

2. ***Controllability.*** Starting from the first F-path in each of the F-scan-paths, the next F-path is chosen from the rest of the unconnected F-path candidates such that it is the least cost. The process continues until the registers are arranged in F-scan-paths to guarantee essential value justification that includes all registers. If in the process incompatibility is detected, backtrack is done until all F-scan-paths are mutually compatible.
3. ***Slicing.*** When the registers are arranged for control, it may occur such that one or more F-scan-paths are longer than the others. Since the length of the F-scan-paths is determined, we slice the long F-scan-paths and move the register or set of registers to shorter F-scan-paths to balance the lengths of all F-scan-paths.
4. ***Observability.*** To make all F-paths essentially propagable, we finally connect all F-scan-paths to the F-paths connected to POs. We choose the connection such that it is the cheapest one.

In the heuristic algorithm, we also consider the performance degradation in the cost. If the number of gate-delays in an F-path is more than the gate-delay of automatically augmenting a MUX in between registers, this F-path is considered very costly and will never be chosen to be included in an F-scan-path.

2.6 Test Environment Generation Procedure

The testability of the circuit-under-test (CUT) is guaranteed if at least both essential value justification and essential error propagation are satisfied for all registers. After applying F-Scan DFT for the circuit, all the registers are guaranteed to be essentially justifiable and propagable. The *test environment* of the CUT therefore consists of the scheduled signal assignment values and scheduled output response needed to perform a complete F-scan cycle. It involves the F-scan-in phase, test phase, and F-scan-out phase, wherein F-scan-in and -out are overlapped.

Test patterns, on the other hand, are generated through an available ATPG tool after synthesizing the circuit to gate-level. The *test sequence* is then derived by embedding the test patterns to the test environment. This includes the input test vectors and the test response. We use the generated test sequence to test

the F-scannable ADD circuit. This means that though the application of test sequence is done at ADD level, the generation of patterns is done at the gate-level. Hence, the fault coverage obtained after applying ATPG on the gate-level combinational circuit is not real. This is because the circuit may change from ADD level to gate-level. Thus, in the case where the synthesized circuit is different from the ADD-level circuit, fault simulation has to be performed in order to determine the true fault coverage of the test patterns.

F-scan-in Phase. To do *F-scan-in*, all read nodes used for data transfer in the F-scan-paths must contain their respective test patterns in order to justify these patterns to the write nodes. The necessary read nodes that will activate the F-scan-paths should also be controlled to their activating values. The F-scan-in environment therefore includes the schedule of signal assignments that completes the F-scan-in phase. This schedule depends on the order of the registers in the F-scan-paths. Direct value assignments are scheduled according to which F-scan-paths are to be activated, e.g. 1 or 0 for scan/hold pin and initialize pin. One cycle in this phase ends when all registers satisfy *essential value justification*.

Test Phase. The *test phase* happens by setting the circuit to normal mode where all read nodes (registers) are used as input-registers and the same registers (also write nodes) are used as output-registers for testing the circuit. Here, the test-mode environment includes the PI values (if needed), the output response, and the scan/hold/initialization pins assignment that will turn the circuit to normal mode, i.e. zero value.

F-scan-out Phase. To complete the test environment, *F-scan-out* is done. F-scan-in phase and F-scan-out-phase are overlapped, i.e. pipelined, after the first scan-in cycle. Thus, the signals that activate F-scan-in also enable F-scan-out at the same time. This is illustrated in Figure 2.13(d).

In F-scan-out phase, the values (and errors) in all registers are scanned-out. In order to check all the register values obtained after test phase, the test response, which is the output pattern of the circuit after test, is compared with the generated expected response.

Since the F-paths are not necessarily *I-paths* (or complete F-paths), the exact value of the generated test patterns may not be justified to the registers during F-scan-in. Also, the test response may not be the same with the expected response.

Thus, adjustments to the test sequence may be done. However, for simplicity, we did not adjust the patterns in the test sequence for our experiments and the results show that such difference is negligible.

Also, the F-scan-paths may be tested separately to ensure that the scan operation is without faults. This can be done by simply testing the CUT without going into normal mode.

2.6.1 An Example

A sample case to generate the test environment and test sequence is shown in Figure 2.13. The original circuit, E (without augmentation), is shown in Figure 2.13(a). The state register and the control values to the ADNs are not shown, and so these are not included in the examples test environment. Figure 2.13(b) shows the F-scannable circuit E. The F-paths are indicated with thicker lines and the mask is presented as a $C0$ element. The test environment is given by Figure 2.13(c), wherein it shows that a complete F-scan cycle for this circuit is equal to five clock cycles, $t0$ to $t4$. Figure 2.13(d) gives the test environment and the resulting test sequence given the test patterns $TP1$ and $TP2$ and test responses $TR1$ and $TR2$. Shown in the test sequence, the first F-scan-in occupies $t0$ and $t1$. Test phase happens in $t2$. From $t3$ to $t4$, F-scan-in and F-scan-out are overlapped. The same goes on until all the test patterns generated by ATPG are embedded to complete the test sequence.

2.7 Case Study

In order to present the effectiveness of our proposed method, we provide a motivational example of the application of F-Scan design to an ADD circuit, of which its results are compared with full scan and orthogonal scan. Note that this case study does not consider the entire circuit but only the data path part. The original ADD circuit is given in Figure 2.14(a). The augmented circuit through F-Scan, full scan, and orthogonal scan are shown in Figures 2.14(b), (c), and (d), respectively. Each register and pin in the circuit is 32-bit wide. For simplicity, the state register and the control signals to ADNs are not shown, since these are not considered during DFT.

Table 2.1. Pin and area overhead for the three DFT methods

DFT Method	No. of Extra Pins	No. of Extra Gates	No. of Masks Added	No. of Holds Added	No. of ADN Added
F-scan	1	140	1	0	1
Orthogonal Scan	2	270	8	2	0
Full Scan	1	485	0	0	5

Table 2.2. Test application time of the three DFT methods

DFT Method	Test Application Time (in cycles)
F-scan	$[(2+1) \times$ no. of test vectors$] + 2$
Orthogonal Scan	$[(4+1) \times$ no. of test vectors$] + 4$
Full Scan	$[(4+1) \times$ no. of test vectors$] + 4$

For this case study, we considered the type of full scan, which allows parallel scan-in and -out of test vectors using multiple paths. In orthogonal scan, we augmented the circuit by utilizing data paths for scan and opting to use hold functions to handle operation x, labelled in Figure 2.14(a). We evaluate pin and area overhead in Table 2.1. Orthogonal scan has an extra pin overhead compared to F-Scan and full scan because of the hold function augmented to deal with operation x. The number of augmented gates for each method is computed according to the number of mask functions, hold functions, and ADNs added to the circuit. Moreover, as can be observed in Table 2.2, F-Scan also has the estimated shortest test application time because it only takes two cycles for simultaneous scan-in and scan-out of test vectors in the circuit. Since the DFT methods used are full scan approaches, the combinational area remains the same and hence, the lengths of test vectors are approximately equal.

In the next section, we show results for ITC'99 benchmark circuits comparing F-scan and gate-level full scan only. It is because these two methods are both capable of applying DFT to the entire circuit uniformly, unlike other RTL DFT methods previously studied.

2.8 Experimental Results

We applied the proposed method to 20 ITC'99 Benchmark Circuits. No experiments are done on b16 and b19 because the ADD representation of these circuits cannot be produced. The original functions of the circuits are given in the Appendix. [1] This proves that F-scan is applicable to any circuit that can be described at functional RTL (ADD) and any functional path between registers can be used given that it satisfies essential value justification and essential error propagation.

In the previous section, we have shown that our method is superior to other scan-based techniques, such as orthogonal scan, without the consideration of the controller part. Thus, for our experiments in this subsection, we did not compare with other scan-based techniques because such methods have different approaches for the controller and datapath parts. We only compare with full scan because this technique, similar with F-scan, can be applied uniformly to the entire functional RTL circuit.

Table 2.3 presents the area overhead comparison of F-scan against gate-level full scan. For each benchmark, the synthesis was done using DesignCompiler of Synopsys. Column 1 contains the benchmark circuit names. Columns 2 and 3 show the number of flip-flops and the number of PIs and POs for each benchmark, respectively. Column 4 corresponds to the original area of the circuit. Columns 5 and 6 present the number of augmented pins and the resulting area overhead of gate-level full scan design, respectively. Similarly, such results for F-Scan are given in Columns 7 and 8. From these results, we can observe that for all benchmark circuits, F-scan has significantly lesser area overhead compared with gate-level full scan. For smaller circuits like b01 and b02, the area overhead of F-scan can be equal to that of full scan, but not greater. Moreover, for the biggest benchmark b18, the area overhead of full scan is 15.16% of the size of the circuit while for F-scan, the overhead is only 2.69%. This means that our proposed method is most effective for high-density circuits. Also, our results show that situations when there's an increased area overhead due to additional data flow caused by F-scan do not occur. In fact, there is even a case when the area of the circuit decreased after F-scan augmentation compared to the original area. For b12, due to F-scan DFT, optimization during synthesis allow binding of some

parts of the circuit, which cannot be done without the F-scan additional circuitry. Thus, this unique case of reduction of area occurred.

Next, we present the ATPG results of F-scan and gate-level full scan in Table 2.4. Here, we show the fault coverages, test application time, and CPU time (ATPG) for both cases. For all benchmarks, we generated the test patterns using TetraMax of Synopsys using combinational ATPG on synthesized circuits. Moreover, for F-scan, we performed fault simulation in order to obtain the true fault coverage of the test patterns applied on the ADD-level. This is because the synthesized gate-level circuit, where ATPG is done, may be different from the ADD circuit, where the test is applied.

In Table 2.4, the first column contains the ITC'99 Benchmark Circuits. Columns 2 and 6 show the ATPG fault efficiency and fault coverage for full scan and F-scan, respectively. Column 7 provides the real fault coverage for F-scan through fault simulation. Columns 3, 4, and 5 show the number of test patterns, test application time, and ATPG CPU time (in seconds) for full scan, respectively. The same information for F-scan are given in Columns 8, 10, and 11. Column 9 gives the F-scan length, which is the number of clock cycles per complete scan. The test application time of F-scan is based on this and the number of test patterns. Column 12 shows the CPU time for fault simulation. From these results, we can observe that F-scan is able to achieve high fault efficiency and fault coverage for all of the benchmark circuits. We can also observe from the results of fault simulation that the test patterns produced by doing ATPG on synthesized F-scannable circuit achieve high fault coverage even when applied at the ADD-level circuit. For most circuits, F-scan has better fault coverage compared to full scan. This is because upon augmentation using F-scan, the structure of the F-scannable circuit becomes different from the original circuit used for full scan. The additional circuitry improves the accessibility of most parts of the circuit, thus some faults that are not detectable by full scan are made detectable by F-scan. Furthermore, the increase in the number of faults of F-scannable circuits is due to the augmented circuitry. All faults in the augmented part are made detectable in F-scan as confirmed in the experiments, thus increasing the fault coverage of F-scan compared to that of full scan. On the other hand, even if the fault efficiency achieved by ATPG is 100% for F-scan, this cannot be always

achieved after fault simulation by the current ATPG approach and further improvements in the future are necessary. There is a need for a new ATPG method that guarantees 100% fault efficiency.

Test application time is also significantly reduced for F-scan. We assumed only one scan chain for full scan for simplicity. It is possible to create multiple scan chains for full scan and in that case the test application time will be comparable to F-scan's.

2.8.1 A Special Case: Circuit b12

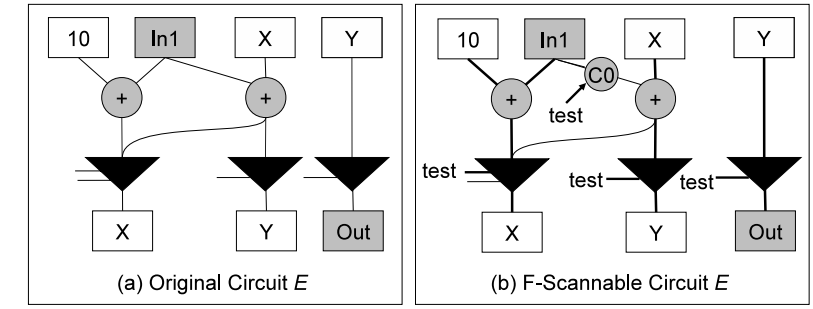
From the experimental results of area overhead in Table 2.3, it can be observed that circuit b12 has an extreme reduction in area overhead after augmenting for F-scan compared to other circuits. Thus, we call this circuit "a special case".

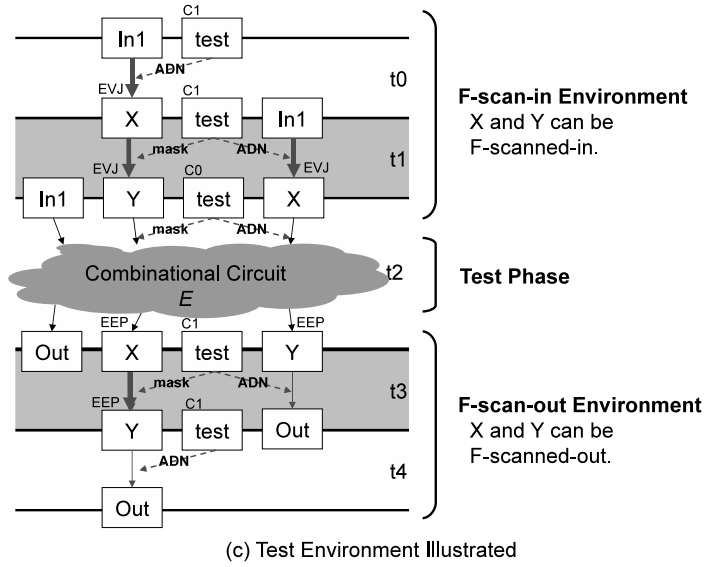
In other benchmark circuits such as b05, b08, b15, b17, and b18 that has memory elements, the F-scan approach is similar to full scan. This means that each memory element is put in a scan chain to scan-in and -out test vectors. However, for b12, the case is different. The circuit b12 has a RAM or random-access memory. The scheme for reading from the memory in b12's function uses addressing, which means that the entire memory can be read through the output of the signal connected to the memory (data-out bitwidth) and sweeping through all the addresses of the memory one at a time. The same scheme is also true for data-in. Since this functionality is already in the circuit, this is used for scanning-in and -out through F-scan, thus minimal circuitry is required for testing the memory. The only augmentation done is to control the address signals so that they will not be random during test mode.

