ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2008, Vol. 37, No. 1

FUNCTIONAL DELAY CLOCK FAULT MODELS

Eduardas Bareiša, Vacius Jusas, Kęstutis Motiejūnas, Rimantas Šeinauskas

Software Engineering Department, Kaunas University of Technology
Studentų 50-406., LT-51368 Kaunas, Lithuania

Abstract. The test can be developed at the functional level of the circuit. Such an approach allows developing the test at the early stages of the design process in parallel with other activities of this process. The problem is to choose the right fault model because the implementation of the circuit is not available yet. The paper introduces three new fault models for synchronous sequential circuits: functional clock at-speed, functional clock static-based and functional clock delay. The introduced models are based on the primary input values, on the primary output values and on the state bits values of the programming prototype. The presented experimental results explore the possibilities of the functional test that is constructed on the base of the static-based fault model to detect the transition and stuck-at faults. The fault coverage of the functional static-based, stuck-at and transition faults corresponds with one another quite well.

1. Introduction

The new nanometer process technology increases the number of tiny transistors squeezed onto a single chip. The distance between transistors is scaling down. The adoption of nanometer processes results in the new classes of defects that affect signal timing. The defect spectrum now includes more problems such as high impedance shorts, in-line resistance, and cross-talk between signals, which are not always detected with the traditional static-based tests, known as stuck-at tests. Detecting these delay defects requires a test approach that can apply test patterns at the rated speed of the device under test.

Semiconductor companies creating these nanometer designs are seeking to create high quality, cost-effective tests for these devices. Defects per million (DPM) rates increase, unless companies add new types of tests better suited to detect new failure types. The increase of DPM rates can harm both a company's financial well-being along with their reputation. However, the leading ASIC vendors documented that their DPM rates were reduced by 30 to 70 percent by adding at-speed testing to their traditional stuck-at tests [1].

Test generation is being developed in two directions. The usual trend is when the test is generated for the circuit at the structural level. In this case, the main problem is the test generation time, because it directly influences the time-to-market. The task of test generation is quite complicated, especially for sequential circuits. Therefore, the technique of design for testability (DFT) is applied during the design of such circuits. This helps to reduce the cost of test development. But the scan design allows a synchronous sequential circuit to be brought into the states that the circuit cannot visit during functional operation. As a result, it allows the circuit to be tested using test patterns that are not applicable during functional operation. This leads to unnecessary yield loss. The other disadvantage of DFT approach is that it adds extra delay to the circuit.

The other important direction of test generation is the functional test development at the high level of abstraction. In the initial stages of the design, the structural implementation of the design is not known. Therefore the task of the test generation is more complex, because the test has to be generated for all the possible implementations. But the test development can be accomplished in parallel with other design stages. In this case, the time of test generation is not a critical issue. During design process, the software prototype of the circuit is created according to the specification. The software prototype simulates the functions of the circuit, enables to calculate the output values according to the input values. The functional test can be generated on the base of the software prototype. The test patterns generated in such a way can be used for the verification purposes as well. If the generation of the functional test encounters some difficulties, in order to facilitate the task of test generation the state variables of the software prototype can be used as the primary inputs and the primary outputs. In such a case, the generated test can be only applied for the scan designed circuit, but the correspondence between the state variables and the flip-flops of the scan register has to be established. The functional test is very valuable also for the testing of intellectual property (IP) components whose implementation

details are not known to the designer of the system on a chip.

The size of a functional test is usually much larger than that of an implementation-dependent one to assure good fault coverage for many implementations. When the synthesis of a high level description into a particular implementation is completed, the minimization of the functional test according to the particular implementation can be provided in order to exclude the test patterns that do not detect the faults of the particular implementation. Next, the list of undetected faults can be formed, and the deterministic methods can be used to detect the faults from this list. The adaptation of the functional test according to the particular implementation is much simpler task than a generation of the test from the scratch. The process of adaptation doesn't require the long hours and it has a weak impact on the overall time of the design. That is a strong advantage of the functional test. If the high level description is resynthesized, the functional test remains the same. It has to be only adapted to the new implementation.

It is worth noting that the functional test generated at a high level of abstraction can be useful even in the case where the test for the gate level structure can be easily computed or the full scan is used. These two different approaches of the test development shouldn't be opposed. The functional test sometimes is used as the basis for the gate level test generation [2]. The other advantage of the functional test over the gate level test is that the gate level test covers the functional faults only partly. Therefore, the functional test can detect the defects, which are not covered by the gate level test. The use of the functional test allows designing a partial scan only for the faults that are not detected by the functional test.

