IEICE TRANS. INF. & SYST., VOL.E96–D, NO.8 AUGUST 2013

PAPER Special Section on Reconfigurable Systems

Field Slack Assessment for Predictive Fault Avoidance on Coarse-Grained Reconfigurable Devices

Toshihiro KAMEDA[†], ***Nonmember***, **Hiroaki KONOURA**[†], ***Student Member***, **Dawood ALNAJJAR**[†], ***Nonmember***, **Yukio MITSUYAMA**[††a)], **Masanori HASHIMOTO**[†b)], ***and*** **Takao ONOYE**[†c)], ***Members***

SUMMARY This paper proposes a procedure for avoiding delay faults in field with slack assessment during standby time. The proposed procedure performs path delay testing and checks if the slack is larger than a threshold value using selectable delay embedded in basic elements (BE). If the slack is smaller than the threshold, a pair of BEs to be replaced, which maximizes the path slack, is identified. Experimental results with two application circuits mapped on a coarse-grained architecture show that for aging-induced delay degradation a small threshold slack, which is less than 1 ps in a test case, is enough to ensure the delay fault prediction.

key words: *field test, fault avoidance, coarse-grained reconfigurable device*

1. Introduction

Life-time deterioration due to aging effects is becoming a serious concern in nano-scale integrated circuit design. Aging effects induce an increase in propagation delay, which eventually results in a timing failure making the circuit unusable. To cope with aging-induced delay increase, a certain amount of design margin is usually assigned to make the probability of timing faults negligibly small. However, such a design margin involves significant overhead in circuit speed and area, and degrades the performance of a designable circuit. On the other hand, due to device miniaturization, the necessary margin is increasing significantly. Furthermore, the amount of delay increase is statistically uncertain and varies depending on the operating conditions. Such uncertainty does not only increase the margin further, but also raises the possibility that the delay increase due to aging effects exceeds the design margin in one of the existing paths on a chip.

For coping with aging-induced delay increase, two approaches are studied; suppressing aging effects [1], [2] and eliminating faulty modules [3], [4]. Though aging suppression is effective for extending device life-time, it cannot stop the aging effects completely and its efficacy is limited. On a reconfigurable device, many identical BEs are integrated, regardless of homogeneous or heterogeneous architectures, where BE is a basic element composing the reconfigurable device. This paper focuses on the second approach of faulty module elimination with module replacement, especially in reconfigurable devices that are compatible with module replacement. Using the reconfiguration capability, a faulty BE is replaced with a healthy unused BE [3], [4].

On the other hand, the delays of BEs are different due to manufacturing variability and different environmental conditions for aging. Basically, a used BE and an unused (spare) BE are functionally identical, but their delays could be different. If the delay of a spare BE is shorter than that of a used BE and the replacement between them improves setup timing constraints, the replacement is successful. Note that such a pair of BEs is different from a chip to another since manufacturing variability and aging conditions affect the delay of each BE.

To ensure the successful replacement, we need to identify which BE should be eliminated and which BE should be the substitute. For non-delay faults, the identification of a faulty BE is relatively easy, since a single BE is causing functional incorrectness and BE-by-BE testing, e.g. [5], is effective. In addition, by replacing the faulty BE with a healthy BE, the functional correctness can be achieved. On the other hand, a delay fault occurs when the delay increase accumulates along the path consisting of several BEs, wasting the timing slack. This means that it is not easy to identify which BE should be replaced. In addition, it is not guaranteed that the replacement with a healthy BE resolves delay fault problem, because the replacement involves delay modification due to rerouting.

In this paper, we focus on and propose a procedure for identifying a pair of faulty BE and healthy BE to avoid setup delay faults in field. The proposed procedure for coarse-grained reconfigurable architecture adds a small circuitry to each BE to estimate the slack in a path delay test, which enables delay fault prediction. This fault prediction is applied to the paths of mapped application circuits. The proposed procedure finds the pair of faulty BE and healthy BE that maximizes the slack of the path predicted to cause a timing fault in near future. We experimentally verify how much slack is necessary to ensure the fault prediction.