The difference in area overhead can therefore be attributed to the optimization of the circuit during synthesis. Circuit b12 consists of four modules, each of them distinct from one another. After F-scan augmentation, sharing of elements in between modules after optimization is observed. The flow of the optimization process done by the DesignCompiler tool is outside the scope of this work, thus it is not explained here. However, it is notable that there are cases wherein F-scan augmentation can allow maximal optimization of circuits, such as in the case of b12.

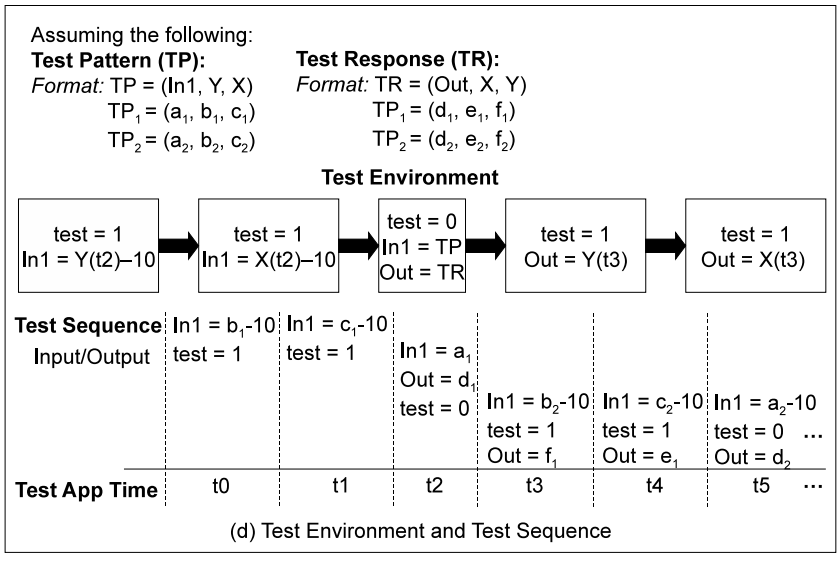
2.9 Conclusion

A new approach to functional RTL scan called F-scan has been presented. It maximally utilizes available functional elements and paths in the circuit to insert scan paths for testing. The proposed method reduces area overhead due to test compared to full scan design, as shown by the experimental results. Test application time is also superior against full scan. Although the test generation method discussed here is applicable for F-scan, further improvements especially for reduction of over-testing and test power can still be done. Moreover, 100% fault efficiency can still be achieved with an ATPG method that guarantees it. The technique for improving F-scan ATPG is discussed in the next chapter.





(c) Test Environment Illustrated



(d) Test Environment and Test Sequence

Figure 2.13. Sample case to show the test process.

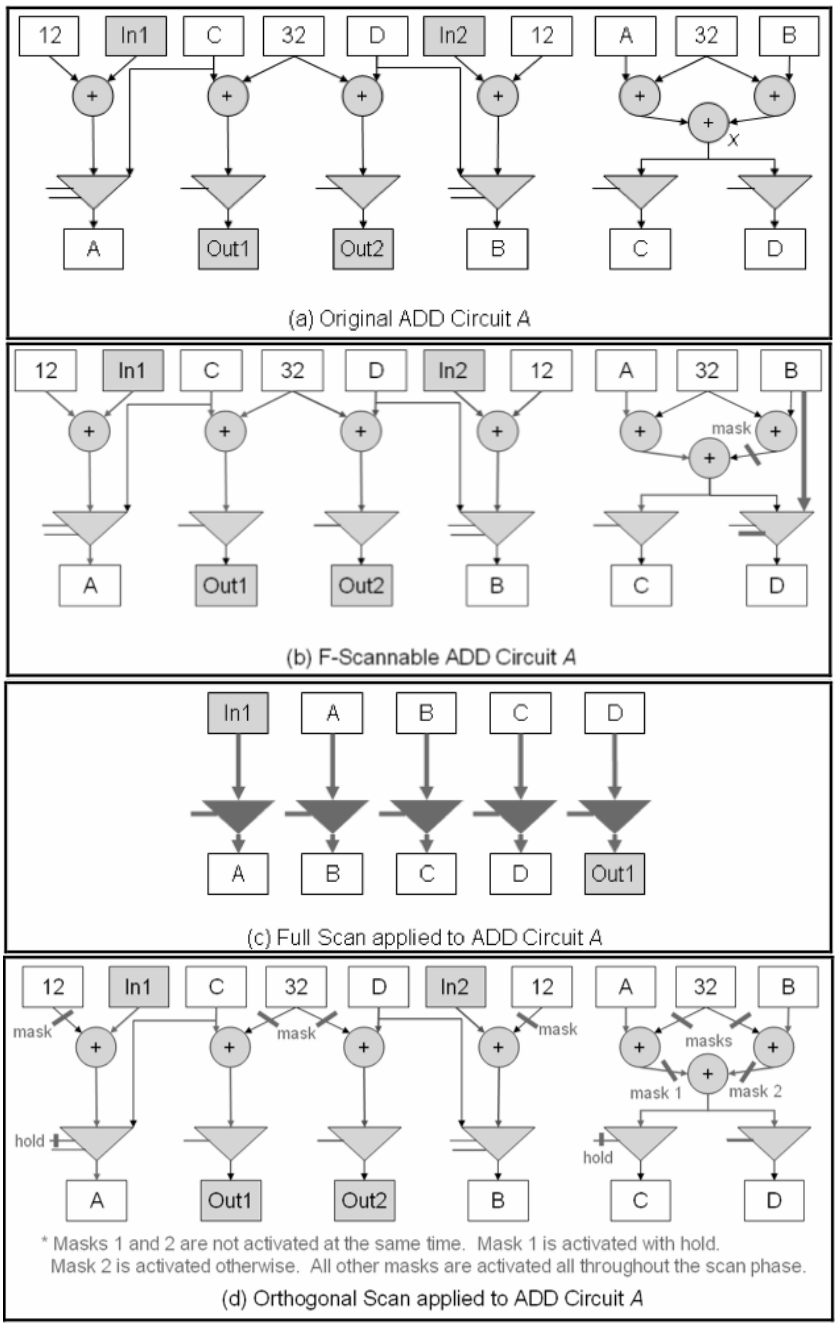


Figure 2.14. ADD Circuit A (a) and three DFT methods applied (b-d).

Table 2.3. Area overhead results

Ckts	FFs	PI/PO	Orig. Area (Units)	Full Scan		F-Scan	
				+P	AOH (%)	+P	AOH (%)
b01	5	2/2	86	1	23.26	1	23.26
b02	4	1/1	69	1	23.19	1	23.19
b03	30	4/4	360	1	33.33	1	7.22
b04	66	11/8	1014	1	26.04	1	2.66
b05	34	1/36	933	1	14.58	2+9	11.68
b06	9	2/6	135	1	26.67	1	21.48
b07	49	1/8	687	1	28.53	2+8	9.32
b08	21	9/4	299	1	28.09	1	16.05
b09	28	1/1	337	1	33.23	2	9.50
b10	17	11/6	291	1	23.37	1	18.90
b11	31	7/6	697	1	17.79	1+1	9.04
b12	121	5/6	2005	1	24.24	2+5	-52.27
b13	53	10/10	680	1	31.18	2	12.35
b14	245	32/54	11150	1	8.79	2+1	5.76
b15	449	36/70	8493	1	21.15	2+5	9.15
b17	1415	37/97	26336	1	21.49	2+5	3.69
b18	3320	36/23	87508	1	15.16	2+6	2.69
b20	490	32/22	23459	1	8.36	2+1	4.70
b21	490	32/22	23065	1	8.50	2+1	5.86
b22	735	32/22	34856	1	8.43	2+1	0.99

Table 2.4. Automatic test pattern generation results

Ckts	Gate-Level Full Scan				F-Scan						
	ATPG Fault Eff. (Fault Cov.)	No. of TP	TAT	ATPG CPU Time (s)	ATPG Fault Eff. (Fault Cov.)	Fault Sim Coverage	No. of TP	F-Scan Length	TAT	ATPG CPU Time (s)	Fault Sim Time (s)
b03	100% (96.87%)	47	1487	0	100% (100%)	100.00%	52	8	476	0	0
b04	100% (91.48%)	91	6163	0.91	100% (92.69%)	92.69%	107	10	1187	0.69	0.01
b05	100% (98.04%)	128	4514	0.01	100 % (98.46%)	98.43%	131	6	923	0.02	0
b06	100% (100%)	26	269	0	100% (100%)	100.00%	34	4	174	0	0
b07	100% (94.63%)	89	4499	0.01	100% (98.24%)	98.15%	88	5	533	0.01	0
b08	100% (98.76%)	62	1385	0	100% (100%)	100.00%	60	4	304	0.01	0
b09	100% (100%)	35	1043	0	100% (99.83%)	99.83%	57	28	1681	0	0
b10	100% (100%)	65	1187	0	100% (99.84%)	99.84%	71	3	287	0	0
b11	100% (100%)	116	3743	0.01	100% (100%)	100.00%	124	4	624	0.01	0.01
b12	100% (99.97%)	251	30743	0.02	100% (99.74%)	99.73%	166	44	7514	0.02	0
b13	100% (99.40%)	73	3995	0.01	100% (100%)	100.00%	79	20	1679	0.01	0
b14	100% (99.45%)	954	234929	0.8	100% (99.89%)	99.62%	985	9	9859	6.92	0.14
b15	100% (99.28%)	830	373949	51.23	100% (99.95%)	99.94%	910	8	8198	169.06	0.06
b17	100% (99.06%)	2132	3020327	62.39	100% (99.93%)	99.92%	2237	24	55949	115.83	0.34
b18	100% (99.12%)	5363	17813843	260.65	100% (99.78%)	99.63%	5622	48	275526	419.79	3.39
b20	100% (99.46%)	1747	858267	238.87	100% (99.69%)	99.58%	1799	18	34199	288.78	0.61
b21	100% (99.46%)	1728	848938	245.09	100% (99.73%)	99.58%	1756	18	33382	297.17	0.63
b22	100% (99.45%)	2414	1777439	253.74	100% (99.70%)	99.57%	2467	27	69103	308.13	0.7

Chapter 3

Constrained ATPG for F-scan

We have introduced F-scan in the previous chapter, which is yet incomplete in terms of test generation. In this chapter, we complete the method by proposing a constrained automatic test pattern generation (ATPG) technique that ensures high fault coverage. The problem with the ATPG method used in the previous chapter is that it does not guarantee 100% fault efficiency using fault simulation. It is important to achieve high fault efficiency and consequently, high fault coverage, in order to ensure high quality of chips. However, aside from high fault coverage, it is also increasingly important to consider reduction of over-testing to minimize yield loss. The work described in this chapter is aimed at achieving these.

An inherent feature of the constrained ATPG we propose here is the ability to generate test patterns that are functionally reachable states as much as possible. In this case, there is a possibility to prevent over-testing. This idea can be related to *pseudo-functional scan testing*, which has been proposed [37, 38, 39, 40, 41, 42, 43] to address the problems of scan tests. In this technique, any test pattern that is scanned-in conforms closely to a functionally-reachable state. The idea is that if the scanned state is functionally reachable, then that part of the circuit during test operates almost similarly with its normal mode. This way, over-testing and yield loss problems can be reduced [40].

Pseudo-functional scan test depends on the ability to identify reachable or unreachable states before ATPG. Our approach, however, is a constraint-based ATPG technique that tries to generate legal test patterns as much as possible. If this method is applied on the circuit with complete justification and propagation environment during test without any changes on the functional elements and paths, we can assure that all test patterns are legal. Our algorithm for F-scan described in the previous chapter aims for high fault coverage with the least area overhead possible. This is done by utilizing the available functional elements and paths for testing as much as possible, thus the circuit generally works close to normal mode during testing. Thus, as high fault coverage is achieved with F-scan,

there can also be a possibility of reducing over-testing. For this chapter, we focus on ATPG for stuck-at-fault model, but, path-delay faults and transition faults may also be tested using our approach.

The rest of the chapter is organized as follows. Section 3.1 discusses the need to reduce over-testing. In Section 3.2, we introduce and explain the constrained ATPG method for F-scannable circuits. Experimental results are given in Section 3.3. We briefly discuss the feasibility of using F-scan and constrained ATPG to practical designs in Section 3.4. Finally, this chapter is concluded in Section 3.5.

3.1 The Need to Reduce Over-testing

One of the difficult challenges identified in the 2007 International Technology Roadmap for Semiconductors [44] for test and test equipment is *potential yield losses*. Whenever any test or inspection process rejects as faulty a device that would function correctly in the target system, manufacturing yield loss occurs. One of the eminent causes of yield loss is over-testing.

Over-testing occurs when a digital circuit is thoroughly tested, usually through scan methods, such that it makes the circuit go to states that never occur when the circuit is in normal mode. If a fault is detected in the circuit in such states, the chip will be discarded. However, since this problem may never occur during the operation of the chip, discarding it may not be necessary. This is why over-testing also means yield loss.

Making sure that the circuit is tested well is necessary for high quality of chips released to the market. Nevertheless, since the industry are also concerned with profit, it is also better to have high yield while maintaining high quality of chips. DFT methodologies can be used to address such problem of balancing between over-testing and high fault coverage. F-scan is our solution.