The use of the functional test to detect stuck-at faults has a long history. It was noticed that the functional test at the rated speed detects more faults, especially the faults related to a delay of the circuit. Such a test mode is called functional at-speed testing. In this approach, test engineers apply test patterns developed through functional simulation to the device's pins using a high-speed, high-pin-count tester. Later at-speed testing was applied to the scan methodology. But the functional delay fault models are defined only for the combinational circuits.

The purpose of this paper is to propose the functional delay and static-based fault models for the synchronous sequential circuits. The paper is organized as follows. We review the problems of the synchronous sequential circuits testing in Section 2. We introduce the new functional delay fault models for the synchronous sequential circuits in Section 3. We explore the possibilities of the functional test that is constructed on the base of the static-based fault model to detect the transition and stuck-at faults in Section 4. We finish with conclusions in Section 5.

2. Related work

Static-based functional fault models are introduced and analysed in the book [3]. Textual, mutation, state machine, specific, delay fault models were analysed. The input-output pin pair, input-input-output pin triplet, activating function terms fault models were introduced. The application of these models for test generation produces the test sets of different sizes. The larger set of tests means the better quality of tests. All the proposed fault models were explored and investigated experimentally. On the basis of these results, we can select an appropriate fault model according to the complexity of the problem being solved.

An iterative model of the synchronous sequential circuits imitates its functioning expanded time-wise. Having such an iterative model of device, functional static-based fault models can be used for test generation. Further these test patterns are transformed into the test sequences [3].

At-speed testing is crucial for reducing test time and in capturing frequency-dependent defect mechanisms that arise from process complexities [1]. The two at-speed scan fault models at gate level most widely used today include the path delay model and the transition delay model. Compared to static testing with the stuck-at fault model, testing logic at-speed requires a test pattern with two parts. The first part launches a logic transition value along a path, and the second part captures the response at a specified time determined by the system clock speed. If the captured response indicates that the logic involved did not transition as expected during the time unit, the path fails the test and is considered to contain a defect. While paths can start or end at the device input or output pins, testing for these types of paths requires high resolution clocking typically provided by the automatic test equipment (ATE). These capabilities are usually available only on high-end testers. Fortunately, some advanced automatic test pattern generation (ATPG) tools can utilize internal phase-locked loops (PLL) to automatically provide these high-resolution clocks [4].

Path delay tests target the path delay fault model, which models manufacturing defects or process problems that can cause cumulative delays along the design's critical paths [5]. A static timing analysis tool can identify the design's critical paths, which are used by the ATPG tool to create test patterns for faults along these paths. The total number of possible paths in a design is enormous, so the path delay fault model is generally used for only a limited number of user-defined critical paths.

To implement a path delay test, the process is as follows. A static timing verification/analysis tool generates a list of critical paths and these paths are used as input to the ATPG tool. When reading in the path file, the ATPG tool checks to make sure each path is real, meaning it is a path that can be exercised during the design's functional mode. Oftentimes, many of the

paths reported by static timing are functional false paths, or in other words, they are paths not actually used in the design's functional operation. This happens because the timing tool is only concerned with adding up the timing delays of the elements in the path. The ATPG tool analyses the paths and, when it determines that a path is a real functional path, it generates patterns to test it. When the ATPG fault model is set to path delay, the fault list contains two faults per path, a slow-to-rise and a slow-to-fall fault. It is important to note that when the ATPG tool generates path delay tests and provides a test coverage number, this number reflects coverage of only the loaded paths — not all paths in the design.

Transition tests target the transition fault model, which models manufacturing defects that behave as gross delays on gate terminals (pins). Using the transition fault model, each pin is tested for slow-to-rise or slow-to-fall transition behaviour. Two types of transition tests can be created: launch-on-shift and broadside in case of scan design [6]. In the launch-on-shift approach, the last shift of the scan chain load also serves as the transition launch event.

In the launch-on-shift approach, the scan enable signal must be able to turn off very quickly after the last shift clock and let logic settle before the capture clock occurs. For that reason, the scan enable signal usually needs to be routed as a clock signal to accomplish this. Depending on the design frequency required for the test, the scan chains themselves might also be required to shift at system frequencies.

