SUMMARY This paper deals with delay faults on clock lines assuming the launch-on-capture test. In this realistic fault model, the amount of delay at the FF driven by the faulty clock line is such that the scan shift operation can perform correctly even in the presence of a fault, but during the system clock operation, capturing functional value(s) at faulty FF(s), i.e. FF(s) driven by the clock with delay, is delayed and correct value(s) may not be captured. We developed a fault simulator that can handle such faults and using this simulator we investigate the relation between the duration of the delay and the difficulty of detecting clock delay faults in the launch-on-capture test. Next, we propose test generation methods for detecting clock delay faults that affect a single or two FFs. Experimental results for benchmark circuits are given in order to establish the effectiveness of the proposed methods.

key words: test generation, fault simulation, clock line, delay fault

1. Introduction

Consideration of delay faults during testing of modern high-performance LSI circuits is of paramount importance and well recognized. However, so far, generally delay faults only in the combinational part of a circuit have been targeted, and those in flip-flops (FFs) or on clock lines have been rarely targeted. This is in spite of the fact that modern designs include a large number of FFs and the probability of a fatal error caused by a clock line fault can be considerably high and is likely to increase.

Various research works have dealt with clock line faults. A test generation method for designs with on-chip clock generators was proposed [1]. This method took into consideration hardware restrictions originating from the on-chip clock generators. Similarly, a test methodology for testing control signals, including the clock logic of a microprocessor, was proposed [2]. A clock fault was defined to be a fault that can be propagated only to a clock port [2]. The problem of validating FF data hold time requirement was considered with a focus on detecting short paths and the hold time violations [3]. It was assumed that such violations are caused by clock skews and they result into erroneous state transitions.

This paper addresses detection of delay faults on clock lines under the launch-on-capture (LOC) test environment.

In LOC test environment, test vectors are applied at two different clock rates, namely scan shift and system clock rates. The clock rate in the scan shift mode is much slower than in the system clock rate. Clearly, if the amount of delay(s) on a clock line(s) is large, then the resulting fault effect(s) can be catastrophic, as a result the FF(s) driven by such a clock line will always fail to capture a value. Hence, such a fault is easy to detect. On the other hand, if the amount of delay on a clock line is small, but not too small as compared to the system clock cycle, then such a fault is likely to be difficult to detect, and therefore, it should be carefully considered. We believe, and as this paper will demonstrate, that when such type of delay occurs, the scan shift operates correctly, but capturing functional values at faulty FFs is delayed and thus the fault is activated.

To address such faults, let us first consider the clock distribution. Clock signals are distributed through a clock tree network, in general. If a physical defect exists on a fanout stem of the clock tree, then all the FFs fed by such a clock (or many FFs) are affected by the clock delay, thus such a fault seems to be easy to detect. If a physical defect exists on a fanout branch of the clock tree, then the resulting fault is likely to affect a limited number of FFs, thus making it potentially difficult to detect. The latter are the target faults for consideration in this research.

Below we summarize the characteristics of faults we consider in this research:

1. We target delay faults that occur on fanout branches in a clock tree network and affect only a very small number of FFs (one or two FFs).
2. The delay fault under consideration affects neither scan-in nor scan-out operations. Thus, we assume that scan-in vectors are always correctly set and scan-out vectors are always correctly observed even in the faulty circuit.
3. The delay fault that is targeted affects the corresponding FF only during the system clock rate testing. Which means that the FF that is driven by the faulty clock line captures a value with some delay in the system clock rate.

One of the strengths of the test generation method proposed in this paper is that it employs a standard stuck-at ATPG tool. By careful analysis we first identify the conditions for detecting a target clock delay fault, and then we convert these conditions into those for detecting a stuck-

Manuscript received November 5, 2012.
Manuscript revised January 29, 2013.

Yoshinobu HIGAMI††, Member, Hiroshi TAKAHASHI†, Senior Member, Shin-ya KOBAYASHI†, Member, and Kewal K. SALUJA†††, Nonmember

IEICE TRANS. INF. & SYST., VOL.E96–D, NO.6 JUNE 2013

Copyright © 2013 The Institute of Electronics, Information and Communication Engineers
at fault. We develop a logic model of these conditions. The netlist of the circuit is modified by adding modeling logic to the circuit during the ATPG process. This is similar to the technique introduced [4], [5] for generating diagnostic test set. This technique offers an important benefit compared with implementing a whole new test generation method without using an existing tool. Preliminary results related to this research were presented [6].