With F-scan, high fault coverage can be achieved for stuck-at-fault model. Constrained ATPG for F-scan further improves the method in reducing over-testing by allowing the ATPG tool to produce test patterns that are functionally valid patterns as much as possible. By reducing over-testing for stuck-at-faults, it doesn't necessarily mean that the fault coverage should decrease. In Figure 3.1, it is shown that even with high fault coverage, there can be lesser over-testing compared to the one with lower fault coverage. For the sake of discussion of fault

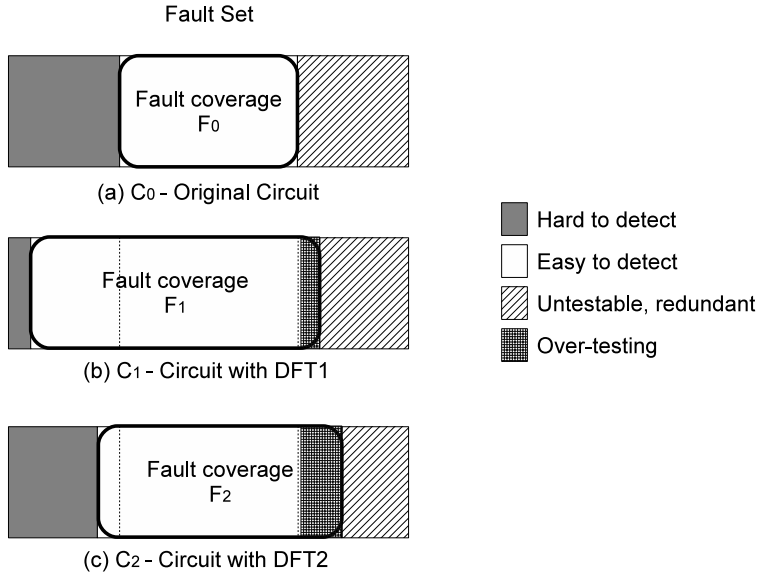


Figure 3.1. Fault coverage and over-testing relationship illustrated.

coverage and over-testing only, the method of ATPG is not specified in this case. The three circuits are: the original circuit C_0, the circuit with DFT1 (achieves high fault coverage but with less over-testing) C_1, and circuit with DFT2 (achieves lower fault coverage but results in higher over-testing) C_2.

The set of faults is the same for the circuits in Figure 3.1(a), (b), and (c). Figure 3.1(a) shows the original circuit's ATPG fault coverage, which includes the easy to detect faults. In Figure 3.1(b), the fault coverage of the circuit with DFT1 increases, which now includes a huge part of the hard to detect faults and a small portion of the redundant faults. This means that there is small occurrence of over-testing. In Figure 3.1(c), on the other hand, the fault coverage of the circuit with DFT2 is more than that of the original circuit's but less than than of the circuit with DFT1. However, despite the lower fault coverage than the circuit with DFT1, over-testing is increased due to more redundant faults detected.

Thus, the following can be said about the case given in Figure 3.1:

- Fault coverage: $F_0 < F_2 < F_1$
- Fault set of over-testing: over-testing$_2$ > over-testing$_1$

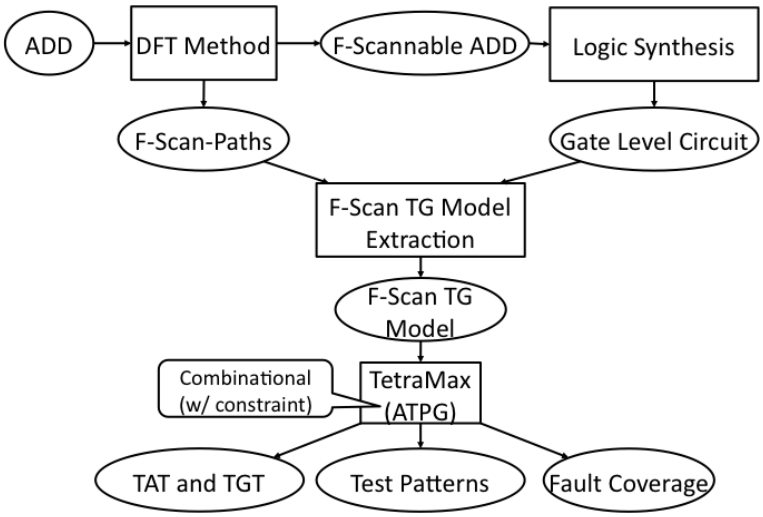


Figure 3.2. Flow of F-scan methodology with constrained ATPG.

With this, we can say that there is a possibility for our proposed constrained ATPG to achieve high fault coverage while reducing over-testing at the same time.

3.2 Constrained ATPG

Most of the works on constrained ATPG first determine legal or illegal states and then, feed these constraints to the ATPG tool. Our approach, however, embeds the constraints on the combinational gate-level circuit by creating the *F-scan test generation model* (FTGM) before ATPG.

The F-scan test generation model (FTGM) is based on the test environment generated for the F-scannable ADD circuit. The extraction of FTGM is summarized as follows:

1. Create combinational F-scan constraint module for each flip-flop in the ADD circuit that will allow both F-scan-in and F-scan-out.
2. Connect the combinational F-scan constraint modules to the synthesized F-scannable ADD circuit according to the schedule in the test environment.

The creation of F-scan constraint module and its connection with the syn-

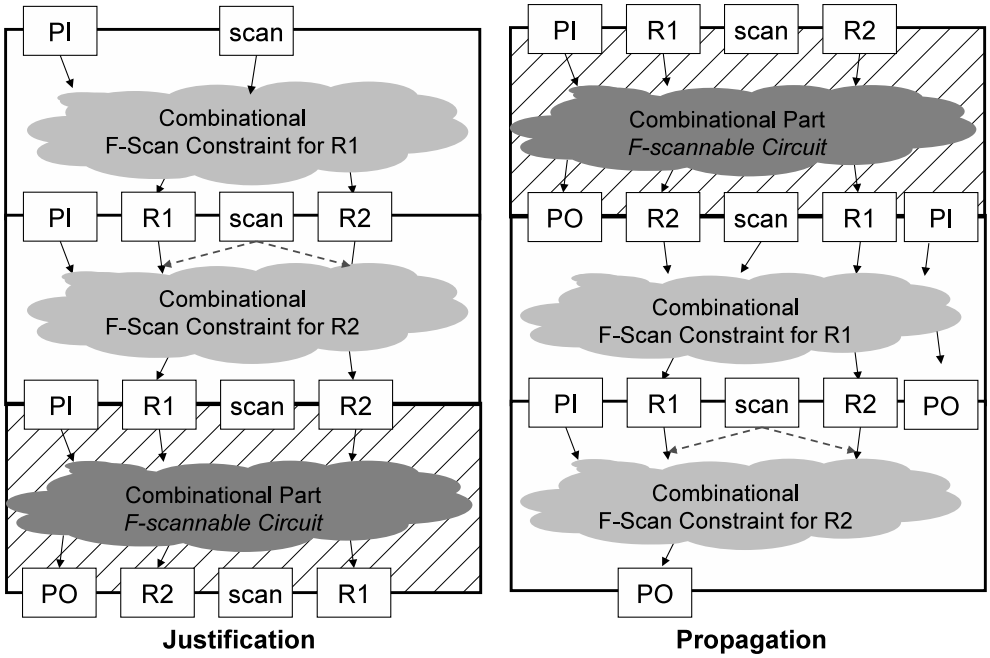


Figure 3.3. F-scan test generation model illustrated.

thesized F-scannable ADD circuit are further discussed in the next subsections. Figure 3.3 illustrates FTGM in a similar way we discussed the test environment in the previous chapter. Using FTGM during combinational ATPG, we control the signals that activate the F-scan-paths by adding PI constraints (1 for scan and 0 for reset). This way, the ATPG engine generate test patterns while considering the F-scan-paths activated, thus, all test patterns and responses generated are guaranteed to be justifiable and propagable through F-scan-paths. Since the F-scan-paths utilize available functional elements and paths in the circuit, the set of test patterns generated is within the functionally-reachable space of the circuit. In effect, occurrence of redundant faults is reduced and fault coverage is increased. Moreover, despite the high fault coverage expected with this method, over-testing is also dealt with because the constrained ATPG generates legal test patterns.

3.2.1 F-scan Constraint Module

After logic synthesis, we extract the combinational part of the gate-level circuit by converting all flip-flops into pseudo-primary inputs (PPIs) and pseudo-primary outputs (PPOs). This way, combinational ATPG may be applied, of which the testability is that of full scan design. Our approach, however, connects these PPIs and PPOs with the F-scan constraint modules before applying combinational ATPG.

The F-scan constraint module for each flip-flop is obtained from the combinational F-scannable circuit using the following steps:

1. Determine which F-scan-path the flip-flop belongs to. The F-scan constraint module will depend on the test environment of the F-scan-path that includes the flip-flop.
2. Each F-scan constraint submodule includes the F-path from a PI or register to a PO or another register. Thus, the first F-scan constraint submodule contains the combinational part that includes the path which connects the PI of the F-scan-path to the next register (according to the order of registers in the F-scan-path). Subsequently, the last F-scan constraint submodule includes the path which connects the last register in the F-scan-path to the PO. Thus, the number of F- scan constraint submodules for a flip-flop corresponds to the F-scan-path length. The creation of an F-scan constraint submodule is done by using the same combinational gate-level circuit, but activating only the F-scan-path that includes the flip- flop (adding PI constraints to control inputs scan and reset). This way, only the combinational part of the involved F-path is extracted.
3. When all of the F-scan constraint submodules are created, they are connected together according to the test environment. With the connection of the submodules, we have the F-scan constraint module for justification and propagation for a flip-flop. As an example, for the F-scannable $CircuitE$ in Chapter 2, Figure 2.13, we have the following order of registers and PI/PO in the F-scan-path: $In1 \rightarrow X \rightarrow Y \rightarrow Out$. So, if all of the PI/PO and registers are 1-bit and we are to create the F-scan constraint module for X, the first F-scan constraint submodule includes the path $In1 \rightarrow X$. Then, since this submodule already ends in X, it is

already considered as the F-scan constraint module for justification of X. This constraint module is then connected to the PPI of the combinational circuit corresponding to X. Similarly, the propagation F- scan constraint module has the following submodules: $X \rightarrow Y$ and $Y \rightarrow Out$. The PPO of the combinational circuit corresponding to X is connected to the submodule $X \rightarrow Y$. This submodule is then connected to the PPO of the submodule $Y \rightarrow Out$ corresponding to Y.

The creation of the F-scan constraint modules allows combinational ATPG to be used to generate test patterns for all kinds of fault models. For this chapter, we provide results for stuck-at-fault model. During experiment, since the combinational ATPG is done on the circuit with F-scan constraints, the fault coverage achieved by the ATPG tool is already the real fault coverage of the test pattern for the circuit. This means that the test pattern generated at gate-level automatically achieves the same coverage as when it is applied on ADD level, due to the constraint modules. Thus, there is no need for fault simulation.

3.2.2 Detection of Redundant Faults

For full scan, ATPG can identify all combinational redundancies, which are not tested. However, all sequential redundant faults in the original circuit become testable after full scan design. Thus, the ATPG have to generate test patterns for those testable faults that were redundant before DFT. That is, over testing. On the other hand, some combinational redundancies may be included in the test domain of constrained ATPG for F-scan. This is because F-scan may have added logic that will allow these redundant faults to be testable, since the DFT aims for high fault coverage. However, some sequential redundancies may be identified to be redundant for F-scan constrained ATPG and escaped for testing. This means that, the more that the circuit mainly uses functional elements and paths during test, the more redundancies can be identified by the constrained ATPG. This, in return, produces lower fault coverage.

3.2.3 Testing F-scan-paths

It is necessary to test the F-scan-paths to ensure that the scan-in data can be justified to all registers correctly and the errors can propagate along these paths

without masking. From the heuristic algorithm used to create F-scan-paths in the previous chapter, all possible errors can be propagated along F-scan- paths. Hence, testing F-scan-paths can be done by scanning-in and out test patterns without switching to normal mode, which can be done at operational speed. By doing so, the errors can be checked if there is any value in the scanned-out data that is not expected. If there are no errors in this test, error masking will never happen and the F-scanned-in values are always correct.

3.3 Experimental Results

For the experiments in this section, we have applied F-scan to 20 ITC'99 benchmark circuits, similar to the experiments done in the previous chapter. However, results for b01 and b02 are not shown anymore due because the results are equal for gate-level full scan and F-scan. The original functions of the circuits are given in the Appendix.

We converted each of the benchmark circuits to its ADD equivalent using the Exploration Tool of Y Explorations, Inc. Afterwards, we used our heuristic algorithm to apply F-scan, which is discussed in the previous chapter. The synthesis is done using DesignCompiler of Synopsys. We generated the test patterns using TetraMax of Synopsys using combinational ATPG on the synthesized circuit (for gate-level full scan) and on the F-scan test generation model circuit (for F-scan). As discussed earlier the fault coverage obtained for F-scan this way is already the real fault coverage of the test patterns as it is applied at the ADD level.

In Table 3.1, the benchmark circuit names are given in Column 1. Gate-level full scan results are shown in Columns 2, 3, 4, and 5 for fault efficiency (fault coverage), number of test patterns, test application time, and ATPG CPU time in seconds, respectively. The same results for F- scan are presented in Columns 6, 7, 9, and 10, respectively. Column 8 contains the F-scan length, which is used to compute the test application time for F-scan.

All of the results present equal or higher fault coverage for F-scan under constrained ATPG compared with full scan due to some structure changes in the circuit after F-scan augmentation. There is an increase in the number of faults in the F-scannable circuits due to the augmented circuitry, all of which are made detectable in F-scan, thus increasing the fault coverage of F-scan for most cases.

Moreover, the additional circuitry may improve accessibility of most parts of the F-scannable circuit, thus some faults that are not detectable by full scan may be made detectable by F-scan.

The fault efficiency is also complete for all benchmarks for both gate-level full scan and F-scan. The complete fault efficiency means that every fault is identified to be either redundant or testable. This is also true for gate-level full scan.

Moreover, the test application time is greatly reduced for F-scan compared to that of full scan, especially for high-volume circuits. Although the test generation time for F-scan is higher for all benchmarks, this is only a one-time cost and when compared with the advantages of a much shorter test application time that is a recurrent cost, the ATPG times cost can be compromised. The main reason as to why test generation time is longer for F-scan is because the current ATPG tools that we used have no information about our F-scan-paths, unlike for full scan, of which todays ATPG tools can handle easily and fast. If the ATPG tool used has the F-scan-path information and is able to utilize it efficiently, then the test generation time can be faster.

3.4 Application to Industrial Designs

Most large integrated circuits use full scan as fundamental DFT method. However, one constraint is its operating frequency, which is usually slower than operational speed. Moreover, it becomes more difficult to test multi-clock circuits. By using F-scan, the circuit is to be augmented for testing purposes before logic synthesis. Thus, area, power, and timing optimization are done on the circuit that includes the F-scan-paths. Since the F-scan-paths used for scan are functional elements and paths that are already in the circuit, testing can be done at-speed, which means at system clock. This makes it possible to test efficiently not just for stuck-at faults but also for delay faults. For simplicity, the registers and memory elements in the large circuits can be grouped according to clock speed. F-scan-paths can then be created while considering this grouping.