This can be a limitation because most scan chain shifting is done at lower frequencies. If the chains are shifted and tested at-speed, this could result in an unnecessary yield loss. The main advantage of this launch-on-shift approach is that it only requires the ATPG tool to create combinational patterns, which are quicker and easier to generate.

The other transition testing technique is called broadside. In this technique, the entire scan data shifting can be done at slow speeds in test mode, and then two at-speed clocks are pulsed for launch and capture in functional mode. Once the values are captured, the data can be shifted out slowly in test mode. The main advantages of this approach are that it does not require scan chains to shift at-speed or the scan enable signal to perform as a high-speed clock. So, from the design side, it is much simpler to implement.

Additionally, broadside transition testing can also support LSSD-style scan designs. Because of these advantages, most companies have adopted this approach for creating their transition test patterns. The main disadvantage of this broadside approach is that the ATPG problem is now a sequential one, which can increase the test pattern generation time and might result in a higher pattern count.

The experiments demonstrated that launch-on-shift patterns provide higher test coverage than broadside testing [1]. However, the launch-on-shift approach provides additional detection of some non-functional faults that most companies do not want or need to test [7]. Including tests for these faults could even contribute to unnecessary yield loss.

The best results of the test coverage and the shortest size of the test can be achieved in the following way. Firstly, the path delay tests are constructed for the critical paths keeping in mind the constraints of time and the limitations of computer resources. Then, the tests for transition faults are constructed again keeping in mind the constraints of time and the limitations of computer resources. Finally, the tests for stuck-at faults that were not covered by the first two types of tests are constructed [7].

An at-speed test clock is required to deliver timing for at-speed tests [8]. There are two main sources for the at-speed test clocks. One is the external ATE and the other is a on-chip clock. Traditionally, ATE has always supplied the test clocks. In some cases, the ATE is still a viable source for the at-speed testing clocks. However, the sophistication and cost of the tester increase as the clocking speeds and accuracy requirements rise.

The second source of clocks can come from inside the chip itself. More and more designs include a PLL or other on-chip clock generating circuitry. Using these functional clocks for test purposes can provide several advantages over using the ATE [9]. In this situation, the design has both internal and external clocks and control signals. By using something called "named capture procedures," the user can specify the functionality and relationships between the internal and external signals. The ATPG tool then uses these relationships to create accurate at-speed test patterns driven by the on chip clocks.

Control registers are typically used to program which clocks to pulse and when. These internal control registers can be loaded with specific values by the ATPG tool. The ATPG user simply needs to specify the desired register values by using "condition statements" in the named capture procedures. This method is much easier than loading values through a boundary scan test access port (TAP) controller, and it does not require extra external pins to feed in those register values [10].

In fact, at-speed scan testing has replaced at-speed functional testing for the same reasons that stuck-at scan testing replaced functional testing [11]. Despite the mentioned fact, due to delay penalty of design for testability, at-speed functional testing remains still in use. The at-speed scan testing and at-speed functional testing should not be opposed because they complement each other and the both directions of the research are purposeful. Today’s technologies of manufacturing of the circuits require using various test construction methods that are combined on the base of their advantages [3]. This paper is devoted to the problems of the at-speed functional testing. The functional static-based and the functional delay fault models,

which can be applied for the scan circuits, are known only for the combinational circuits [12, 13]. The quality of the test is usually measured by the stuck-at and the transition fault coverage. The functional static-based and the functional delay fault models, which are devoted for the sequential circuits, have to be introduced in order to evaluate the quality of the test on the base of the software prototype model when the structural implementation of the circuit is not known yet.

3. Functional Clock Fault Models Based on Software Prototype

The functional delay fault models for the software prototype without state variables, which can be applied for the iterative circuit model, were analysed in [3]. The sequential circuit is replaced with the model that has the predetermined number of cycles L, when the inputs and the outputs of the circuit are repeated in every cycle. The extended model of the sequential circuit becomes the model of the combinational circuit that has L times more inputs and outputs. All the known methods of the functional test generation can be applied to such a model. This approach has the shortcoming that the large number of inputs and outputs increases significantly the number of the functional faults of the circuit.