This paper is organized as follows. Section 2 introduces related works. The proposed procedure for BE replacement is presented in Sect. 3. Experimental results are shown in Sect. 4 and concluding remarks are given in

Manuscript received November 10, 2012.
Manuscript revised March 8, 2013.

[†]The authors are with JST, CREST and the Department of Information Systems Engineering, Osaka University, Suita-shi, 565–0871 Japan.
[††]The author is with JST, CREST and the School of Systems Engineering, Kochi University of Technology, Kami-shi, 782–8502 Japan.
a) E-mail: mitsuyama.yukio@kochi-tech.ac.jp
b) E-mail: hasimoto@ist.osaka-u.ac.jp
c) E-mail: onoye@ist.osaka-u.ac.jp

DOI: 10.1587/transinf.E96.D.1624

Sect. 5.

2. Test Classification and Related Works

2.1 Test Classification

Tests for reconfigurable devices can be classified into two categories. The first category of the test is manufacturer test. This test is performed to make sure all the BEs on a chip satisfy the given functional and timing specifications under all the possible reconfiguration options. On the other hand, the test in the second category aims to verify whether the mapped application circuit works correctly and it is executed by device users.

The second user test can be regarded as a subset of the first manufacturer test, since unused functionalities are not tested. The mapped circuit uses one of available functionalities in each BE, and only this used functionality is tested in the user test, while all the available functionalities including the functionality used in the mapped circuit are tested in the manufacturer test. In addition, the timing specification of the BEs used for non-critical paths can be relaxed without degrading the performance of the mapped application circuit, while the timing specification needs to be tested and satisfied for each BE in the manufacturer test.

The manufacturer test is independent of the mapped circuit and the same test can be applied to all the devices. However, the number of configurations for test is huge. On the other hand, the user test is dependent on the mapped circuit, and the number of configurations for test is one. Test patterns similar to ASICs and SoCs testing are given and circuit functionalities and timing specifications are verified. Thus, these two tests are different and hence the test scheme must be different. For our purpose, the second user test should be extended so that it can execute tests in the field and identify a pair of faulty and healthy BEs for delay fault avoidance.

2.2 Fault Elimination in Reconfigurable Device

Fault elimination by BE replacement in reconfigurable devices is actively studied. A good survey is found in [4]. Most of the works aims to extend the circuit life-time by eliminating hard faults.

Fault elimination methods in reconfigurable devices are classified into two groups.

Hardware-oriented A faulty BE is automatically swapped with a spare using additional wires, switches, and controllers, along the replacement policy.

Reconfiguration-oriented Inherent reconfigurability is exploited for fault avoidance with partial mapping modification.

The hardware-oriented group includes column swap, row direction swap and neighbor swap. Column swap (e.g. [6]) uses the column (row) redundancy on the reconfigurable device. Row direction swap (e.g. [7]) eliminates a faulty BE using a spare on the same row. Neighbor swap (e.g. [8]) replaces a faulty BE with a neighboring spare to minimize the down-time associated with replacement.

The reconfiguration-oriented group utilizes dynamic reconfiguration capability. Dynamic P&R is a self-repair technique with on-the-fly partial placement and routing (P&R) [9]. Pre-compiled reconfiguration selects a proper configuration from ones that were prepared beforehand and loads it [10]. The attainable reliability and MTTF enhancement and necessary hardware overhead are different among these methods [3].

2.3 Test Methods in Reconfigurable Device

Reference [5] proposes an approach in which an exhaustive test using LFSR (linear feedback shift register) is applied to each BE in a coarse-grained reconfigurable device. TPG (test pattern generator) and RA (response analyzer) are implemented using BEs. Then, CUT (circuit under test), TPG and RA are shifted several times to test all the BEs. References [11], [12] propose an online test scheme for FPGAs. This online test scheme has a special region called STAR (self-test area) and shifts STAR in the device without functional down-time. Inside STAR, an exhaustive test is carried out in each BE. These methods perform BE-by-BE manufacturer test.