Furthermore, F-scans fault coverage is comparable to full scan, thus, it is also applicable to large practical circuits. The main difference is that, designing for F-scan has to be decided before synthesis. Thus, the designer can consider at an early stage as to which augmentation can allow the circuit to be F-scannable.

Since the designer has the information about the essential range of the values that can be stored in each register in the circuit, it is possible to create F-scan-paths considering the essential values and errors. This way, even during scan, only the applicable values can be scanned-in during test. If conflicts arise such that it is not possible to simply augment to allow scanning of essential values to a register, general controllability may be applied such that any value can be scanned, similar to full scan.

3.5 Conclusion

We have presented an approach to constrained ATPG applicable to F-scannable circuits. In this work, we showed the method for test generation for F-scannable circuit that guarantees generation of legal test patterns to the F-scan-paths. Since the F-scan-paths are derived from available functional elements and paths in the functional RTL circuit, the test patterns generated for F- scan testing are within the functional-reachable state of the circuit. Our results show high fault coverage for test patterns produced by our constrained ATPG. In this chapter, we have used stuck-at-fault model during ATPG, but the constrained ATPG can be used for any fault model. By using double capture for broadside delay fault testing, constrained ATPG has the capability to reduce over-testing. However, whenever high fault coverage for delay fault testing is required, it is necessary to extend the constrained ATPG model defined in this chapter to a hybrid model that can allow both skewed-load and broadside techniques to be used during testing. This hybrid model is introduced in the next chapter.

Table 3.1. Results for gate-level full scan combinational ATPG and F-scan constrained ATPG

Ckts	Gate-Level Full Scan Combinational ATPG				F-Scan Constrained ATPG				
	ATPG Fault Eff. (Fault Cov.)	No. of TP	TAT	ATPG CPU Time (s)	ATPG Fault Eff. (Fault Cov.)	No. of TP	F-Scan Length	TAT	ATPG CPU Time (s)
b03	100% (96.87%)	47	1487	0	100% (100%)	51	8	467	0.01
b04	100% (91.48%)	91	6163	0.91	100% (100%)	114	10	1264	0.61
b05	100% (98.04%)	128	4514	0.01	100 % (98.58%)	124	6	874	0.33
b06	100% (100%)	26	269	0	100% (100%)	33	4	169	0.03
b07	100% (94.63%)	89	4499	0.01	100% (98.69%)	97	5	587	0.43
b08	100% (98.76%)	62	1385	0	100% (100%)	66	4	334	0.21
b09	100% (100%)	35	1043	0	100% (100%)	55	28	1623	0.20
b10	100% (100%)	65	1187	0	100% (100%)	69	3	279	0.18
b11	100% (100%)	116	3743	0.01	100% (100%)	124	4	624	0.31
b12	100% (99.97%)	251	30743	0.02	100% (99.85%)	165	44	7469	20.22
b13	100% (99.40%)	73	3995	0.01	100% (100%)	90	20	1910	0.53
b14	100% (99.45%)	954	234929	0.8	100% (99.89%)	988	9	9889	6.92
b15	100% (99.28%)	830	373949	51.23	100% (99.98%)	901	8	8117	209.03
b17	100% (99.06%)	2132	3020327	62.39	100% (99.96%)	2236	24	55924	164.76
b18	100% (99.12%)	5363	17813843	260.65	100% (99.73%)	5499	48	269499	530.69
b20	100% (99.46%)	1747	858267	238.87	100% (99.59%)	1801	18	34237	398.42
b21	100% (99.46%)	1728	848938	245.09	100% (99.62%)	1739	18	33059	410.13
b22	100% (99.45%)	2414	1777439	253.74	100% (99.46%)	2457	27	68823	458.31

Chapter 4

F-scan Delay Fault Testing

Up until this point, the experiments done for F-scannable circuits involve stuck-at fault testing only. Delay fault model is another important fault model to be tested in today's circuits. The difference of delay faults as compared to stuck-at faults is that, they cause errors in the functionality of a circuit based on its timing. These faults are caused by the finite rise and fall times of the signals in the gates and the propagation delay of interconnects between the gates. One advantage of F-scan is the capability of its scan function to run at the same speed as its operation. This means that the same clock speed can be used during scan and normal mode, unlike in full scan wherein the scan mode is usually operated at slower speeds. Due to this, we can say that scan testing using F-scan is at-speed, which is possible because the F-scan-paths used for scan are integrated in the circuit at register-transfer level (RTL). Moreover, the F-scan-paths are synthesized with the circuit. Hence, timing, area, and performance are optimized for the circuit, which includes the F-scan-paths, during synthesis. This means that the structure of F-scannable circuits is built for effective delay scan testing.

In order to detect a delay fault, a pair of test patterns is usually required: the *initialization pattern* and *launch pattern*. The first pattern initializes the target faulty circuit line to a desired value. On the other hand, the second pattern launches a transition at the circuit line and propagates the fault effect to the primary output(s) and/or scan flip-flop(s). These two patterns are needed to detect transition faults: slow-to-rise faults and slow-to-fall faults [17]. Two approaches, namely, *skewed-load* [17] and *broad-side* [45], are utilized to apply two-pattern tests to standard scan designs. The next chapter describes how these approaches differ in the application of the second pattern of each pattern pair during delay scan testing.

In this chapter, we propose a hybrid model of the F-scannable circuit for delay ATPG that achieves high fault coverage. This approach combines the advantages of both skewed-load and broad-side. A previous work by Wang [46] presented a hybrid delay scan method for full scan. Wang's work chooses a small set of

selected scan flip-flops to be controlled by the skewed-load approach and the rest are controlled by the broad-side approach. Although results show that their approach achieves higher fault coverage for most of the circuits compared with the broad-side approach, the fault coverage of skewed-load is still superior. High fault coverage for delay scan testing is important to ensure the quality of the circuit. Thus, our method aims to achieve fault coverage that is equal or higher to skewed-load approach for full scan, without the disadvantages of hardware cost and scan-enable timing difficulties.

Another work has been previously done by Ko [47, 48], who proposed an RTL scan design for skewed-load at-speed test. One of the concerns of skewed-load is the difficulty of handling scan enable to go from test to normal mode during the testing cycle. This work agrees with our approach such that doing DFT at functional RTL makes it feasible for testing to be done at-speed. However, since Ko's approach is just a way to create full scan chains at RTL, F-scan has greater area overhead reduction. Moreover, the issue of over-testing is not tackled by Ko's work.

Another inherent feature of our hybrid model for F-scannable circuits is the integrated constraint in the hybrid model such that the generated test patterns are functionally reachable states as much as possible. In this case, there is a possibility to reduce over-testing, which is a result of scan methods that allow generation of scan patterns that are illegal during functional mode. Although this is not investigated by this work, the amount of reduction of over-testing depends on the F-scan DFT algorithm. Whenever F-scan-paths created in the circuit utilize available functional elements and paths for testing as much as possible, the circuit will generally work close to normal mode during testing. Thus, if there are lesser F-scan-paths created by adding circuitry for scan, over-testing is minimized. Other works [37, 39, 42, 43] have also proposed different techniques to identify the delay faults that are not functionally testable to reduce over-testing. These works generate test patterns for scan using the information of functionally untestable delay faults, which makes the scan tests as close to functional tests as possible.

The rest of the chapter is as follows. Section 4.1 introduces the scan-based delay test techniques. The proposed hybrid model for F-scan delay fault testing

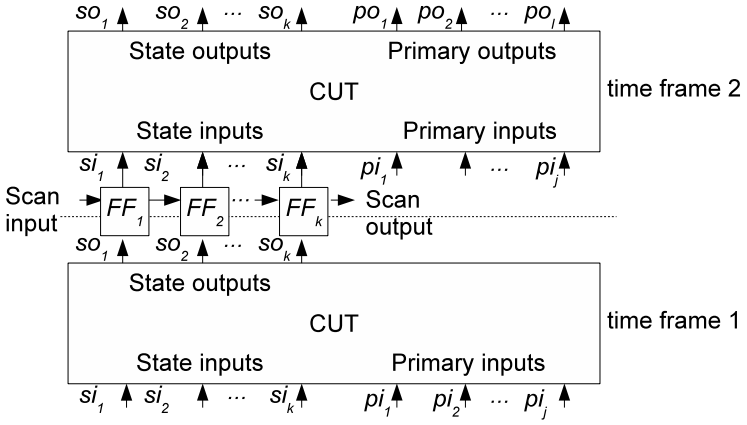


Figure 4.1. Two-frame version of full scan circuit.

is detailed in Secttion 4.2. Experimental results are provided in Section 4.3. Explanation on the need for further improvement of the hybrid model is given in Section 4.4. In Section 4.5, the new F-scan test generation model for delay fault testing is described, which uses the standard full scan delay fault ATPG scheme. Experimental results for this new model is shown in Section 4.6. The chapter concludes in Section 4.7.

4.1 Scan-Based Delay Test Techniques

Using full scan for any sequential circuit allows test patterns for transition faults to be generated through combinational ATPG on a two time frame version of the circuit. Figure 4.1 shows a general representation of a two-time-frame version of a full scan circuit employed on a sequential circuit [46]. The circuit has j primary inputs, $pi_1, pi_2, ...pi_j$, l primary outputs, $po_1, po_2, ...po_l$, and k state inputs and outputs, $si_1, si_2, ...si_k$ and $so_1, so_2, ...so_k$. State outputs so_i, where $i = 1, 2, ..., k$, of the first time frame copy are connected to the state inputs si_i of the second time frame copy of the circuit. This is a representation of the original circuit, wherein the state output so_i and state input si_i pairs are connected in a feedback loop through scan flip-flops FF_i. All the scan flip-flops use scan-enable to control whether to operate in normal or scan mode.

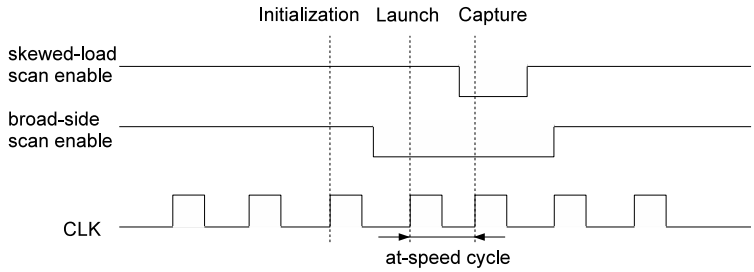


Figure 4.2. Delay scan-based test timing diagram.

The two traditional approaches to delay scan testing are skewed-load and broad-side. The timing difference for the scan enable signal between these two approaches are shown in Figure 4.2. For both methods, the initialization pattern of a delay test pattern pair is first loaded into all the scan flip-flops by n consecutive scan shifts, where n is the number of scan flip-flops in the scan chain. The launch pattern is applied after all the initialization patterns have been scanned in. Then, the response to the launch pattern is captured by the scan flip-flops at the next clock cycle after going to normal operation (at-speed). The clock speed during scan is usually slower compared to normal clock operation. Another difference between the two traditional methods is the source of the second pattern. This is described in the next subsections.

4.1.1 Skewed-load

In skewed-load approach, both the initialization and launch vectors of the pattern pair are delivered through the scan cells themselves [17]. The second vector is obtained by shifting in the first pattern by one more scan flip-flop and scanning-in a new value into the scan chain input. This means that both patterns come from the scan-input. Test patterns for this approach can be generated using combinational ATPG with little modification, similar to full scan. The scan enable signal is at logic high from initialization of first patterns to the launching of second set of patterns. Then, it is switched to logic low to configure the circuit to normal operation at the next clock cycle, which is also the capture cycle. Since the clock during normal operation is faster and the clock during scan is slower,

the switching of the scan enable signal needs to be fast and precise to match the timing of the circuit. Due to this, the scan enable signal should be driven by a strong clock buffer or complicated buffer tree. This requirement is very costly and it causes longer design time. Moreover, there is *shift dependency* in skewed-load approach, since only two possible patterns are available for the second pattern, wherein the difference is only at the first scan flip-flop that is directly connected to the scan input. If there is a transition delay fault that requires an initialization pattern with 0 at state input si_{i-1} and a launch pattern with 1 at state input si_i, then this fault is *shift dependency untestable*, assuming that the scan chain has non-inverting outputs for all scan flip-flops. [46]

4.1.2 Broad-side

Broad-side technique is more commonly used because it solves the timing problem of scan enable in skewed-load. In this approach, the first set of patterns is scanned in to the scan chain and the second vector is derived from the combinational circuit's response to the first vector [45]. Since the launch pattern does not come from the scan input, the scan enable signal can be switched to logic low right after initialization. This makes broad-side cheaper to implement. Despite the advantages of broad-side in terms of cost, fault coverage achieved by broad-side is lower compared to skewed-load [45]. Since the second pattern is produced by the circuit response to the first pattern, the number of possible patterns that can be applied as launch patterns is limited, unless the circuit can switch to all 2^n states, where n is the number of flip-flops. Thus, a fault is *function dependency untestable* whenever a state required to activate and propagate a transition delay fault is an invalid state [46]. Additionally, the test patterns for broad-side are generated using sequential ATPG using two time-frames. This makes test generation time longer. The number of test patterns generated by broad-side is also typically larger than those generated by skewed-load [49].

With these, there is motivation to create a method that can realize the advantages of both skewed-load and broad-side, while solving their individual disadvantages as well.

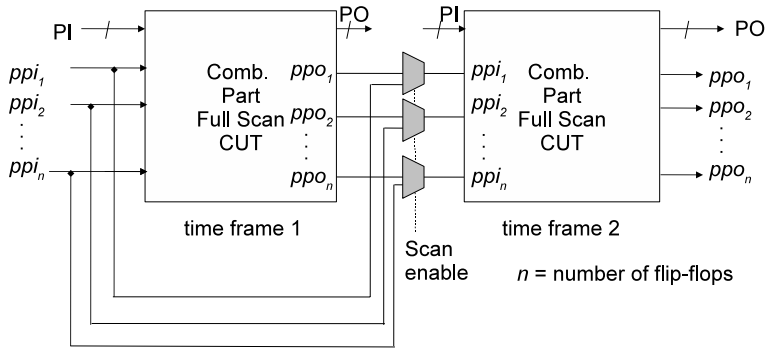


Figure 4.3. Hybrid two-pattern test generation model for full scan.