Let's consider the functional delay fault models for the software prototype that has the state variables. The functional test consists of the test sequences. During the testing, before the launching the first test sequence the reset command, which sets the flip-flops to the known state, is issued. After loading the test stimulus, which is pulsed by the clock signal, the responses are captured at the outputs of the circuit. We will relate the functional fault models to the clock cycle and we will call them the functional clock fault models. The functional fault model relies not only on the values of the inputs and outputs but on the values of the state variables as well. According to the proposed model, the number of the functional faults is significantly less than in the case of the iterative circuit model and is independent of the number of the clock cycles L as well as the length of the test sequence.

During the functional testing, the sequence of stimuli X^t, t=1,2,…,L, is applied to the inputs of the circuit. The responses are captured at the outputs of the circuit for each time unit. If the clock signal is applied at the rated frequency, such a testing is called the functional at-speed delay testing. The applied inputs stimuli affect the states of the flip-flops. The transitions propagate along the paths of the circuit. The delays of the paths do not allow propagating the transition during one time unit. The delay fault effects are captured at the flip-flops and measured at the outputs during every clock cycle. The functional at-speed testing enables to detect the delay faults.

For simplicity, let's say that the delay defect is single and it is observed only during the one time unit. The impact of the transitions at the inputs of the circuit and in the state of the flip-flops is captured at the flip-flops and is measured at the outputs of the circuit. The values captured at the flip-flops can be observed during the same time unit or later. The value of the state of the flip-flop is observed at the outputs, if the change of this value impacts the change of the value at some outputs during the same time unit or during the next time units.

The functional faults can be detected during each time unit. The delay faults captured in the state variables are only probably detectable. Their final detection is acknowledged by the fact that the captured value in the state variable is observed at the outputs during later time units.

We will propose the functional fault models of the software prototype for the at-speed testing. Let's say, the software prototype model has *n* inputs, *m* outputs and *v* bits of state variables that can be associated with the flip-flops of the circuit. During the time unit *t*, the stimulus $<x_1^t, x_2^t,...,x_n^t>$ is applied to the inputs of the circuit and the state of the circuit $<y_1^{t-1}, y_2^{t-1},...,y_v^{t-1}>$ that was set during the previous time unit is assumed. We denote the complete input stimulus of the time unit by $P^t=<p^t_1,p^t_2, \ldots,p^t_i, \ldots, p^t_{n+v}>$, where $p^t_i:=x_i^t$, i:=1, 2,…,n, and $p^t_{n+k}:=y_k^{t-1}$, k:=1,2, …,v. Therefore, the input stimulus has *n+v* signal values. After application of input stimulus during the time unit *t*, the responses $<z_1^t,z_2^t,...,z_j^t,...,z_m^t>$ are measured at the outputs and the new values are captured in the state bits $<y_1^t, y_2^t...,y_v^t>$. The complete output pattern of the time unit is $R^t=<r^t_1,r^t_2, \ldots,r^t_j, \ldots, r^t_{n+v}>$, where $r^t_i:=z_j^t$, j:=1,2,…,m, and $r^t_{m+k}:=y_k^t$, k:=1,2, …,v. Therefore, the output pattern has *m+v* signal values. During each time unit, for every pair of inputs/outputs (i, j), four delay faults are considered: (ri_i,ri_j), (ri_i,f_j), (f_i,ri_j), (f_i,f_j) that correspond to all the possible combinations of the rising (ri) and falling (f) transition faults.

The matrix $D=||d_{a,b}||_{2(n+v),2(m+v)}$ will be used to denote the detected functional delay faults of the software prototype. The entry of the matrix is $d_{a,b}:=1$ if the corresponding functional delay fault is detected, $d_{a,b}:=0$ in the opposite case. Each input/output (i, j) is associated with four entries of the matrix $d_{2i-1,2j-1}$, $d_{2i-1,2j}$, $d_{2i,2j-1}$, $d_{2i,2j}$ that correspond to functional delay faults (ri_i, ri_j), (ri_i, f_j), (f_i, ri_j), (f_i, f_j). The entry of the matrix $d_{2i-1,2j-1}$ is set to 1 if the functional delay fault (ri_i, ri_j) is detected. That corresponds to the situation where the transition 0→1 is on the input *i*, the transition 0→1 is on the output *j*, and the blocked transition on the input disables the transition on the output. The entry of the matrix $d_{2i-1,2j}$ is set to 1 if the functional delay fault (ri_i, f_j) is detected. That corresponds to the situation where the transition 0→1 is on the input *i*, the transition 1→0 is on the output *j*, and the blocked transition on the input disables the transition on the output. The entry of the matrix $d_{2i,2j-1}$ is set to 1 if the functional delay fault (f_i, ri_j) is detected.