Delay testing for reconfigurable devices is also studied [13], [14]. Reference [14] maps identical paths and compares their delays using oscillators and counters. The delay difference is used as a fault criterion. This method gives some information on delays of the tested paths, but it cannot directly guide the elimination of faulty BEs, namely, it cannot tell when and how faulty BEs are eliminated while keeping the functionality and satisfying the timing specification. This method belongs to manufacturer test. Reference [13] proposes a method for FPGA that prepares and uses sets of configuration information for testing critical paths of the mapped application circuit. TPG and RA are implemented around the path under test, and at-speed test is executed. This method is categorized into user test.

3. Proposed Fault Avoidance Procedure

This section presents the proposed testing procedure for coarse-grained reconfigurable devices aiming at predictive delay fault avoidance.

3.1 Requirements

The goal of the faulty module replacement is to extend the life-time of application circuits mapped on reconfigurable devices. The aim of this work is to present a procedure for ensuring the BE replacement without causing errors. For this aim, the following requirements are listed.

- User test is good enough for ensuring the mapped circuit behavior. Manufacturer test could be over-testing

in most cases and shortens the life-time.
- Delay faults must be predicted before timing errors would happen. Error detection must be accompanied by an error recovery system (e.g. Razor [15]) and it is expensive.
- Testing needs to guide faulty BE elimination. Just identification of faulty BE is not enough.
- During faulty BE elimination, other circuits on the same reconfigurable device should be able to continue their operation. For this, clock signal manipulation is not allowed, and the system clock must be used for testing as well.

For non-delay faults, BE-by-BE manufacturer tests like [5], [11], [12] are effective, since a single BE usually causes errors, though it might be somewhat over-testing since unused functionalities are also tested. The identified faulty BE is then replaced with a spare BE. On the other hand, for path delay faults, identifying a faulty BE to be replaced is difficult, since a path delay fault occurs when the delay increase accumulated along the path exhausts the timing slack. Furthermore, even if we could know which BE induces the largest delay increase, the BE is not always the one to be replaced. It is because the replacement of BEs, which is assumed to be executed by neighbor swap or partial dynamic reconfiguration in this work, cannot necessarily improve the slack depending on the neighboring BE usage and the performance of spare BE. The routing modification involved with BE replacement could spoil the replacement, and the performance of spare BE fluctuates due to manufacturing variability and aging. This means that we need to identify a pair of BEs to be replaced that improves the slack.

3.2 Fault Avoidance Procedure

Figure 1 shows the proposed procedure for eliminating the faulty BE. When the circuit enters in standby mode, fault prediction starts. We here assume that circuits have two operational modes; active mode and standby mode. In the active mode circuits are functioning, while they are idling in the standby mode. The standby mode is often exploited for clock gating and power gating. This work carries out the proposed procedure in the standby mode. We first select a path and assess the path slack. The mechanism of this slack assessment will be explained in the next section. If the slack is too small, we go into BE replacement phase. Otherwise, we select and evaluate the next path. This path evaluation continues until the standby mode ends.

When the path slack is estimated to be smaller than the threshold value, we try to increase the path slack by replacing one of BEs composing the path with a spare BE, where this process corresponds to "identify a pair of BEs" in Fig. 1. Here, there is no clue as to which BE should be replaced. Therefore, an exhaustive evaluation is performed as follows. We first pick a BE on the path, and find a spare BE that can be used for the BE replacement. We then replace them and assess the path slack. This assessment is

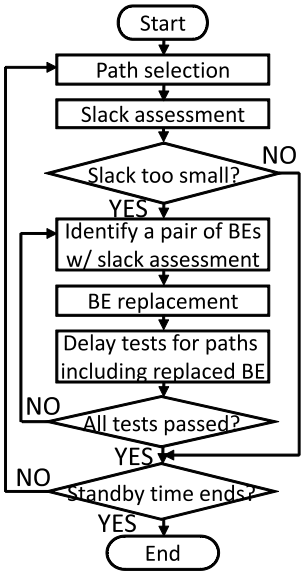

Fig. 1 Proposed Procedure of Fault Avoidance.