The rest of the paper is organized as follows. In Sect. 2, our fault model is described. In Sect. 3, we compare flush test and LOC test from the viewpoint of detection of clock delay faults. In Sect. 4, fault simulation results are shown and the amount of delay that should be targeted for fault detection is discussed. In Sect. 5, test generation methods for detecting delay faults on clock lines are explained, and experimental results are presented. In Sect. 6, the paper is concluded.

2. Fault Model

Clock signals are distributed by clock tree network. As mentioned before, and also shown in the results presented in the paper [7], clock delay faults on stems that affect a large number of FFs cause catastrophic failure and hence they are easy to detect. Therefore, in this paper, we only consider clock delay faults on fanout branches that affect a limited number (only one or two) of FFs.

Below, we explain our fault model and its behavior using Figs. 1 and 2. Figure 1 contains the timing information and Fig. 2 contains the high level structural information to explain the operation of an LOC test in the presence of a clock delay fault. The upper two waveforms in Fig. 1 show a fault free clock signal and the clock signal with a delay fault. We assume a typical LOC test [8], [9], where the shift clock rate is slow, and the launch and capture clock rate is fast (usually same as the system clock rate). The clock cycles in the scan-in and scan-out modes are longer than in the LOC mode. In the figure, \( t_0 \) and \( t_1 \) denote the time of the first and the second clock edge of the fault-free FFs in the LOC mode, respectively. Also \( t'_0 \) and \( t'_1 \) denote the time of arrival of the first and the second clock edge of a faulty FF in the LOC mode, respectively. The faulty clock signal is delayed by some amount relative to the fault free clock signal. It is assumed that the amount of the delay \( (t'_0 - t_0) \) and \( (t'_1 - t_1) \) is small compared with the scan clock cycle, but not so small compared with the system clock cycle. In this situation, all FFs, including the FF with faulty clock, capture the values correctly in the scan shift mode, because there is sufficient slack due to longer clock cycle (slower clock rate during scan shift mode). However, in the LOC mode with the system clock rate, a delayed capture of the scan-in value at the faulty FF can result into erroneous propagation of a value to a FF or POs. We explain this below.

Figure 2 shows a two-timeframe circuit, which represents the behavior during the LOC mode. Suppose that \( FF_1 \) is faulty, i.e. arrival of clock edge to \( FF_1 \) is delayed due to clock delay fault. Also suppose that vector \((FF_1, FF_2, FF_3) = (0, 0, 0)\) is shifted in during the scan shift operation, and logic of the circuit is such that \((FF_1, FF_2, FF_3) = (1, 1, 1)\) is captured at \( t_0 \) in the fault-free circuit during LOC conditions.

Lower waveforms of Fig. 1 show the signals at \( FF_1 \) and \( FF_2 \) in the fault-free and faulty circuits, respectively. In the faulty circuit, \( FF_1 \) captures 1 at \( t'_0 \), which is delayed. As a result scan-in value 0 remains at \( FF_1 \) before \( t'_0 \), and this value is different from the correct functional value 1. Due to this delayed capture at \( FF_1 \), \( FF_2 \) captures 1 at \( t_1 \) in the faulty circuit, while it captures 0 in the fault-free circuit. Note that \( FF_2 \) captures a functional value at \( t_1 \) in the fault-free and the faulty circuit, because its clock signal is not delayed. When the amount of delay \( (t'_0 - t_0) \) and \( (t'_1 - t_1) \) is not so small compared with the system clock cycle, \( FF_1 \) keeps the incorrect value for some time and it is propagated to \( FF_2 \) by \( t_1 \).

The clock delay fault model considered in this paper assumes that the amount of delay is not very small compared with the system clock cycle but it is much smaller than the scan clock cycle. Thus we assume that the fault effect can be propagated to some FF or FFs, except for the faulty FF itself. As Fig. 2 shows, in order to produce an erroneous value, a scan-in value and a functional value need to be different for the faulty FF. Also such an erroneous value needs to be propagated to a FF.
3. Flush Test versus LOC Test

In this section, we discuss the difference between the flush test and the LOC test. Flush test is a testing method where test data are applied at scan inputs and they are shifted along scan chains. FFs are in the scan shift mode, and they capture only scan shift values. The contents of FFs are shifted out from the scan outputs. It is known that the flush test is effective for detecting defects on FFs and scan chains [10].