4.1.3 Hybrid Model for Full Scan

In this section, we show that a hybrid model for full scan can be used used to generate delay fault test patterns for circuits. This model is called hybrid because it allows generation of test patterns for delay testing using both broad-side and skewed-load approaches. In Figure 4.3, the circuit has primary inputs PI, primary outputs PO, and n pseudo primary inputs and outputs, $ppi_1, ppi_2, ...ppi_n$, and $ppo_1, ppo_2, ...ppo_n$. The pseudo primary inputs and outputs represent the input and output of scan flip-flops, thus, n is the number of scan flip-flops in the circuit. The pseudo primary inputs are connected to the pseudo primary outputs through the scan flip-flops. However, since the scan flip-flops are multiplexed, the scan enable signal controls which delay scan mode the circuit-under-test is configured. If scan enable signal is logic high during launch, the second set of patterns will come from the scan input (skewed-load). If it is logic low during launch, the second set of patterns will be the response of the combinational circuit to the first set of patterns (broad-side).

Although this approach improves the fault coverage of broad-side approach, the timing problem for the scan enable signal of skewed-load patterns is still not solved. The fault coverage of this hybrid method for ITC'99 benchmark circuits is shown in the experimental results. This ATPG model is different from what is used for the traditional skewed-load and broad-side approaches.

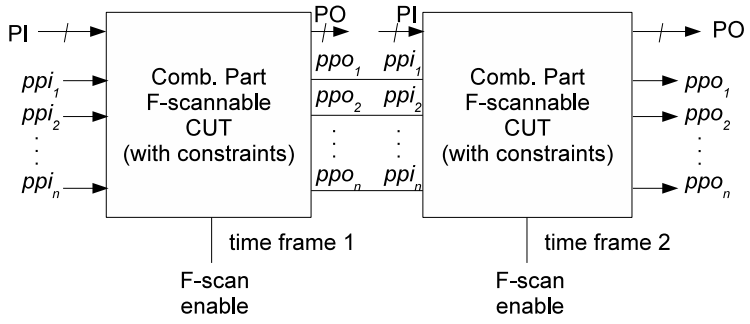


Figure 4.4. Hybrid two-pattern test generation model for F-scan.

4.2 Hybrid Delay Test Generation Model for F-scan

The proposed hybrid two-pattern test generation model for F-scan is similar to the hybrid model for full scan presented in the previous section. This is shown in Figure 4.4. The combinational part of the CUT is copied to two time frames, wherein constraints regarding the F-scan-paths are included. These constraints are included in order to generate test patterns that are functionally reachable states as much as possible. The pseudo primary inputs, $ppi_1, ppi_2, ...ppi_n$, and pseudo primary outputs, $ppo_1, ppo_2, ...ppo_n$, where n is equal to the number of flip-flops in the CUT, are each connected according to the F-scan-paths in both circuits. Since the F-scan-paths are created to connect registers, after synthesis to gate-level, each register is represented by a set of flip-flops of which the number depends on the bitwidth of the register. The pseudo primary outputs ppo_i, where $i = 1, 2, ..., n$, of the first time frame copy are directly connected to the pseudo primary inputs ppi_i of the second time frame copy. Unlike in the hybrid full scan model, there are no multiplexers that connect the pseudo primary pin pairs between the two time frame copies of the CUT. This is because the F-scan-paths are already in the combinational part of the CUT. One pseudo primary input and output pair represents a single-bit F-scan-path. Thus, multiple pseudo primary inputs and outputs represent the parallel and simultaneous scan operation of F-scan.

During ATPG, the first time frame copy is constrained to a constant value

at the faulty site while in the second time frame copy, stuck-at-fault is assumed. A transition between the non-faulty value to the faulty value is tested whether the fault is detected. Since this model is hybrid, the combinational ATPG tool automatically generates test patterns that can be either skewed-load or broad-side. If the F-scan enable signal of the first time frame is set by ATPG to be 1, the delay testing mode is skewed-load. If it is 0, the delay testing mode is broad-side.

Since the F-scan-paths are connected in such a way that registers propagate test patterns in parallel, there is no shift dependency in between flip-flops serially. Instead, the dependency is parallel. During skewed-load mode, each F-scan-path receives scan input according to the bitwidth of each of the F-scan-paths. This way, there are more possible patterns available for the second pattern. Moreover, since the combinational ATPG has a choice to either use broad-side or skewed-load mode to generate the test patterns, whenever a fault is hard to activate and propagate with broad-side, skewed-load approach is chosen instead.

Another advantage of this hybrid model for delay fault testing, aside from the improvement of fault coverage for the traditional broad-side approach, is that the scan enable timing problem of skewed-load is solved. Since the circuit can operate at the same speed (at-speed) during normal mode and F-scan mode, there is no need for additional circuitry to control the F-scan enable at the exact fast timing. The timing diagram for both skewed-load and broad-side testing modes are shown in Figure 4.5.

Regarding test generation time, improvement can be achieved when the ATPG tool used has information on handling F-scan-paths. The available ATPG tools today are already equipped to handle full scan, thus ATPG can be done more efficiently. In our experiments, we have used an available ATPG tool to generate test patterns, which deals with the circuit as if F-scan-paths are just part of the circuit but it is not known that they are used for scan, thus increasing the test generation time. Hence, a more efficient ATPG tool for F-scan can still be done.

4.2.1 Handling Error Masking

Delay fault testing for F-scan can be applied at operational speed for both scan and normal modes, wherein the F-scan-paths may have errors that may result

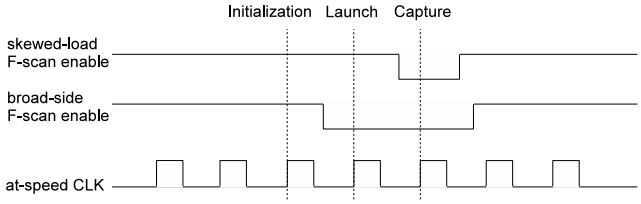


Figure 4.5. Hybrid delay F-scan-based test timing diagram.

in error masking. From the heuristic algorithm used to create F-scan-paths described in Chapter 2, all possible errors can be propagated along F-scan-paths. Thus, we can test the F-scan-paths beforehand by scanning-in and -out test patterns without going to normal mode at operational speed and then, by checking if there are errors. If there are no errors, then error masking will never happen when testing the F-scannable circuit.

4.3 Experimental Results

In our experiments, we applied both F-scan and gate-level full scan to ITC'99 Benchmark Circuits. F-scan is applied based on the heuristic algorithm described in Chapter 2. The synthesis and combinational ATPG are done using DesignCompiler and TetraMax of Synopsys, respectively. The combinational part of each benchmark is copied to two time frames and the constraints of the F-scan-paths are included. The fault list is first generated for each benchmark, then these faults are tested using the hybrid model. On the other hand, gate-level full scan chains are added to the benchmarks using DesignCompiler. Three ATPG methods were done for full scan. For skewed-load approach, TetraMax is used using the basic scan method ATPG, which is combinational. For broad-side, the fast sequential ATPG of TetraMax for two time frames is used. For the hybrid full scan method, the broad-side approach is done first, followed by the skewed-load approach, which is used to detect the faults that are not detected using broad-side. The results for fault coverage of the three full scan approaches are shown in Table 4.1. Hybrid full scan results are compared with hybrid F-scan results in Table 4.2. The number of test patterns for each method are given in Table 4.3.

Table 4.1. Results for gate-level full scan delay fault ATPG

Ckt	**Faults**	**Skewed-load FC (FE)**	**Broad-side FC (FE)**	**Hybrid FC (FE)**
b03	852	82.63% (100%)	95.89% (100%)	97.77% (100%)
b04	2620	83.21% (99.69%)	94.24% (99.73%)	94.66% (100%)
b05	3148	87.26% (99.56%)	84.34% (99.27%)	92.57% (99.49%)
b06	402	91.54% (100%)	88.81% (100%)	97.51% (100%)
b07	1694	90.50% (100%)	86.48% (100%)	97.17% (100%)
b08	814	85.38% (100%)	88.21% (100%)	96.44% (100%)
b09	784	96.81% (100%)	96.17% (100%)	99.62% (100%)
b10	858	92.89% (100%)	89.51% (100%)	97.20% (100%)
b11	2072	86.20% (100%)	85.23% (100%)	95.22% (100%)
b12	6050	94.21% (100%)	90.83% (100%)	98.60% (100%)
b13	1756	92.14% (100%)	88.44% (100%)	97.10% (100%)
b14	36376	96.71% (97.20%)	83.28% (84.66%)	98.32% (99.12%)
b15	23202	79.47% (98.22%)	58.19% (60.54%)	93.40% (99.35%)

In Table 4.1, the fault coverage results of the hybrid model for full scan are superior against the results of both broad-side and skewed-load approaches for all circuits. This proves that the hybrid model is the best among all methods in terms of fault coverage. Note that in these three approaches, the ATPG models are different from each other. The hybrid model for full scan is also different compared to the hybrid model for F-scan ATPG. In Table 4.2, we compare our method, hybrid F-scan, to the superior full scan delay fault ATPG model, hybrid full scan. For most of the circuits, except b12 and b13, the fault coverage of hybrid F-scan is better than that of hybrid full scan. There is an increase in the number of faults for most circuits due to circuit augmentation done by F-scan, all of which are made detectable in F-scan, thus increasing the fault coverage of F-scan for most cases. Nevertheless, we can also observe that the number of faults for b05 and b12 using hybrid F-scan is lower than that of hybrid full scan. This can be attributed to the effect of increased accessibility of registers and memory

Table 4.2. Fault coverage results for hybrid full scan and hybrid F-scan

Ckt	**Hybrid Full Scan**		**Hybrid F-Scan**	
	Faults	**FC (FE)**	**Faults**	**FC (FE)**
b03	852	97.77% (100%)	1074	99.35% (100%)
b04	2620	94.66% (100%)	3116	96.37% (100%)
b05	3148	92.57% (99.49%)	2668	93.07% (100%)
b06	402	97.51% (100%)	566	98.23% (100%)
b07	1694	97.17% (100%)	2142	96.69% (100%)
b08	814	96.44% (100%)	1046	99.81% (100%)
b09	784	99.62% (100%)	1080	98.52% (100%)
b10	858	97.20% (100%)	1122	99.91% (100%)
b11	2072	95.22% (100%)	2464	99.03% (100%)
b12	6050	98.60% (100%)	3296	95.45% (100%)
b13	1756	97.10% (100%)	2214	96.03% (100%)
b14	36376	98.32% (99.12%)	41690	99.29% (99.94%)
b15	23202	93.40% (99.35%)	29464	95.71% (99.95%)

elements due to F-scan.

Moreover, since the ATPG tool used in this experiment does not have any information about the F-scan-paths, unlike for full scan, of which the ATPG tool already knows how to handle scan chains, the highest efficiency of ATPG may not be achieved for F-scan. The two time frame model for F-scan is time consuming because the constraint in the first time frame restricts the fault simulation. However, with the results, we have proven that it is possible to do delay fault testing on F-scannable circuits that can achieve high fault coverage. In Table 4.3, the number of test patterns for both hybrid methods (full scan and F-scan) are relatively the same, except for b14 and b15 wherein hybrid F-scan generated more test patterns. For these circuits, a method to reduce the time for fault simulation is employed and in effect, redundant patterns still remain. Thus, if a more effective ATPG tool for F-scan is available, it is possible to reduce both the test generation time and number of test patterns produced for F-scan.

Table 4.3. No. of test patterns of different delay fault ATPG techniques

Ckt	**Skewed-load Full Scan**	**Broad-side Full Scan**	**Hybrid Full Scan**	**Hybrid F-Scan**
b03	61	74	87	93
b04	143	181	187	226
b05	200	219	311	210
b06	40	39	53	58
b07	117	150	204	151
b08	85	83	116	125
b09	76	71	94	95
b10	83	92	123	119
b11	149	159	216	224
b12	451	627	812	301
b13	109	127	171	151
b14	1881	1483	2457	5237
b15	1040	916	1755	3525

4.4 Improving the Hybrid Model for F-scan Delay Fault Testing

The hybrid model of F-scan delay fault testing has been proven effective to achieve high fault coverage compared to the standard techniques of skewed-load nad broad-side for gate-level full scan. A disadvantage however is that, using the two-time frame model with constraints takes too much effort and time during test generation. Hence, we improve the F-scan delay fault test generation model in this work.

Although test generation time is only a one-time cost in digital circuit testing, it becomes very crucial as the circuit becomes very large. If the ATPG method takes almost a week for a subset of an 80386 processor, it will take months or even years to finish for the complex microprocessors that we have today. Thus, it will be impossible to apply such ATPG method practically.

We propose a new scan-based delay fault test generation method applicable

to F-scannable RTL circuits using the standard gate level full scan delay fault ATPG. Previously, we used a standard combinational stuck-at ATPG method for two-time frames, wherein a constant value for the fault is assigned in the first time frame and stuck-at-fault is tested in the second time frame. As the circuit grows larger, the difficulty of fault simulation in this manner becomes harder and longer to handle. This is because the current ATPG tools do not have any information as to how to handle F-scan-paths. There may be other methods that can ease the difficulty of fault simulation for F-scan delay fault testing. However, considering the availability and popularity of the standard full scan delay fault testing approach, it is interesting to apply such ATPG method to a high-level DFT method. Using the standard full scan delay fault ATPG, which the current ATPG tools are already equipped to handle, improves test generation by magnitudes of time. This work also shows how F-scan can be practically integrated to the current digital design flow.

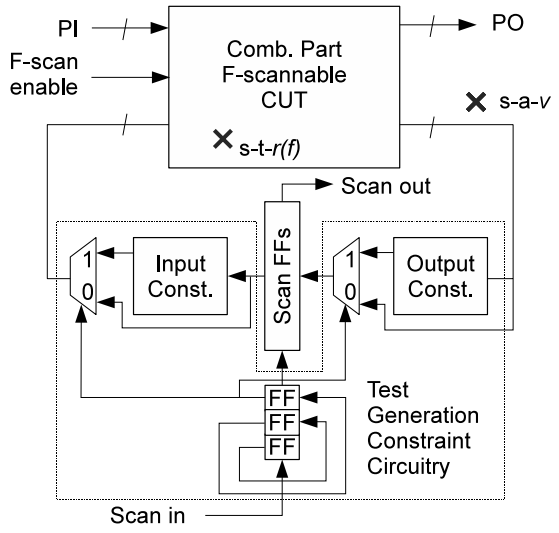


Figure 4.6. F-scan test generation model for delay fault testing using standard full scan delay fault ATPG.