That corresponds to the situation where the transition 1→0 is on the input *i*, the transition 0→1 is on the output *j*, and the blocked transition on the input disables the transition on the output. The entry of the matrix $d_{2i,2j}$ is set to 1 if the functional delay fault (f_i, f_j) is detected. That corresponds to the situation where the transition 1→0 is on the input *i*, the transition 1→0 is on the output *j*, and the blocked transition on the input disables the transition on the output. We denote the second value of the transition on the input by *c*. We denote the second value of the transition on the output by *d*. Then we denote the four cases of the corresponding delay faults by the expression $d_{2i-c,2j-d}=1$. We will use this notation in the rest of the paper.

The different rules of the assessment of the value change of input stimulus P to the output pattern R allow introducing three different new fault models: the functional clock at-speed (FCaS), the functional clock static-based (FCS), and the functional clock delay (FCD). We start with the presentation of the FCaS fault model.

For the clarity of the presentation, we separate the input stimulus of the time unit *t* into two parts: the values $<x_1^t, x_2^t,...,x_n^t>$ on the primary inputs, the values $<y_1^{t-1}, y_2^{t-1},...,y_v^{t-1}>$ of state bits. We separate also the output pattern of the time unit *t* into two parts: the values $<y_1^t, y_2^t...,y_v^t>$ of state bits, the values $<z_1^t,z_2^t,...,z_j^t,...,z_m^t>$ on the primary outputs. The two cases are distinguished:

1. The impact of the signal transition is observed at the outputs during the same time unit. The entry of the matrix $d_{2i-c,2j-d}$ is set to 1 if i≤n and the change of the value on the input $x_i^t:= x_i^{t-1}$ (was $x_i^t\neq x_i^{t-1}$) causes the change of the value at the output $z_j^t:= z_j^{t-1}$ (was $z_j^t\neq z_j^{t-1}$). The entry of the matrix $d_{2i-c,2j-d}$ is set to 1 also if i>n and the change of the value of state bit $y_i^t:= y_i^{t-1}$ (was $y_i^t\neq y_i^{t-1}$) causes the change of the value at the output $z_j^t:= z_j^{t-1}$ (was $z_j^t\neq z_j^{t-1}$), where $c=x_i^t$, $d=z_j^t$, j=1,2, …,m, i:=1,2, …,n+v.
2. The impact of the signal transition is observed at the outputs during the next time units. The entry of the matrix $d_{2i-c,2j-d}$ is set to 1 if i≤n and the change of the value on the input $x_i^t:= x_i^{t-1}$ (was $x_i^t\neq x_i^{t-1}$) causes the change of the value of the output state bit $y_j^t:= y_j^{t-1}$ (was $y_j^t\neq y_j^{t-1}$), which is observed during the next time units t+1, t+2, …,L. The entry of the matrix $d_{2i-c,2j-d}$ is set to 1 also if i>n and the change of the value of state bit $y_i^t:= y_i^{t-1}$ (was $y_i^t\neq y_i^{t-1}$) causes the change of the value of the output state bit $y_j^t:= y_j^{t-1}$ (was $y_j^t\neq y_j^{t-1}$), which is observed during the next time units t+1, t+2, …,L, where $c=x_i^t$, $d=z_j^t$, j=m+1,m+2,…,m+v, i:=1,2, …,n+v.

In such a way, all the stimuli of test sequence are treated in order to fill up the matrix D, except the first one, because the first one has no stimulus before.

The FCS fault model differs from the FCaS model in two ways:

1. All the stimuli of test sequence are treated;
2. The impact of complementing the value of the input stimuli to the values of the output pattern is analysed.