For the sake of simplifying the discussion, we consider the error propagation between two FFs, where the clock signal on the source FF is fault-free. The following variables, as shown Fig. 3, are used in the discussion that follows:

- $t_s$: clock cycle of the flush test
- $t_c$: clock cycle of the LOC test
- $d$: delay of a faulty clock line
- $p_s$: propagation delay between the two FFs on the scan chain
- $p_c$: propagation delay along a sensitized path between the two FFs

The condition on the duration of the clock delay fault for which an erroneous value is captured at the destination FF is described below. In order to detect a delay fault, the amount of delay $d$ should satisfy the condition (1) for the flush test and condition (2) for the LOC test.

\[
\begin{align*}
\text{Flush test:} & \quad d > t_s - p_s \\
\text{LOC test:} & \quad d > t_c - p_c
\end{align*}
\]

Since there is no gate between two FFs on the scan chain, typically $p_s$ is smaller than $p_c$. Therefore the amount of delay $d$ of the flush test is larger than that of the LOC test. If we would like flush test to detect a delay fault of duration $d$ of the LOC test, then $t_s$ needs to be smaller than $t_c$, but that would be difficult or impossible, because $t_s$ is the system clock cycle. As a result, we can safely state that LOC test can detect smaller amount of clock delay defects than a flush test. If we apply the flush test with fast scan shift clock (same as or faster than the system clock), then small amount of clock delay defects that the proposed method aims to detect can be detected. However, in order to supply such a fast scan shift clock in high-speed LSIs, the cost and difficulties for designing the scan clock mechanism will be high. This is a disadvantage of the flush test with fast scan shift clock.

4. Fault Simulation

In this section, we show fault simulation results in order to discuss the relation between the amount of delay and the number of detected faults. The amount of delay assumed in this paper is sufficiently small compared to the scan clock cycle, but not so small compared to the system clock cycle. If it is much smaller than the system clock cycle, then the clock delays are neither activated nor detected for most of the FFs. Therefore, we perform fault simulation with varying the amount of delay.

We first explain the characteristics of the fault simulator. The fault simulator used here has been implemented using the technique introduced in the paper [11], where 7 logic values (0, 1, $R$, $F$, $B$, $\bar{B}$, $X$) were used instead of $R$ and $F$.) The values 0, 1, $X$, $R$ and $F$ denote stable 0, stable 1, unknown value, rising transition 0→1 and falling transition 1→0, respectively. The values $B$ and $\bar{B}$ denote transition 0→1 and 1→0 that are propagated from a fault site, respectively. Tables 1 and 2 show AND operation and OR operation using the 7 logic values, respectively. For computation of transition times, following notation is used. Without loss of generality we assume that $d_1$ is later than $d_2$, where $d_1$ and $d_2$ denote the latest transition time of input $x_1$ and $x_2$, respectively. The latest transition time of the output of an AND gate and OR gate is calculated as shown in Tables 3 and 4, respectively. Figure 4 shows an example of the AND calculation by waveforms, where inputs $x_1$ and $x_2$ have $R$ with the latest transition time $d_1$ and $d_2$, respectively, and the output results in a $R$ with the latest transition time $d_1 + 1$, assuming the gate delay to be 1 unit. Using this 7 logic values simulation method with latest transition time calculation, we can investigate the delay of a clock signal that is detected by a test.

Before investigating the effect of clock delay faults in

![Fig. 3](image-url)
Table 3  Latest transition time on the output of an AND gate.

| x1 \ x2 | 0   | 1   | R   | F   | B   | \bar{B} | X   |
|---------|-----|-----|-----|-----|-----|---------|-----|
| 0       | 0   | 0   | 0   | 0   | 0   | 0       | 0   |
| 1       | 0   | 0   | d2 +1 | d2 +1 | d2 +1 | d2 +1 | 0   |
| R       | 0   | d1 +1 | d1 +1 | 0   | d1 +1 | 0   | 0   |
| F       | 0   | d1 +1 | d1 +1 | d2 +1 | d1 +1 | d2 +1 | 0   |
| B       | 0   | d1 +1 | d1 +1 | 0   | d1 +1 | 0   | 0   |
| \bar{B} | 0   | d1 +1 | d1 +1 | d2 +1 | d1 +1 | 0   | 0   |
| X       | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

Table 4  Latest transition time on the output of an OR gate.

| x1 \ x2 | 0   | 1   | R   | F   | B   | \bar{B} | X   |
|---------|-----|-----|-----|-----|-----|---------|-----|
| 0       | 0   | 0   | d2 +1 | d2 +1 | d2 +1 | d2 +1 | 0   |
| 1       | 0   | 0   | 0   | 0   | 0   | 0       | 0   |
| R       | d1 +1 | 0   | d2 +1 | d1 +1 | d2 +1 | d1 +1 | 0   |
| F       | d1 +1 | 0   | 0   | d1 +1 | 0   | d1 +1 | 0   |
| B       | d1 +1 | 0   | d2 +1 | d1 +1 | d2 +1 | d1 +1 | 0   |
| \bar{B} | d1 +1 | 0   | 0   | d1 +1 | 0   | d1 +1 | 0   |
| X       | 0   | 0   | 0   | 0   | 0   | 0       | 0   |

Fig. 4  Example of waveforms for AND operation.