repeated for all the possible pairs of the BEs on the path and the replaceable spare BEs. Once the slack values of all the possible pairs are available, we select the pair that maximizes the path slack and replace the pair of BEs. After that, the paths that goes through the replaced BE are tested. If all the tests pass, the BE replacement succeeds. If all the tests do not pass, we choose another pair and execute the tests. At a glance, this exhaustive evaluation seems to be time-consuming. However, in case of coarse-grained reconfigurable device, the number of BEs composing a path is not large, e.g. at most five, and the number of spares is limited for area efficiency. Therefore, the number of possible pairs is limited. Besides, a problem on how much spares are necessary to succeed BE replacement is beyond the scope of this paper, and we will study that in the near future in association with [3].

When implementing a circuit on a reconfigurable device, 100% BE utilization ratio cannot be obtained in practice and a certain portion of BEs remain unused. Such unused BEs can be used as spare BEs. In addition, to improve the efficiency of BE replacement, we may need to place unused BEs intentionally as spare BEs. Thus, spare BEs are placed in the design time. On the other hand, the placement of spare BEs affects the replacement efficiency, as suggested in [3]. The intelligent placement of spare BEs is included in our future works.

With this procedure, we can replace a faulty BE that will start to induce timing errors in the near future with a spare cell without causing any timing errors originating from the replacement. Besides, the frequency of the slack assessment depends on the frequency and duration of the standby mode, and then the frequency of the slack assessment is application dependent. According to the frequency and duration of the standby mode, we need to determine an appropriate threshold value of the slack. The threshold value

of the slack is experimentally investigated in Sect. 4.

3.3 Slack Assessment for Fault Prediction

Ordinary path delay tests tell whether the path slack is positive or negative. When the path slack becomes almost zero, we need to perform fault avoidance, but such information is not available. To predict path delay faults, we add selectable delay just in front of the capture FF as shown in Fig. 2. In a normal operation, the upper path in the selectable delay is used. On the other hand, in slack assessment, the lower path with tunable delay is selected. In this case, the tunable delay is inserted in the path under test, which means the setup timing constraint becomes tighter by the amount of tunable delay. Under this condition, if the path delay test passes, the path has slack larger than the amount of tunable delay. In contrast, if the path delay test fails, the path slack is smaller than the amount of tunable delay. By changing the amount of tunable delay, we can know the range of the slack. In the example of Fig. 2, the slack is estimated to be between 100 ps and 200 ps. A similar idea of this selectable delay is found in timing error predictive flip-flop [16].

To apply the above idea of slack assessment, the selectable delay is inserted in front of pipeline registers in the reconfigurable device as illustrated in Fig. 3. With this insertion, all the paths mapped in the device go through at least one selectable delay. In this example, the cycle time is assumed to be 10 ns, while the pipeline registers in BE2 and BE3 are disabled. This path under test does not pass the test when the selectable delay is 200 ps in BE4. In this case, the path slack is smaller than 200 ps and BE replacement is invoked. The right figure is the slack assessment when BE2 is replaced with BE2′. The test passes even when the selectable delay is set to 800 ps, which means that the replacement improves the slack from <200 ps to > 800 ps.

The slack assessment with multiple threshold slack values Th_1 and Th_2 ($Th_1 \gtrsim Th_2$) can be used to improve the efficiency of BE replacement. Th_1 triggers a process to prepare configuration information for identifying a BE pairs, where this process will be executed on, for example, an embedded processor. Then, BE replacement invoked by Th_2 will use this prepared configuration information. In this case, the computational time to prepare the configuration information can be concealed in the proposed procedure of fault avoidance.

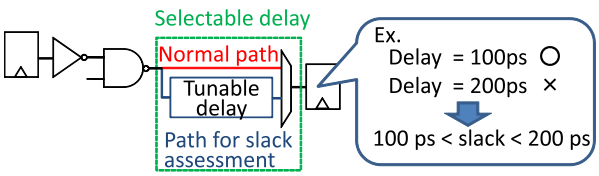

Fig. 2 Slack Assessment with Selectable Delay.