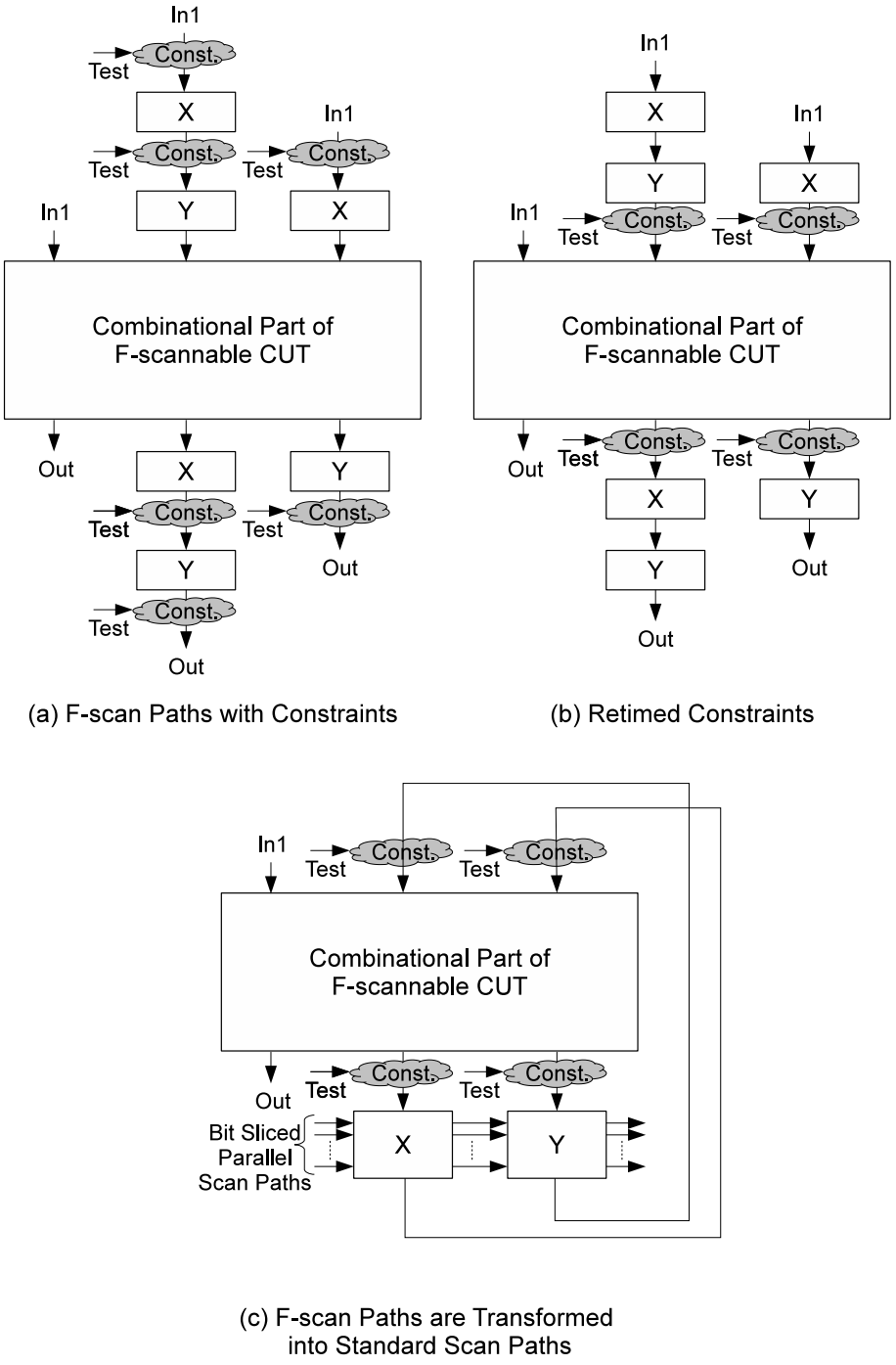


Figure 4.7. Transformation of F-scan-paths with constraints to standard scan paths.

4.5 New F-scan Test Generation Model for Delay Faults

The proposed F-scan test generation model (FTGM) for delay faults also uses the hybrid approach, which can produce both skewed-load and broad-side test patterns. However, instead of using the two-time frame combinational stuck-at ATPG of which the difficulty for fault simulation grows exponentially with circuit size, the new method uses standard full scan delay fault ATPG. With this approach, the currently available commercial tools can be maximally used to efficiently generate tests for F-scannable circuits.

As discussed previously, F-scan is applied to digital circuits at functional RTL while the ATPG is done at gate-level. Thus, F-scannable circuits are first synthesized to gate level. Designers can include the F-scan-paths in the circuit design at higher levels of abstraction, and these paths can be retained even after logic synthesis. In fact, the circuit is optimized in terms of timing, power, and size including the F-scan-paths for test during synthesis, making it possible to include the F-scan-paths during ATPG by using constraint modules discussed in Chapter 3.

In order to make the gate-level F-scannable circuits compatible with full scan delay fault ATPG, we add full scan chains after logic synthesis. Afterwards, the full scan chains are reordered according to the F-scan-paths. The full scan chains will be used during delay fault ATPG only. Application of tests will still be at RTL. The full scan chains are not needed during test application because the F-scannable circuits are already integrated with F-scan-paths to be used for scanning-in and -out test data. The only purpose of the full scan chains is to make ATPG faster, thus the flip-flops in an F-scannable circuit are not converted to scan flip-flops.

This new F-scan test generation model for delay fault testing proposed in this paper is shown in Figure 4.6. From here onwards, we call this model New FTGM, in contrast to the previous method which is called here onwards as Hybrid FTGM. The New FTGM is also hybrid but it uses a new technique to generate test patterns more effectively. The New FTGM is basically similar to the full scan test generation model for delay fault testing, however we converted the F-scan-paths with constraints to standard scan paths. Since F-scan automatically uses multiple parallel paths during test (according to bitwidth of registers), multiple full scan

chains are used during ATPG whenever the F-scan-paths have multiple bitwidths. Reordering the multiple scan chains is necessary to represent the F-scan-paths during ATPG. The scan chains are reordered based from the F-scan-paths (bit-wise), thus allowing the possible patterns to be produced compatibly with the F-scannable circuit. Moreover, we added the input and output constraints to the new test generation model. The reordering and the placement of the constraints in the New FTGM are illustrated in Figure 4.7.

Control flip-flops are added to be used as control signal to shift the circuit from test to normal mode and vice versa. We constrain the value of the three control flip-flops to "101" in the scan-in pattern to enable and disable the constraints during scan-in, normal mode, and scan-out.

Using this model, full scan delay fault ATPG can be effectively used for F-scannable circuits. The test generation time is drastically improved because the available ATPG tool can be maximally utilized. Moreover, the fault coverage achieved is also high while the test patterns generated are functionally reachable states as much as possible.

4.6 Test Generation Time Results

For our experiments with New FTGM, we used standard full scan and F- scannable versions of ITC99 RTL Benchmark Circuits, b03, b04, b06, b07, b08, b09, b10, b11 and b13. For both versions, we synthesize gate level circuits from the benchmark circuit using DesignCompiler (Synopsys). For scan path insertion, we used DesignCompiler also. F-scannable version, we apply the heuristic algorithm proposed in Chapter 2 to make the RTL benchmark circuit F-scannable.

In these experiments, we compare test quality and test generation time required for achieve the quality among full scan (Full Scan), the previous F-scan with FTGM using combinational stuck-at ATPG (Hybrid FTGM) and our proposed F-scan with FTGM using standard full scan delay fault ATPG (New FTGM). We used TetraMAX (Synopsys) as a full scan transition fault ATPG and a combinational stuck-at fault ATPG on SunFireV490 workstation with 1.8GHz UltraSPARC IV+ 4 and 32GB memory (SunMicrosystems).

Table 4.4 shows the fault coverage (FC) for these three methods. We ran TetraMax until achieving fault efficiency (FE) 100%. Under the fault efficiency,

Table 4.4. Fault Coverage Results for Hybrid Full Scan, Hybrid FTGM, and New FTGM

Ckt	**Hybrid Full Scan FC (FE)**	**Hybrid FTGM FC (FE)**	**New FTGM FC (FE)**
b03	97.77% (100%)	99.35% (100%)	99.69% (100%)
b04	94.66% (100%)	96.37% (100%)	96.21% (100%)
b06	97.51% (100%)	98.23% (100%)	98.75% (100%)
b07	97.17% (100%)	96.69% (100%)	99.30% (100%)
b08	96.44% (100%)	99.81% (100%)	99.95% (100%)
b09	99.62% (100%)	98.52% (100%)	99.79% (100%)
b10	97.20% (100%)	99.91% (100%)	100% (100%)
b11	95.22% (100%)	99.03% (100%)	99.20% (100%)
b13	97.10% (100%)	96.03% (100%)	99.09% (100%)

Table 4.5. Number of Faults Tested using Hybrid Full-Scan, Hybrid FTGM (Combinational ATPG), and New FTGM (Full Scan Delay Fault ATPG)

Ckt	**Full Scan**	**Hybrid FTGM**	**New FTGM**
b03	852	1074	978
b04	2620	3116	2954
b06	402	566	562
b07	1694	2142	2002
b08	814	1046	702
b09	784	1080	948
b10	858	1122	1086
b11	2072	2464	2378
b13	1756	2214	2080

Table 4.6. Test Generation Time Results for Hybrid Full Scan, Hybrid FTGM, and New FTGM

Ckt	**Hybrid Full Scan (s)**	**Hybrid FTGM (s)**	**New FTGM (s)**
b03	0.10	95.18	0.05
b04	155.71	2001.10	0.72
b06	0.01	45.70	0.02
b07	3.24	57.50	1.59
b08	0.06	115.30	0.12
b09	1.91	101.60	0.31
b10	0.07	140.80	0.03
b11	14.31	480.00	0.70
b13	0.30	531.00	0.31

fault coverage for these methods are all comparable.

The number of faults tested for each ATPG model is shown in Table 4.5. For hybrid full scan and new FTGM, the target faults are transition faults, while hybrid FTGM targets stuck-at faults. Full scan and F-scan induce different gate level circuits and therefore full scan and new FTGM target different numbers of faults. For hybrid FTGM, its combinational part is the same as new FTGM but the target fault models are different from each other and therefore the ATPG tool reported different values.

Time required for the test generation is shown in Table 4.6. For hybrid full scan and new FTGM, CPU times reported by TetraMAX are denoted. For hybrid FTGM, we need to call TetraMax and our batch files multiple times in order to not obtain CPU time (user time) for the method. Therefore, elapsed time is denoted for the method. Although the time includes system overhead, we can observe that this new method improves test generation time by several orders of magnitude.

From the above results, we have shown that the test generation time of the proposed test generation method is very much improved compared to that of the

previous method and it is comparable with or faster than standard full scan delay fault test generation.

4.7 Conclusion

In this chapter, we have presented a hybrid two-pattern test generation model for F-scan to be used for delay fault testing. We have already shown the effectiveness of F-scan against full scan for stuck-at faults in the previous chapters. Since the F-scan-paths, augmented at RTL, are designed such that they utilize the available functional elements and paths to connect all registers, they are usable to test any arbitrary RTL circuit with any fault model available for full scan. We have proven the effectiveness of our hybrid model used for F-scan in terms of fault coverage against the traditional delay testing techniques for full scan, skewed-load and broad-side. We have also compared our work with a similar hybrid model applied to full scan. F-scan has high fault coverage for delay faults on all benchmarks using the hybrid model. Also, the hybrid model can be configured to generate broad-side test patterns only, wherein there is higher possibility of reduction of over-testing, since the launch test patterns will come from the functional circuit only.

The hybrid model for F-scan solved the timing problems of skewed-load for full scan because F-scan can operate at operation speed during scan and normal modes. This means that there is no need to meet a strict timing requirement for the scan enable signal during skewed-load approach. Moreover, since both skewed-load and broad-side methods are integrated in one delay test technique for F-scan, the concern on shift dependency and function dependency with regards to the flexibility of the patterns produced during ATPG is given solution.

The problem of very long test generation time of the hybrid model for F-scan delay fault ATPG has also been solved using new FTGM, which uses standard full scan delay fault ATPG. Unlike the first method which uses combinational stuck-at fault ATPG for two-time frame expansion model, new FTGM is faster and does not need to undergo the tedious and time consuming fault simulation. This allows us to utilize well-developed high-performance commercial delay fault ATPG tools for delay fault testing of F-scannable circuits. The experimental results show that the test generation method improved very much compared to

that of the hybrid FTGM and it is comparable with or faster than standard full scan delay fault test generation (for circuits with gate-level full scan).

Chapter 5

Conclusion and Future Work

5.1 Thesis Summary

The aim of the work presented in this thesis is to develop methods that can make digital circuits easily testable. The primary work is aimed at the functional register-transfer level (RTL) since circuits are increasingly designed and described in this level of abstraction. We believe that it is important to deal with test problems as early as possible. This is because by considering testing at higher levels, the test problems that may occur during production, operation, and maintenance may be reduced.

This thesis has three major contributions. The first one is the fundamental concept of this work, which is the development of a design for testability (DFT) method applicable to functional RTL circuits. Our proposed technique called F-scan maximally utilizes available functional paths and elements in the circuit for test, hence reducing area overhead due to test. Test application time is also kept at the minimum. The second contribution is the constrained automatic test pattern generation (ATPG) for F-scannable circuits. This ATPG method aims to achieve high fault coverage whenever necessary. It also can reduce over-testing, especially for broad-side delay fault testing. The concept of constraint modules and the F-scan test generation model have been developed. The third contribution is the hybrid model for F-scan delay fault testing with high fault coverage. Since constrained ATPG needs to be extended for effective delay fault testing, the hybrid model has been conceptualized. A way to improve test generation time for the hybrid model is also proposed in this work.

5.2 Conclusion

In this thesis, we have developed a DFT method that can make digital circuits easily testable called F-scan and several ATPG techniques for F-scannable circuits namely, constrained ATPG and hybrid model for F-scan delay fault testing, which includes a new testing model for faster delay fault test generation. The proposed

methods have been proven to be better than gate-level full scan design.