Table 5  Test set and candidate faults.

| circuit   | target | size | trans_det |
|-----------|--------|------|-----------|
| s9234     | 224    | 194  | 146       |
| s13207    | 663    | 127  | 481       |
| s15850    | 588    | 144  | 429       |
| s35932    | 1728   | 54   | 1728      |
| s38417    | 1627   | 139  | 1466      |
| s38584    | 1451   | 258  | 1388      |

For all the benchmark circuits listed in Table 5 the simulation results for “Minimum delay” and “Maximum delay” extend the simulation for considering a real delay situation, where each gate and signal line has various delay amounts. However, we do not have any data of the real delay amounts for the benchmark circuits, and thus we used the unit delay model. Test vectors obtained for detecting gate transition faults as mentioned above were used for this simulation.

Table 6 shows the results for a specific FF, FF-148, in s9234 benchmark circuit. The system clock cycle was set to 59 based on the length of the critical path in this circuit. We noticed that for a very large clock delay, nearly the duration of clock cycle, for this FF a total of 60 vectors among the all 194 applied test vectors, detected this fault. Next we varied the amount of delay on the clock signal, in steps of 1 unit, (corresponding to \( t_0 - t_0 \) in Fig. 1) to determine the number of test vectors that detected the fault. For example, when the amount of delay of the faulty clock signal was 20, only two test vectors detected the fault among the 60 original detecting test vectors. The results in Table 6 imply that the fault on FF-148 is never detected if the delay is less than 20 units on the clock line. On the other hand when the amount of delay is 45 or more, this fault was detected by all 60 detecting test vectors.

Table 7 shows the results for minimum delay, maximum delay and median delay for all the candidate faults in s9234. The candidate faults include clock lines for each FF one at a time. “Minimum delay” means the amount of delay such that a candidate fault is detected by at least one test vector. For example, in the results shown in Table 6, the minimum delay was 20 units. In Table 7, there are 9 faults whose minimum delay is 1 to 5 units. “Maximum delay” means the amount of delay such that a candidate fault is detected by all the detecting test vectors. For example, in the results shown in Table 6, the maximum delay was 45 units. In Table 7, there is one fault whose maximum delay is between 21 to 25 units. “Median delay” means that the amount of delay such that a candidate fault is detected by half of all the detecting test vectors. For example, in the results shown in Table 6, the median delay was 42 units.

Tables 8, 9 and 10 show the simulation results for three other benchmark circuits, namely, s13207, s15850 and s38584, respectively. For these circuits the sizes of transition test sets used consisted of 127, 144 and 258 vectors, respectively as shown in Table 5. From these tables we observe that for the circuit s38584, about 60% of all the faults detectable by the transition test set are detected when the clock signal is delayed by about 40 units or more, which is about two thirds of the system clock cycle. This also suggests that 30% of the clock delay faults will not be detected by the transition tests if the clock delay was smaller than 36 units.

a circuit, we provide the information about the test set and candidate faults. Table 5 lists the ISCAS’89 benchmark circuits which are candidates for investigation. Candidate fault set for each circuit consists of those in which only one FF is affected and their number is listed under “target”. We use the test set that detects transition faults in the gates in the circuit and the size of the test set (number of vectors in the test set) is listed under “size”. These test vectors were generated by using a method similar to the one proposed in the paper [12]. The “trans_det” lists the number of faults detected by the transition test set provided the clock signal to each FF was delayed by one full clock cycle. Clearly these are the candidate faults which need to be considered for fault detection by the above method. In the rest of this section the candidate fault set will consist of the faults that appear under the column “trans_det”. The goal here is to show that many of these faults may not be detected if the clock delay was shorter than a full clock period.

Now we show results for a number of ISCAS’89 benchmark circuits. We assume each gate has unit delay and signal lines are ideal with no delay. Note that we can easily
Table 6  The number of patterns detecting clock delay on FF-148 in s9234.

| delay vectors | 20 | 21 | 22 | 24 | 25 | 39 | 40 | 41 | 42 | 43 | 44 | 45 |
|---------------|----|----|----|----|----|----|----|----|----|----|----|----|
|               | 2  | 7  | 8  | 10 | 13 | 20 | 25 | 29 | 42 | 49 | 53 | 60 |

Table 7  The relation between the delay and the number of detected faults in s9234.