The entry of the matrix D is assigned to 1 in the following two cases:

1. The impact of the signal transition is observed at the outputs during the same time unit. The entry of the matrix $d_{2i-c,2j-d}$ is set to 1 if i≤n and the change of the value on the input $x_i^t:= 1-x_i^t$ causes the change of the value at the output z_j^t. The entry of the matrix $d_{2i-c,2j-d}$ is set to 1 also if i>n and the change of the value of state bit $y_i^{t-1}:= 1-y_i^{t-1}$ causes the change of the value at the output z_j^t, where $c=x_i^t$, $d=z_j^t$, j=1,2, …,m, i:=1,2, …,n+v.
2. The impact of the signal transition is observed at the outputs during the next time units. The entry of the matrix $d_{2i-c,2j-d}$ is set to 1 if i≤n and the change of the value on the input $x_i^t:= 1-x_i^t$ causes the change of the value of the output state bit y_j^t, which is observed during the next time units t+1, t+2, …,L. The entry of the matrix $d_{2i-c,2j-d}$ is set to 1 also if i>n and the change of the value of state bit $y_i^{t-1}:=1- y_i^{t-1}$ invokes the change of the value of the output state bit y_j^t, which is observed during the next time units t+1, t+2, …,L, where $c=x_i^t$, $d=z_j^t$, j=m+1,m+2,…,m+v, i:=1,2, …,n+v.

Capturing the values at the output after each input stimuli slows down the speed of the application of test sequence and limits the size of the delay defect. It is possible to eliminate these shortcomings, if the values at the output are captured after the last stimulus of test sequence. In this case, the effect of the delay defect is captured in the state bits and it is observed at the output after the last stimulus of test sequence was processed. Such an application uses the third type of the functional clock fault model – FCD. The entry of the matrix D is assigned to 1 in the case where the impact of the signal transition is observed at the outputs after the last stimulus of test sequence was processed in the following two cases:

1) if i≤n and the change of the value on the input $x_i^t:= 1-x_i^t$ causes the change of the value at the output z_j^L,
2) if i>n and the change of the value of state bit $y_i^{t-1}:= 1-y_i^{t-1}$ causes the change of the value at the output z_j^L, where $c=x_i^t$, $d=z_j^t$, j=1,2, …,m, i:=1,2, …,n+v.

The test sequence can detect more FCaS faults than FCD faults. The sequence of length L-1 can be formed from the original test sequence of length L, if to remove the last two stimuli. The sequence of length L-2 can be formed from the test sequence of length L-1 by removing the last two stimuli. Continuing in this way it is possible to obtain L-2 test sequences, where the shortest test sequence will contain only two test stimuli. The obtained sequences will detect the same amount of the FCD faults as well as the original test sequence of length L detected FCaS faults. The

formed test sequences can be analysed in the decreasing order of their length. Such a process allows removing the test sequences that do not detect the new FCD faults. The obtained test sequences will detect the same amount of FCD faults as the original test sequence detected FCaS faults.

The total number of FCS faults is 4(n+v)(m+v), meanwhile the total number of FCaS faults is 4(n-1+v)(m-1+v), which is the same as for FCD faults. But quite a large number of FCS, FCaS and FCD faults can be undetectable. Their undectability can be verified in the following way. Let's say the stimulus $P=<p_1,p_2, \ldots,p_i, \ldots, p_{n+v}>$, where $p_i:=x_i^t$, i:=1,2,…,n, and $p_{n+k}:=y_k^{t-1}$, k:=1,2, …,v is applied to the inputs of corresponding combinational circuit, and responses are captured at the outputs of this circuit and denoted by the pattern $R=<r_1,r_2, \ldots,r_j, \ldots, r_{n+v}>$, where $r_i:=z_j^t$, j:=1,2,…,m, and $r_{m+k}:=y_k^t$, k:=1,2, …,v. The functional faults of the sequential circuit that correspond to the undetectable PP faults of the combinational circuit can not be detected as well [3]. The undetectable PP faults of the combinational circuit may be determined using the methods of random search and the simulation. But the methods of random search can not guarantee that they uncover all the testable faults. Therefore, these methods can not uncover all undetectable faults. The results of the random search methods can be trusted only with some degree of the reliability.

4. Experiments

To explore the features of the FCS fault model, we have chosen the benchmarks b01 and b02 from the benchmark suite ITC99. The random search was used to generate the test sequences consisting of seven stimuli. The values of the reset signal were generated randomly as well. For each stimulus of the test sequence, the FCS faults were determined that are detected by this stimulus according to the rules of the FCS fault model presented in the previous section. In such a way, 21 test sequences were selected for the benchmark b01, and 10 test sequences were selected for the benchmark b02. The software model of b01 has 3 inputs, 2 outputs and 3 state bits. Therefore, 120 FCS faults are counted for the benchmark b01, 92 of them can be detected without placing constraints on the state values, 28 faults are undetectable. All the detectable FCS faults were detected by the generated test sequences. The software model of b02 has 2 inputs, 1 output, and 3 state bits. Therefore, 80 FCS faults are counted for the benchmark b02, 54 of them can be detected without placing constraints on the state values, 26 faults are undetectable. 52 FCS faults were detected by the generated test sequences. The generated test sequences were treated against 3 types of the faults: FCS, stuck-at and transition. The contribution of each test sequence was evaluated separately for each type of the faults. The results are presented in Figure 1 and Table 1.