For F-scan DFT method, the means of utilizing available functional paths and elements as much as possible make F-scan significantly better than gate-level full scan in terms of hardware overhead. The experiments on ITC'99 benchmark circuits indicate that for high-density large circuits, F-scan is increasingly advantageous compared to gate-level full scan in terms of chip area. Moreover, since the DFT augmentation is done before synthesis, performance, power, area, and timing can be all optimized including the F-scan-paths already. This results in faster test development because there is no need to re-order scan chains or change the structure of the circuit to abide with the constraints at the gate-level, which is usually a problem when doing DFT at gate-level.

Constrained ATPG proves effective in achieving high fault coverage for stuck-at faults. Also, only valid test vectors are produced using this method because of the constraints given by the F-scan-paths, hence reducing over-testing. Test application time is kept at the minimum because of the presence of multiple multi-bit F-scan-paths. Test volume is relatively similar for F-scan and gate-level full scan.

Moreover, the hybrid model for F-scan delay fault testing results in high delay fault coverage whenever needed by the design. On the other hand, if yield loss is a concern, over-testing can be reduced if the hybrid model is set to broadside mode only. Experiments on ITC'99 benchmarks are done to quantify the advantages of F-scan. Further, we reduce the test generation time by extending this hybrid model to be able to generate test patterns using the standard full scan delay fault ATPG. This is a new way of making an advanced technology compatible with the currently available test tools and equipment.

These three methods combine for a model that can easily test digital circuits with the advantages of lower area overhead and minimum test application time compared with gate-level full scan and less test generation time against the fault simulation done using the hybrid model for delay fault ATPG only.

5.3 Future Works

Possible future works regarding the F-scan DFT method and the ATPG techniques are identified. These will further improve the performance and implemen-

tation of F-scan to digital circuits, moreso for large industrial circuits.

5.3.1 Automation of F-scan DFT Augmentation Process

In Chapter 2, F-scan heuristic method for identifying F-scan-paths have been discussed. Although this step has been automated and the identification of F-scan-paths is fast, the augmentation of the digital circuit itself with the mask functions and necessary DFT elements is done manually. The development of a system to automatically augment the circuit will be helpful in applying F-scan to industry circuits. NAIST ADD 2 format has been developed in our laboratory and this can be useful in automating the DFT augmentation process.

5.3.2 Improvements on Test Generation Time

Another interesting area is the use of SAT-based techniques for constrained ATPG. In order for the ATPG tool to learn the constraints in the circuit, SAT-based techniques can be used, hence improving the test generation time and the quality of test patterns generated. This is also promising in the improvement of test generation time for F-scan delay fault testing.

In [50], an approach to SAT-based ATPG that utilizes structural information is presented. Application of this technique may not only mean an improvement in test generation time but also in the reduction of test patterns. In this work, a post-processing step is done to result in very compact test patterns, which when applied to F-scannable circuits, a huge advantage in test application time can be predicted.

5.3.3 F-scan and Power

The effects of F-scan on power consumption during test and methods to minimize it are fields of study that are not yet explored in this work but are also important parameters in improving today's testing technologies. In most cases, power consumption during test mode is higher than that during normal operation [51]. Also, the switching activity of the test patterns could affect their capability to detect faults, moreso since the circuit operates at high speed even during testing. Thus, it is necessary to determine the optimum switching activity of the

test patterns for F-scan that can reduce power consumption and error due to fast switching as much as possible. First thing to analyze is how much synthesis optimizes power in F-scannable circuits. Then, if further reduction to power consumption for F-scannable circuits is feasible, techniques on how to do so will follow.

5.3.4 F-scan at the System-Level

A system is a collection of modules, such as PCBs or printed circuit boards, which consists of collections of ICs [6]. It is necessary to apply DFT at this level in order to test parts of the circuit more effectively and less costly. A popular approach is the use of boundary scan. It has a standard called Joint Test Action Group (JTAG) 1149.1 Boundary Scan standard [52]. The use of boundary scan for F-scannable circuit modules will be discussed in this subsection.

The primary reasons for using boundary scan are to allow for efficient test of board interconnect and to facilitate isolation and testing of chips either via the test bus or by built-in self-test hardware. With boundary scan, chip-level tests can be reused at the system level [6]. In Fig. 5.1 [53], a system with boundary scan is shown. Each I/O pin is connected to an internal hardware that provides a register at that pin position. The serial connection of these registers around the periphery of the chip at the pins is known as the *boundary register*. This means that the input to the system can either come from the input pins or from loading a pattern serially into the boundary registers. Similarly, the output of the system can directly drive the output pins or it can be shifted out through the boundary registers. The *TDI* pin is the serial input to the boundary register and the *TDO* pin is the corresponding serial output.

Discussing further on the remaining parts of boundary scan hardware, *Device ID* register provides the device identification. The *bypass* register bypasses the boundary register for this component. This is useful when all boundary registers are chained into a single long shift register and it is desired to reduce the length by ignoring hardware on components that are not involved in the current test. The *instruction* register can be loaded with an instruction, which enables various different operation modes of the test hardware. The *TCK* pin provides the test clock for the boundary scan hardware, and must be capable of operating

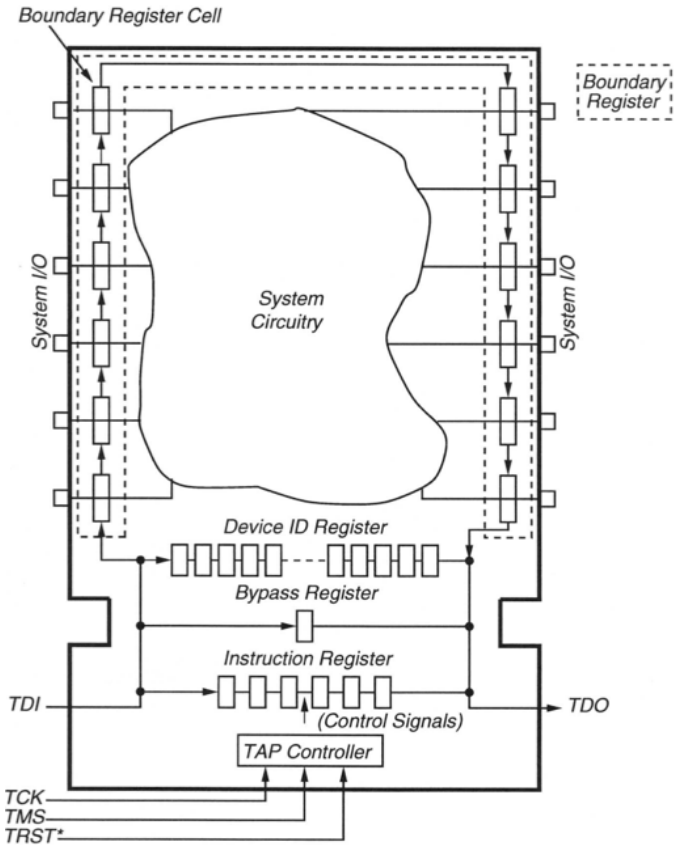


Figure 5.1. Schematic of a system with boundary scan.

at an independent clock rate from the system clock rate, asynchronously from the system circuitry. The *TMS* pin provides the *test mode select* signal, which causes the hardware to enter various testing modes. The optional *TRST** signal provides an asynchronous reset capability for the boundary scan hardware.

F-scan has been conceptualized at the chip level and it is assumed that all I/O pins of the circuit can be accessed by the test equipment at the same time. If there are multiple modules in a system, wherein each module is F-scannable, boundary scan may be applied given that the timing of shifting in values to boundary registers is faster than the F-scan test clock. This will ensure that test patterns are already shifted in to the internal I/O pins of every module. Since the internal I/O pins of each module cannot be driven directly at the same

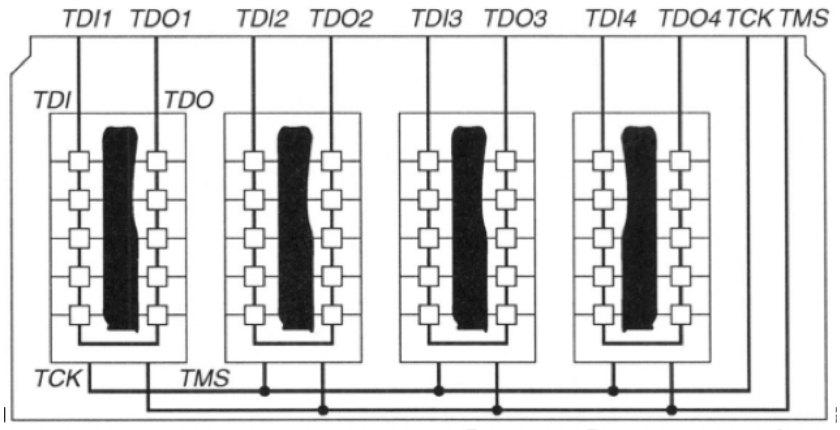


Figure 5.2. Each module with independent boundary scan chain.

time using boundary scan, it may not be possible to run test in F-scannable circuits at system speed. Thus, the traditional boundary scan approach can be applicable for stuck-at-fault testing, but further research can be done to improve the method for delay fault testing in F-scannable circuits. However, there are certain advantages for using boundary scan for stuck-at-fault testing. First, as shown in Fig. 5.2, if each module has an independent boundary scan chain, it is possible to test multiple disjoint modules simultaneously, thus reducing test time of the entire system. Secondly, since the *TDI* and *TDO* pins drive test data, if there are augmented I/O pins in individual modules, the cost will not be translated as external I/O pins. Hence, the effect of additional data pins due to F-scan on hardware overhead is minimal. However, the control pins such as hold and scan enable should be driven directly by the Tap Controller.

5.3.5 F-scan for Diagnosis

Another aspect of extension can be the applicability of F-scan for diagnosis and debug. Since full scan is effective for debug and F-scan has the same testability as full scan, the use of F-scan for debug can also be explored.

There are two approaches for fault diagnosis [6]. The first is done before testing itself. It uses fault simulation to determine the possible responses to a

given test in the presence of faults. The database constructed in this step is called *fault dictionary*. To locate faults, one tries to match the actual response obtained for the CUT with one of the precomputed responses stored in the fault dictionary. If this look-up process is successful, the dictionary indicates the corresponding fault(s). For F-scannable circuits, a similar approach can be done. However, it is necessary to consider that the creation of the fault dictionary is dependent on the F-scan-paths as well. Unlike full scan, F-paths are not necessarily I-paths, such that a value from a register propagating to another register or an output may change. Therefore, it is necessary to assume fault-free F-scan-paths during fault simulation. Moreover, it is necessary to create a model for fault simulation that includes the F-scan-paths while using available ATPG tools. Problem in fault simulation time similar to the delay fault testing method discussed in this thesis may arise.

Another approach relies on an *effect-cause analysis*, in which the effect (the actual response obtained from the CUT) is processed to determine its possible causes (faults). The use of this approach for F-scannable circuits is an exciting subject to research on.

Acknowledgements

I first came to Japan and in Nara Institute of Science and Technology in the spring of 2006 where I first experienced cutting-edge technology in VLSI design and test research in Fujiwara Laboratory. I had no solid background on digital circuit testing and I just started studying VHDL and Verilog in my previous university in the Philippines. However, in just a few days, with the help of the professors and the lab students who tutored me during my short stay, I was able to learn a lot and I was inspired to pursue this field. I went back again the next year to stay longer and work partially on my masters thesis. Not only was I able to learn the CAD tools in the lab, but I was able to feel how wonderful it was to be a student in Fujiwara lab. I then hoped that someday after I finish my masters degree, I will be able to go back to NAIST and take my PhD studies.

It's now almost five years since I first set foot in this university and it still feels like a dream. Now, this thesis, which is the fruit of two and a half years worth of work, is now in your hands. This would not have been possible without the support of the people who helped me through this endeavor. I am now taking this opportunity to extend my heartfelt thanks to all who took part in allowing me to achieve this challenging feat.

First and foremost, I am deeply indebted to my supervisor, Professor Hideo Fujiwara, from whom I learned the most during my stay here in Japan. In my early days as a PhD student, he made sure that my transition and adjustment will be smooth, especially that I am not yet very fluent in the Japanese language. In the few years that I have spent under his guidance, I was able to grow as a researcher in the field of VLSI testing and as a person. I will never forget the advices that he gave and stories that he shared. He is always very enthusiastic and motivating during our meetings, yet also critical in his attention to details, which allowed me to have deeper understanding on every aspect of our researches together. His unfaltering patience and understanding, despite the fact that I mostly miss crucial deadlines, make me go on even during the times that I'm about to give up. And also, he trusted me and gave me numerous opportunities to explore the world and present our work in conferences. Fujiwara-sensei is undoubtedly one of the best mentors I have ever had in my life, and for that I am very thankful.

I also express my gratitude to Assistant Professor Satoshi Ohtake for his hands-on guidance and support to me especially in doing experiments and understanding the results that I get. I am fortunate to be able to work with Ohtake-sensei and discuss with him about research even during those times when we were stranded in Vienna due to the Icelandic volcano eruption. His joyful and comfortable aura made me enjoy doing my work in the lab. His full support and encouragement are also reasons for further extensions of my research. For patiently explaining how experiments can be done and for proofreading and commenting on my papers even during the wee hours of the night, thank you very much.

I would like to thank the members of the thesis committee, Professor Yasuhiko Nakashima and Associate Professor Michiko Inoue, who provide me constructive comments regarding my work. Special mention to Michiko-sensei who always give critical questions, important comments, and valuable suggestions during my presentations in the laboratory. Her way of thinking is amazing and her wisdom keep on helping me improve the way I introduce my work and answer difficult questions during conferences and official presentations in the university.

I am also very thankful to Assistant Professor Tomokazu Yoneda for his valuable contributions on the improvement of my work. Discussions with Yoneda-sensei during presentations in the lab allow me to realize aspects of my work that I need to understand further and define clearly. For his help during my early days in the lab, especially for lending me a very handy laptop that I can use during presentations, thank you.

There were some foreign visitors who stayed quite long in our laboratory with whom I have discussed about research and gained friendship. I would like to especially thank Ms. Norlina Paraman, Dr. Zhiqiang You, and Dr. Chia Yee Ooi for the time we spent together in learning about our research work and Japanese culture. I felt very happy during their stay in NAIST. Dr. You also provided a very warm welcome to Changsha, China when we were there for ASICON'09, of which I am thankful.