| delay | 1-5 | 6-10 | 11-15 | 16-20 | 21-25 | 26-30 | 31-35 | 36-40 | 41-45 | 46-50 | 51-55 | 56-60 |
|-------|-----|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| min   | 9   | 3    | 12    | 16    | 11    | 14    | 18    | 21    | 4     | 13    | 19    | 6     |
| max   | 0   | 0    | 0     | 0     | 1     | 8     | 2     | 18    | 42    | 34    | 33    | 8     |
| median| 0   | 0    | 0     | 0     | 1     | 16    | 30    | 36    | 42    | 11    | 7     | 3     |

Table 8  The relation between the delay and the number of detected faults in s13207.

| delay | 1-5 | 6-10 | 11-15 | 16-20 | 21-25 | 26-30 | 31-35 | 36-40 | 41-45 | 46-50 | 51-55 | 56-60 |
|-------|-----|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| min   | 9   | 8    | 1     | 23    | 16    | 3     | 1     | 28    | 27    | 134   | 174   | 57    |
| max   | 0   | 0    | 0     | 3     | 12    | 5     | 4     | 23    | 4     | 132   | 209   | 88    |
| median| 0   | 0    | 0     | 9     | 15    | 13    | 7     | 8     | 18    | 20    | 145   | 188   | 57    |

Table 9  The relation between the delay and the number of detected faults in s15850.

| delay | 16-20 | 21-25 | 26-30 | 36-40 | 41-45 | 46-50 | 51-55 | 56-60 | 61-65 | 66-70 | 71-75 | 76-80 | 81-85 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| min   | 2     | 1     | 1     | 3     | 44    | 31    | 41    | 53    | 73    | 73    | 49    | 45    | 14    |
| max   | 2     | 1     | 0     | 0     | 21    | 47    | 3     | 21    | 34    | 33    | 38    | 128   | 56    |
| median| 2     | 1     | 1     | 1     | 35    | 32    | 16    | 42    | 59    | 91    | 91    | 45    | 14    |

Table 10 The relation between the delay and the number of detected faults in s38584.

| delay | 1-5 | 6-10 | 11-15 | 16-20 | 21-25 | 26-30 | 31-35 | 36-40 | 41-45 | 46-50 | 51-55 | 56-60 |
|-------|-----|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| min   | 8    | 6    | 11    | 16    | 16    | 20    | 67    | 110   | 247   | 324   | 560   | 2     |
| max   | 0    | 0    | 4     | 5     | 4     | 0     | 4     | 23    | 29    | 30    | 1286  | 2     |
| median| 0    | 5    | 11    | 14    | 10    | 10    | 27    | 53    | 147   | 418   | 690   | 2     |

Fig. 5  Results on minimum delays.

Fig. 6  Results on maximum delays.

are shown in Figs. 5 and 6, respectively. In these figures X-axis denotes the percentage of the amount of additional delay in the presence of fault relative to the cycle time of the system clock. The Y-axis denotes the cumulative percentage of the number of detected faults. Note that Y-axis is normalized with respect to the total number of candidate faults shown in Table 5 under the column “trans_det” for each benchmark circuit. We make the following observations from Fig. 5. The curve labeled by s9234 traces the point of X = 55% and Y = 51%. According to the definition of Minimum delay, for the circuit s9234, 51% of the detected faults were detected by at least one test vector, when the amount of delay is 55% of the clock cycle. Similarly, for the circuit s38584, 52% of the detected faults were detected by at least one test vector, when the amount of delay is 85% of the clock cycle. From Fig. 6 we observe that for the circuit s9234, 54% of the detected faults were detected by all the detecting test vectors, when the amount of delay is
77% of the clock cycle and for the circuit s38584, only 10% of the detected faults were detected by all the detecting test vectors, when the amount of the delay is 94% of the clock cycle.

These results quantify the amount of delay that can be tested by our simulation methodology. In the test generation methods, described in the next section, the length of the fault propagation path is not considered. Instead, when a fault effect is propagated to a FF or a PO through a path (long, short or any path), the fault is regarded as detected. However, based on the results presented in this section, we can say that when the amount of delay is about 50 to 80% of the system clock cycle or longer, over half of the faults can be detected. Although, we should note that even smaller size of delay can be tested depending on fault site or test clock cycle.

5. Test Generation for Clock Delay Faults

In this section, we explain test generation methods for clock delay faults for two cases as follows:

1. The case of a single FF affected by a clock delay
2. The case of two FFs affected by a clock delay

In the test generation methods, we only consider logic behavior but not timing behavior with a specific delay duration. Hence, generated test vectors may not detect target faults, if the amount of the delay is too small. It is noted that small delay is out of range of the targets by the proposed methods. (In this context, ‘small delay’ means delay which is too small compared to the system clock cycle.)