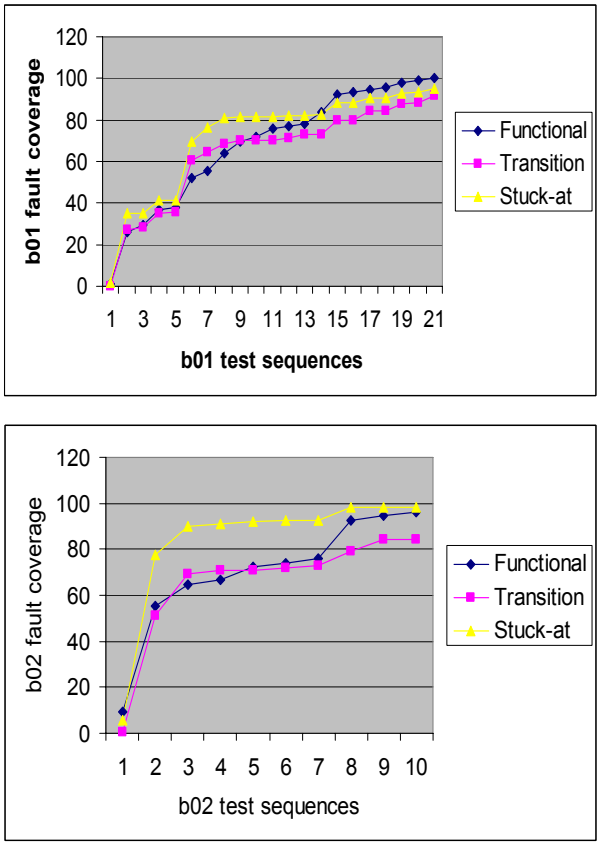

Figure 1. The process of covering faults of benchmarks b01 and b02

We can see from Figure 1 that the fault coverage for all the types of faults coincides quite well. However, if we looked carefully to Table I, we could see that the fault coverage of FCS faults increases constantly but that is not always true for the other two types of faults. One reason of this phenomenon is that the FCS faults are more general, and they are applicable to any implementation of the circuit. Therefore, the increase in the fault coverage for the stuck-at and transitions faults of the particular implementation may not be observable.

Table 2 shows the possibilities of the test to detect the other types of faults. The fault coverage for the intended type of the faults is shown in bold. The test targeted for the functional faults covers the stuck-at faults better than the transition faults. The test generation tool TetraMax was used to construct the test sequences for the stuck-at and transition faults. TetraMax was not able to achieve the complete fault coverage of the transition faults even for these small circuits. Meanwhile, the test sequences for stuck-at faults have higher fault coverage and they detect the transition and FCS faults quite well. These few experiments show that the coverage of the FCS faults coincides better with the stuck-at fault coverage than the transition fault coverage, but the functional test detects the transition faults quite well, too.

Table 1. The process of covering faults of benchmarks b01 and b02

Number of test sequenc	b01 fault coverage (%)			b02 fault coverage (%)		
	FCS	Transition	Stuckat	FCS	Transition	Stuckat
1	1,09	0,00	1,56	9,26	0,65	5,92
2	26,09	27,19	35,00	55,56	51,32	77,63
3	29,35	28,44	35,00	64,81	69,08	90,13
4	36,96	35,00	41,56	66,67	71,05	90,79
5	38,04	35,63	41,56	72,22	71,05	92,11
6	52,17	60,63	69,38	74,07	71,71	92,76
7	55,43	64,69	76,56	75,93	73,03	92,76
8	64,13	68,75	80,94	92,59	78,95	98,03
9	69,57	70,31	81,25	94,44	84,21	98,03
10	71,74	70,31	81,25	96,30	84,21	98,03
11	76,09	70,31	81,56			
12	77,17	71,19	82,19			
13	78,26	72,81	82,19			
14	83,70	73,13	82,50			
15	92,39	79,69	88,13			
16	93,48	80,00	88,13			
17	94,57	84,38	90,31			
18	95,56	84,38	90,31			
19	97,83	87,81	92,81			
20	98,91	88,13	93,12			
21	100	91,56	95,31			