My thanks to Dr. Hyunbean Yi as well for our long conversations about research and the future. Although we are not working on similar tracks, we enjoy discussing our work. For our stimulating discussions and for being a great

companion during laboratory activities, thank you very much.

I would also like to express my heartfelt gratitude to Ms. Aiko Sato for her tremendous help with official correspondence and various tasks in the lab. Her support during my first days in NAIST to activate gas in the dorm and her help in corresponding with others in Japanese for me all made my stay in Japan pleasant and not stressful. I will also treasure her friendship even after I finish my studies. I will always remember her concern for me and her cheerful greetings every single day, all of which I am very thankful.

The members of Fujiwara Laboratory are very warm and helpful too. I would like to thank them, especially Hiroshi Iwata, for making me feel part of the lab and for helping me with various things regarding official tasks and lab equipment use. I also thank the other past members of the laboratory namely, Dr. Fawnizu Azmadi Hussin for taking time to prepare all his documents as my reference for my application as a PhD student in NAIST and for other official transactions; and to Dr. Thomas Edison Yu whose initiatives actually allowed me to come to Japan during my first and second visit and for introducing the best laboratory that I could have been for my PhD. Without them, I could have been elsewhere and I wouldn't have had the chance to do the wonderful work that I pursue now. To all the present and past members of Fujiwara lab, thank you for your friendship and for making my stay pleasurable.

This work is partly supported by NAIST Student Support Abroad (short term) and Japan Society for the Promotion of Science (JSPS) under Grants-in-Aid for Scientific Research (B) (No. 20300018) and Young Scientists (B) (No. 22700054) under the grant for activity, education, and research. I am very thankful to all the sponsoring organizations that funded my work and my trips during conferences.

I am studying under a Japanese government scholarship, and so I am extremely grateful to the Ministry of Education, Culture, Sports, Science, and Technology (MEXT) for supporting me throughout my PhD course in NAIST. With this support, I am able to live comfortably in Japan while studying.

To all my friends here in Japan, Ate Erlyn, Igor, Marina, Henry, and the others, I thank you all for making me feel at home with your company. Our celebrations and even our downs together, these will all be kept in a special

place. I know that in you guys, I've found friends who I can call family.

Of course, I am also thankful for my forever supportive and inspiring family back in the Philippines: my father Margarito, my mother Evelyn, my brother Elvin Marlo, and my new sister Hazel Key. They always send me goodies whenever I miss food from our country. They are available to chat with whenever I need moral support. Their prayers boost me towards finishing my tasks. In times when I feel the lowest, despite the distance and seas in between, their love push me forward to be back on track again. I miss them so much and I am very thankful.

For the man who I found here in Japan, who gives me light and showers me with love every single day, who is most supportive in everything that I do... I thank you. Nico, my dear husband, our life together is just beginning and I am very grateful that God allowed us to meet, to love, and to live happily with each other's company. Throughout my work in this thesis, you have helped me in so many ways, all of which I appreciate with all my heart.

And to Allah, our creator, I believe that this work is part of your bigger plan for me in this world. I praise you for all the blessings, the good health, the wealth, and the happiness that you continuously bestow upon us.

These words are not enough, but I am really, really very thankful to you all.

-o0o-

And I don't even know what kind of things I said
My mouth kept moving and my mind went dead
Picking up those pieces now where to begin
The hardest part of ending is starting again

- Waiting for the End by Linkin Park

"But you've decided to take anything, head on, not because it's easy but because it'll make the difference."

References

[1] F. Corno, M. S. Reorda, and G. Squillero, "RT-level ITC'99 benchmarks and first ATPG results," *IEEE Design and Test of Computers*, vol. 17, pp. 44–53, Sept. 2000.

[2] D. D. Gajski, *Silicon Compilation*. Massachusetts: Addison-Wesley Longman Incorporated, 1988.

[3] D. D. Gajski, F. Vahid, S. Narayan, and J. Gong, *Specification and Design of Embedded Systems*. USA: Prentice Hall, 1994.

[4] G. D. Micheli, *Synthesis and Optimization of Digital Circuits*. McGraw-Hill Inc., 1994.

[5] U. Lauther, P. Duzy, and P. Michel, *The Synthesis Approach to Digital System Design*. Norwell, MA, USA: Kluwer Academic Publishers, 1992.

[6] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*. New York, USA: IEEE Press, 1990.

[7] A. Grochowski, D. Bhattacharya, T. R. Viswanathan, and K. Laker, "Integrated circuit testing for quality assurance in manufacturing: history, current status, and future trends," *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 44, pp. 610–633, Aug. 1997.

[8] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Boston: Springer, 2000.

[9] S. Chiusano, F. Corno, and P. Prinetto, "RT-level TPG Exploiting High-Level Synthesis Information," in *Proceedings of the 1999 IEEE VLSI Test Symposium*, pp. 341–346, 1999.

[10] S. Mourad and Y. Zorain, *Principles of Testing Electronic Systems*. New York, USA: Kluwer Academic Publishers, 2nd ed., 2000.

[11] H. Bhatnagar, *Advanced ASIC Chip Synthesis*. USA: John Wiley and Sons, Inc., 2002.

[12] J. F. Wakerly, *Digital Design: Principles and Practices*. USA: Prentice Hall, 3rd ed., 1999.

[13] J. Rajski and J. Tyszer, *Arithmetic Built-in Self-Test for Embedded Systems*. Upper Saddle River, NJ, US: Prentice-Hall, Inc, 1998.

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

[15] S. Deniziak and K. Sapiecha, "Developing a high-level fault simulation standard," *Computer*, vol. 34, pp. 89–90, May 2001.

[16] K. S. Kim, S. Mitra, and P. G. Ryan, "Delay defect characteristics and testing strategies," *IEEE Design and Test of Computers*, vol. 20, pp. 8–16, September 2003.

[17] J. Savir and S. Patil, "Scan-based transition test," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, pp. 1232–1241, Aug. 1993.

[18] H. Fujiwara, *Logic Testing and Design for Testability*. Cambridge, Massachusetts: MIT Press, 1985.

[19] S. Dey, A. Raghunathan, and R. K. Roy, "Considering testability during high-level design," in *Proceedings of the 1998 Asia and South Pacific Design Automation Conference*, pp. 205–210, Feb. 1998.

[20] Y. Huang, C.-C. Tsai, N. Mukherjee, O. Samman, D. Devries, W.-T. Cheng, and S. M. Reddy, "On RTL scan design," in *Proceedings of the 2001 IEEE International Test Conference*, pp. 728–737, 2001.

[21] R. Gupta and M. A. Breuer, "Partial scan design of register-transfer level circuits," *Journal of Electronic Testing*, vol. 7, no. 1, pp. 25–46, 1995.

[22] C.-C. Lin, M. Marek-Sadowska, M.-C. Lee, and K.-C. Chen, "Cost-free scan: a low-overhead scan path design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, pp. 852–861, Sept. 1998.

[23] S. Bhattacharya and S. Dey, "H-SCAN: A high level alternative to full-scan testing with reduced area and test application overheads," in *Proceedings of 14th VLSI Test Symposium*, pp. 74–80, May 1996.

[24] T. Asaka, S. Bhattacharya, S. Dey, and M. Yoshida, "H-SCAN+: a practical low-overhead RTL design-for-testability technique for industrial designs," in *Proceedings of the 1997 IEEE International Test Conference*, pp. 265–274, Nov. 1997.

[25] R. B. Norwood and E. J. McCluskey, "Ortohgonal scan: Low overhead scan for data paths," in *Proceedings of the 1996 IEEE International Test Conference*, pp. 659–668, 1996.

[26] C. Y. Ooi and H. Fujiwara, "A new scan design technique based on pre-synthesis thru functions," in *Proceedings of the 15th Asian Test Symposium*, pp. 163–168, 2006.

[27] K. Takabatake, T. Masuzawa, M. Inoue, and H. Fujiwara, "Non-scan design for testable data paths using thru operation," in *Proceedings of the 1997 Asia and South Pacific Design Automation Conference*, pp. 313–318, Jan. 1997.

[28] H. Wada, T. Masuzawa, K. K. Saluja, and H. Fujiwara, "Design for strong testability of rtl data paths to provide complete fault efficiency," in *Proceedings of the 13th International Conference on VLSI Design*, pp. 300–305, 2000.

[29] S. Ohtake, H. Wada, T. Masuzawa, and H. Fujiwara, "A non-scan dft method at register-transfer level to achieve complete fault efficiency," in *Proceedings of the 2000 Asia and South Pacific Design Automation Conference*, pp. 599–604, 2000.

[30] S. Ohtake, S. Nagai, H. Wada, and H. Fujiwara, "A dft method for rtl circuits to achieve complete fault efficiency based on fixed-control testability," in *Proceedings of the 2001 Asia and South Pacific Design Automation Conference*, pp. 331–334, Feb. 2001.

[31] H. Fujiwara, H. Iwata, T. Yoneda, and C. Y. Ooi, "A non-scan design-for-testability for register-transfer level circuits to guarantee linear-depth time expansion models," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, pp. 1535–1544, Sep. 2008.

[32] M. Abadir and M. Breuer, "A knowledge-based system for designing testable VLSI chips," *IEEE Design and Test*, vol. 2, pp. 56–68, July 1985.

[33] V. Chaiyakul, D. D. Gajski, and L. Ramachandran, "High-level transformations for minimizing syntactic variances," in *Proceedings of the 30th international Design Automation Conference*, pp. 413–418, 1993.

[34] I. Ghosh and M. Fujita, "Automatic test pattern generation for functional RTL circuits using assignment decision diagrams," in *Proceedings of the 37th Annual Design Automation Conference*, pp. 43–48, 2000.

[35] M. E. J. Obien and H. Fujiwara, "A DFT method for functional scan at RTL," in *Proceedings of the 10th IEEE Workshop on RTL and High Level Testing*, pp. 6–15, Nov. 2009.

[36] M. E. J. Obien, S. Ohtake, and H. Fujiwara, "Constrained ATPG for functional RTL circuits using F-scan," in *Proceedings of the 2010 International Test Conference*, no. 21.1, pp. 1–10, Nov. 2010.

[37] Y.-C. Lin, F. Lu, K. Yang, and K.-T. Cheng, "Constraint extraction for pseudo-functional scan-based delay testing," in *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, pp. 166–171, 2005.

[38] Y.-C. Lin, F. Lu, and K.-T. Cheng, "Pseudofunctional testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, pp. 1535–1546, Aug. 2006.

[39] I. Pomeranz, "On the generation of scan-based test sets with reachable states for testing under functional operation conditions," in *Proceedings of the 41st Annual Design Automation Conference*, pp. 928–933, 2004.

[40] M. Syal, K. Chandrasekar, V. Vimjam, M. S. Hsiao, Y.-S. Chang, and S. Chakravarty, "A study of implication based pseudo functional testing,"

in *Proceedings of the 2006 IEEE International Test Conference*, no. 24.3, pp. 1–10, Oct. 2006.

[41] W. Wu and M. S.Hsiao, "Mining sequential constraints for pseudo-functional testing," in *Proceedings of the 16th Asian Test Symposium*, pp. 19–24, Oct. 2007.

[42] F. Yuan and Q. Xu, "On systematic illegal state identification for pseudo-functional testing," in *Proceedings of the 46th Annual Design Automation Conference*, pp. 702–707, 2009.

[43] Z. Zhang, S. M. Reddy, and I. Pomeranz, "On generating pseudo-functional delay fault tests for scan designs," in *Proceedings of the 2005 IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems*, pp. 398–405, 2005.

[44] "International technology roadmap for semiconductors (ITRS 2007)." `http://www.itrs.net/links/2007itrs/2007_chapters/2007_Test.pdf`, 2007. Access date: October 2010.

[45] J. Savir and S. Patil, "Broad-side delay test," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 13, pp. 1057–1064, Aug. 1994.

[46] S. Wang, X. Liu, and S. T. Chakradhar, "Hybrid delay scan: A low hardware overhead scan-based delay test technique for high fault coverage and compact test sets," in *Proceedings of the Conference on Design, Automation and Test in Europe - Volume 2*, pp. 1296–1301, 2004.

[47] H. F. Ko and N. Nicolici, "Functional scan chain design at RTL for skewed-load delay fault testing," in *Proceedings of the 13th Asian Test Symposium*, pp. 454–459, 2004.

[48] H. F. Ko and N. Nicolici, "RTL scan design for skewed-load at-speed test under power constraints," in *Proceedings of the 2006 International Conference on Computer Design*, pp. 237–242, 2006.

[49] J. Saxena, K. M. Butler, J. Gatt, R. Raghuraman, S. P. Kumar, S. Basu, D. J. Campbell, and J. Berech, "Scan-based transition fault testing – implementation and low cost test challenges," in *Proceedings of the 2002 International Test Conference*, pp. 1120–1129, 2002.

[50] S. Eggersgluess and R. Drechsler, "Improving test pattern compactness in sat-based atpg," in *Proceedings of the 16th Asian Test Symposium*, vol. 0, pp. 445–452, oct. 2007.

[51] J. Saxena, K. M. Butler, and L. Whetsel, "An analysis of power reduction techniques in scan testing," in *Proceedings of the 2001 IEEE International Test Conference*, pp. 670–677, 2001.

[52] "IEEE standard test access port and boundary-scan architecture," 1994. IEEE/ANSI Standard 1149.1-1994 (revision b), includes supplements 1149.la and 1149.1b.

[53] K. P. Parker, *The Boundary-Scan Handbook*. Boston: Kluwer Academic Publishers, second ed., 1998.

Appendix

A. ITC'99 Benchmark Circuits

Table A.1. Original functions of ITC'99 benchmark circuits [1]

Circuit	**Original Function**
b01	Finite state machine (FSM) comparing serial flows
b02	FSM with binary coded decimal number recognition
b03	Resource arbiter
b04	Compute minimum and maximum
b05	Elaborate contents of a memory
b06	Interrupt handler
b07	Count points on a straight line
b08	Find inclusions in sequences of numbers
b09	Serial-to-serial converter
b10	Voting system
b11	Scramble string with variable cipher
b12	1-player game (guess a sequence)
b13	Interface to meteo sensors
b14	Viper processor (subset)
b15	80386 processor (subset)
b17	Three copies of b15
b18	Two copies of b14 and two of b17
b20	A copy of b14 and a modified version of b14
b21	Two copies of b14
b22	A copy of b14 and two modified versions of b14