5.1 A Single FF Affected by a Clock Delay

This subsection deals with test generation in the case of a single FF affected by a clock delay. In order to make a faulty value at a target FF different from the fault-free value, a test vector must be applied such that the scan-in value is different from the functional value. If a clock delay fault occurs, the FF fails to capture the correct value at the correct timing in the LOC mode.

The conditions for detecting a clock delay that affects a single FF are described below.

[Condition 1]
- The scan-in value must be different from the functional value captured at the faulty FF in the first timeframe.
- There exists at least one sensitized path from the faulty FF to another FF or a PO between the first and the second clock in the LOC mode.

The proposed test generation method employs a standard stuck-at ATPG tool. The conditions for detecting a clock delay are converted into those for detecting a stuck-at fault by adding some logic. A two-timeframe circuit, as shown in Fig. 7, is used in order to generate test vectors under the assumption of the LOC test. A scan-in vector is applied in the first timeframe, and functional values are captured and launched in the second timeframe. The functional values captured at the end of the second timeframe are scanned out.

Now consider that a target fault is a clock delay fault that affects only FF1 in Fig. 7. The proposed method adds a multiplexer as shown in Fig. 8, and targets a stuck-at 0 fault on SEL in the ATPG process. In the second timeframe, a functional value resulting from the first timeframe is launched in the fault free circuit while a scan-in value is launched in the faulty circuit. Therefore, if a test vector is successfully generated for detecting a stuck-at 0 fault on SEL, then it must satisfy Condition 1. This modification of a circuit is only during the ATPG process, and thus it is quite different from a DFT technique. Note that FF1 in the second timeframe is masked. This is because FF1 always captures a correct value at the second clock pulse, as discussed in Sect. 2 in the LOC mode, though the capturing is delayed.

5.2 Two FFs Affected by a Clock Delay

This subsection deals with test generation assuming that a clock delay fault affects exactly two FFs. Two FFs affected by a clock delay fault fail to capture functional values at the correct timing. The conditions for detecting a clock delay fault that affects two FFs are described below.

[Condition 2]
- The scan-in value(s) must be different from the func-
There exists at least one sensitized path from either one or both of the faulty FF to another FF or a PO between the first and the second clock in LOC mode.

Similar to the case explained in the previous section, the test generation method adds some logic to a circuit and employs a stuck-at ATPG tool. Working with the same figure as in the previous section, i.e. Fig. 7, consider that \( FF_1 \) and \( FF_2 \) are affected by a clock delay. In this case, two multiplexers are added as shown in Fig. 9, and a stuck-at 0 on \( SEL \) is targeted for ATPG. In a fault free circuit, the functional values captured at \( FF_1 \) and \( FF_2 \) are launched in the second timeframe, while in the faulty circuit the scan-in values are launched in the second timeframe. Note that \( FF_1 \) and \( FF_2 \) are masked at the end of the second timeframe. If fault effect(s) on \( FF_1 \) or \( FF_2 \) are propagated to \( FF_3 \) then the test generation is successful.

### 5.3 Multiple FFs Affected by a Clock Delay

Although we explained the cases when only one and two FFs are affected by the fault, the problem when more than two FFs are affected by the fault can be solved by similar approach. The method and experimental results related to dealing with faults that affect many FFs are presented [7]. It is shown that the clock delay fault affecting a large number of FFs can be easily detected, and that most such faults can be detected by gate-transition tests [7].

### 5.4 Experimental Results

The following experimental setup was used to carry out test generation for a number of ISCAS‘89 benchmark circuits. We used a computer with 2.6 GHz CPU and 3 GB Memory under Linux OS Environment. Initially, we applied test vectors generated for gate transition faults in a combinational part of the circuit using a method similar to the one proposed in the previous paper [12]. The clock delay faults undetected by the transition test vectors were collected, and they were targeted for generating test vectors using the method proposed in this paper. A stuck-at ATPG tool reported in the previous paper [13] was used. During test generation, faults on those clock lines which drive FFs that feedback to themselves (form a self-loop), are excluded from the target fault list.

Tables 11 and 12 show the results for the case of a single FF affected by a clock delay, as described in Sect. 5.1. Table 11 shows, from left to right, the names of the circuits, the number of target faults, the number of detected faults by the transition test vectors, the number of faults detected by test vectors generated by the proposed method, the number of redundant faults identified by the proposed method, and the fault efficiency of the final test set. It is instructive to note that our method achieved 100% fault efficiency for every circuit, and we also note that for s35932 the initial transition test vectors detected all target faults. Table 12 gives the information about the test generation effort. It shows the names of the circuits, the number of initial test vectors (test vectors for gate transition faults), the number of generated test vectors, and the CPU time in seconds. It is evident that our in-house ATPG tool did quite well. However, clearly CPU time greatly depends on the performance of a stuck-at ATPG tool used as an engine for this method. As a result, our method has potential of driving all the benefits of the existing or newly developed high performance ATPG tools and provide even better performance.