Table 2. The different types of faults

Circuits	Fault coverage %		
	Functional	Transition	Stuck-at
b01	**100**	91,56	95,31
	98,91	**90,94**	97,81
	97,83	92,50	**97,81**
b02	**96,30**	84,21	98,03
	94,44	**85,53**	95,39
	88,89	83,55	**95,72**

2. Conclusion

The functional fault models allow to assess the test quality and to generate the functional test according to the software prototype in the initial stages of the design. The functional fault models were used only for the combinational circuits so far. We presented three new functional delay fault models for sequential circuits: functional clock at-speed, functional clock static-based and functional clock delay. The functional static-based, at-speed and delay faults are measured at the outputs and captured in the state bits for each time unit of the test sequence. The impact stored in the state bits of the functional static-based and at-speed faults can be observed at the outputs during the next time units. The impact of the functional delay faults is measured at the outputs after the last stimulus of the test sequence was processed. The fault coverage of the functional static-based, stuck-at and transition faults coincides quite well. The information on the test quality of the functional test and the functional test sequences can be useful in choosing the testing strategy and in achieving the high coverage of the faults at the gate level of the circuit.

References

[1] **B. Nadeau-Dostie.** Design for AT-Speed Test. *Diagnosis and Measurement*, 1999, 264.

[2] **G. Jervan, Z. Peng, O. Goloubeva, M. Sonza Reorda, M. Violante.** High-Level and Hierarchical Test Sequence Generation. *HLDVT2002: IEEE International Workshop on High Level Design Validation and Test*, 2002, 169-174.

[3] **E. Bareisa, V. Jusas, K. Motiejunas, R. Seinauskas.** Functional Digital Systems Testing. *Kaunas, Technologija*, 2006, 281.

[4] **X. Lin, J. Rajski, P. Reuter, T. Rinderknecht, B. Swanson, N. Tamarapalli.** High-Frequency, At-Speed Scan Testing. *IEEE Design & Test of Computers, September/October* 2003, *Vol*.20, *No*.5, 17-25.

[5] **L.-C. Wang, J.-J. Liou, K.-T. Cheng.** Critical path selection for delay fault testing based upon a statistical timing model. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *Vol*.23, *Issue* 11, *Nov*. 2004, 1550 – 1565.

[6] **J. Jahangiri, N. Mukherjee, W.-T. Cheng, R. Mahadevan.** Achieving High Test Quality with Reduced Pin Count Testing. *Proceedings of the* 14*th Asian Test Symposium,* 18-21 *Dec*. 2005, 312 – 317.

[7] **K.S. Kim, S. Mitra, P.G. Ryan.** Delay Default Characteristics and Testing Strategies. *IEEE Design & Test of Computers*, *September/October* 2003, *Vol*.20, *No*.5, 8-16.

[8] **S. Pateras.** Achieving At-Speed Structural Test, *IEEE Design & Test of Computers***,** September/October 2003, *Vol*.20, *No*.5, 26-33.

[9] **V. Vorisek, T. Koch, H. Fischer.** At-speed testing of SOC IC's. *Proceedings of the Design, Automation and Test in Europe Conference*, *Vol*.3, 16-20 *Feb*. 2004, 120 – 125.

[10] **B.R. Benware, R. Madge, C. Lu.** Effectiveness Comparisons of Outlier Screening Methods for Frequency Dependent Defects on Complex ASICs. *Proceedings of VLSI Test Symposium*, *April- May* 2003, 39-46.

[11] **M. Michael, S. Tragoudas.** ATPG tools for delay faults at the functional level. *ACM Transactions on Design Automation of Electronic Systems, Vol*.7 , *Issue* 1, *January* 2002, 33 - 57.

[12] **I. Pomeranz, S. M. Reddy.** Functional test generation for delay faults in combinational circuits. *International Conference on Computer-Aided Design*, 1995, 0687-0694.

[13] **E. Bareisa, V. Jusas, K. Motiejunas, R. Seinauskas.** Functional Delay Test Quality Assessment at High Level of Abstraction, *Information Technology And Control, Kaunas, Technologija*, 2007, *Vol*.36, No.1, 7 - 15.

Received January 2008.

DOI: 10.5755/j01.itc.37.1.11898