Next we generated tests for the case of two FFs affected by a clock delay fault as described in Sect. 5.2. Similarly to the previous experiment, test vectors for gate transition faults were first used and undetected clock delay faults by the transition test set were collected. These undetected faults were then used as target faults by the proposed method. As initial targets, 1000 pairs of FFs were randomly selected in the absence of any layout information. In practical application of the method, target fault pairs should be selected from the layout information. FFs fed by a fanout branch in a clock tree should be selected as faulty FFs. Tables 13 and 14 show the results, and the meaning of each column heading is same as for Tables 11 and 12, respectively. Once again 100% fault efficiency was achieved for every circuit. CPU
Table 13  Results for the case of two affected FFs.

| circuit | target | trans_det | tpg_det | tpg_red | eff(%) |
|---------|--------|-----------|---------|---------|--------|
| s9234   | 1000   | 865       | 55      | 80      | 100    |
| s13207  | 1000   | 922       | 45      | 33      | 100    |
| s15850  | 1000   | 934       | 32      | 34      | 100    |
| s35932  | 1000   | 1000      | 0       | 0       | 100    |
| s38417  | 1000   | 993       | 7       | 0       | 100    |
| s38584  | 1000   | 999       | 0       | 1       | 100    |

Table 14  Test vectors and CPU time for two affected FFs.

| circuit | trans_vec | tpg_vec | CPU (s) |
|---------|-----------|---------|---------|
| s9234   | 194       | 16      | 434     |
| s13207  | 127       | 21      | 503     |
| s15850  | 144       | 20      | 1258    |
| s35932  | 54        | 0       | 0.25    |
| s38417  | 139       | 7       | 91.1    |
| s38584  | 258       | 0       | 2.4     |

time depends on the number of faults that are targeted for ATPG. In particular, when the number of redundant faults was large, long CPU time was spent.

6. Conclusions

In this paper, we considered the problems of delay faults on clock lines. The amount of the delay assumed in our model is much smaller than the scan shift clock cycle. Such a small delay may not affect the circuit during the scan shift mode, but it can potentially affect the circuit operation during the system mode.

We first showed fault simulation results in order to investigate the relation between the amount of delay and the number of detected faults. Next we proposed test generation method for detecting the clock delay faults. The proposed method uses a standard stuck-at fault test generator. We showed that by adding some extra logic gates for the purpose of modeling a fault, the conditions for detecting a clock delay fault are converted into those for detecting a stuck-at fault.

In our future work, we will develop diagnosis methods for clock delay faults. We will also like to address various other practical situations for such faults and improve the method. For example, we will consider the situation in which multiple clock domains are used, or FFs are driven by the clock signals with different clock skews.

Acknowledgment

The authors would like to thank Prof. Kajihara who provided us the test generator used in the experiments.

References

[1] S. Lin and M. Kassab, “Test generation for designs with on-chip clock generators,” Proc. Asian Test Symp., pp.411–417, 2009.
[2] T.L. McLaurin, R. Slobodnik, K.H. Tsai, and A. Keim, “Enhanced testing of clock faults,” Proc. Int. Test Conf., Paper 31.2, 2007.
[3] S.M. Reddy, I. Pomeranz, S. Kajihara, A. Murakami, S. Takeoka, and M. Ohta, “On validating data hold times for flip-flops in sequential circuits,” Proc. Int. Test Conf., pp.317–325, 2000.
[4] D.H. Baik, Y.C. Kim, K.K. Saluja, and V.D. Agrawal, “Exclusive test and its applications to fault diagnosis,” Proc. Int. Conf. on VLSI Design, pp.143–148, Jan. 2003.
[5] Y. Higami, Y. Kurose, S. Ohno, H. Yamaoka, H. Takahashi, Y. Shimizu, T. Aikyo, and Y. Takamatsu, “Diagnostic test generation for transition faults using a stuck-at ATPG tool,” Proc. Int. Test Conf., Paper 16.3, 2009.
[6] Y. Higami, H. Takahashi, S. Kobayashi, and K.K. Saluja, “Fault simulation and test generation for clock delay faults,” Proc. Asia and South Pacific Design Automation Conf., pp.799–805, 2011.
[7] Y. Higami, H. Takahashi, S. Kobayashi, and K.K. Saluja, “On detecting transition faults in the presence of clock delay faults,” Proc. Asian Test Symp., pp.1–6, 2011.
[8] W. Qiu, J. Wang, D.M.H. Walker, D. Reddy, X. Lu, Z. Li, W. Shi, and H. Balachandran, “K Longest Paths Per Gate (KLPG) test generation for scan-based sequential circuits,” Proc. Int. Test Conf., pp.223–231, 2004.
[9] G. Xu and A.D. Singh, “Delay test scan flip-flop: DFT for high coverage delay testing,” Proc. Int. Conf. on VLSI Design, pp.763–768, 2007.
[10] F. Yang, S. Chakravarty, N.D.-Prasanna, S.M. Reddy, and I. Pomeranz, “Improving the detectability of resistive open faults in scan cells,” Proc. Int. Symp. on Defect and Fault Tolerance in VLSI Systems, pp.383–391, 2009.
[11] H. Takahashi, T. Watanabe, and Y. Takamatsu, “Generation of tenuous tests for small gate delay faults in combinational circuits,” Proc. Asian Test Symp., pp.332–338, 1995.
[12] Y. Higami, K.K. Saluja, H. Takahashi, S. Kobayashi, and Y. Takamatsu, “Increasing defect coverage by generating test vectors for stuck-open faults,” Proc. Asian Test Symp., pp.97–102, 2008.
[13] S. Kajihara, I. Pomeranz, K. Kinoshita, and S.M. Reddy, “Cost effective generation of minimal test sets for stuck-at faults in combinational logic circuits,” IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.14, no.12, pp.1496–1504, 1995.

Yoshinobu Higami  received his B.E., M.E., and D.E. degrees from Osaka University in 1991, 1993 and 1996, respectively. After serving as a research fellow of the Japan Society for the Promotion of Science, he joined Department of Computer Science, Ehime University in 1998. Currently he is an associate professor at Graduate School of Science and Engineering, Ehime University. In 1998 and 2006, he was also an honorary fellow at University of Wisconsin-Madison, U.S.A. He received the IEICE Best Paper Award in 2005 and 2012. His research interests include test generation, design for testability and fault diagnosis of logic circuits. He is a senior member of the IEEE and a member of IPSJ.
Hiroshi Takahashi received the B.E. and M.E. degrees in electronic engineering from Saga University, Saga, Japan in 1988 and 1990, respectively. He received the Dr. degree from Ehime University, Japan in 1996. From 1997 to 1999 he was a Lecturer in the Department of Computer Science, Ehime University. Since 2000, he has been an associate professor at Ehime University. From May 2000 to March 2001, he was a research fellow in University of Wisconsin-Madison, USA. His research interests are test generation and fault diagnosis for digital systems. Dr. Takahashi is a senior member of the IEEE and a member of IPS Japan. He served as the Program committee member of 2003, 2004, 2005, 2007 IEEE Asian Test Symposium, as the Program Vice-Chair of the 2008 IEEE Asian Test Symposium and as the Program Chair of the 2012 IEEE Asian Test Symposium.

Shin-ya Kobayashi received the B.E. degree, M.E. degree, and Dr.E. degree in Communication Engineering from Osaka University in 1985, 1988, and 1991 respectively. He is a Professor at Graduate School of Science and Engineering, Ehime University. His research interests include distributed processing, and parallel processing. He is a member of the Information Processing Society of Japan, the Institute of Electrical Engineers of Japan, IEEE, and ACM.

Kewal K. Saluja obtained his Bachelor of Engineering (B.E.) degree in Electrical Engineering from the University of Roorkee (now IIT- Roorkee), India in 1967. M.S. and Ph.D. degrees in Electrical and Computer Engineering from the University of Iowa, Iowa City in 1972 and 1973 respectively. He is currently with the Department of Electrical and Computer Engineering at the University of Wisconsin-Madison as a Professor, where he teaches courses in logic design, computer architecture, microprocessor based systems, VLSI design and testing, and fault-tolerant computing. Prior to this he was at the University of Newcastle, Australia. Professor Saluja has held visiting and consulting positions at various national and international institutions including University of Southern California, Hiroshima University, Nara Institute of Science and Technology, and the University of Roorkee. He has also served as a consultant to the United Nations Development Program. He was the general chair of the 29th FTCS and he served as an Editor of the IEEE Transactions on Computers (1997–2001). He is currently the Associate Editor for the letters section of the Journal of Electronic Testing: Theory and Applications (JETTA). His research focus is in the areas of Digital Systems Testing, Fault-Tolerant Computing, and Sensor Networks. Professor Saluja has authored and co-authored over 300 technical papers that have appeared in conference proceedings and journals. Professor Saluja is a member of Eta Kappa Nu, Tau Beta Pi, a fellow of the JSPS and a Fellow of the IEEE.