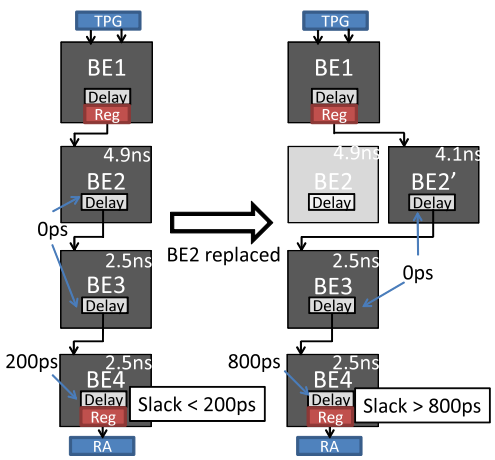

Fig. 3 Slack Assessment in Reconfigurable Devices.

4. Experimental Results

This section experimentally confirms that the proposed procedure can predict timing faults before error occurance.

4.1 Target Architecture

Figure 4 illustrates the target architecture for this study, which is based on a coarse-grained dynamically reconfigurable architecture introduced in [17], [18]. In this architecture, identical BEs are aligned repeatedly in a two-dimensional array. Each BE is connected to adjacent BEs in four directions with two wires for data (D1 and D2) and one wire for flag (F). Interconnections are configured with multiplexers included in BEs. Each BE contains a 16-bit ALU receiving 1 or 2 inputs and performs both arithmetic and logic operation for them. Each pipeline data register (AREG, BREG and YREG) and flag register (FREG) can be enabled and disabled by multiplexer. Selectable delay is inserted at ALU output. Note that a path from YREG to AREG or BREG of the next BE does not include selectable delay, but this path is very short and slack assessment is unnecessary. For improving routing efficiency, 2x-, 3x- and 4x-long interconnects are implemented.

The above architecture was implemented with Verilog-

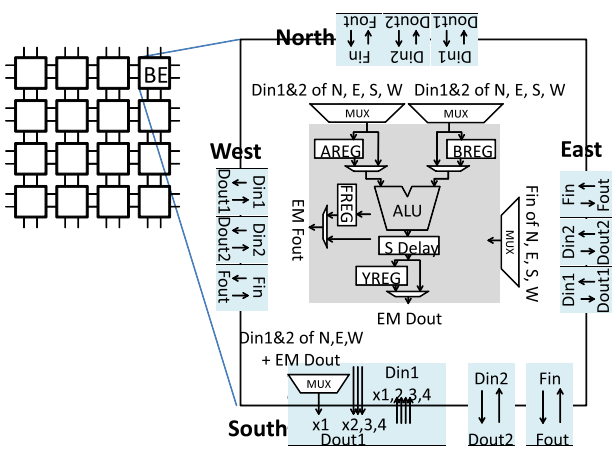

Fig. 4 Target Architecture.

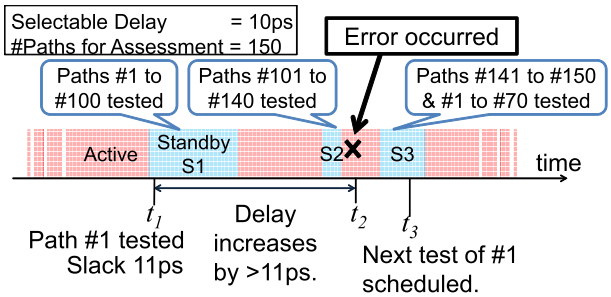

Fig. 5 A situation that a timing error occurss.

HDL and a 4x4 array was synthesized by a commercial logic synthesizer (Synopsys Design Compiler) with an industrial 65 nm standard cell library. The number of gates, which was converted into 2-input NAND gate, is 114,421, which does not include storage to store configuration information for slack assessment. The number of bits to configure a BE is 101 bits. The following experiments assumed that the configuration information was given using a 1-bit bus and its bus clock cycle was 100 ns.

On this 4x4 array, we implemented FIR filter and FFT. Without manufacturing variability, the critical path delay of FIR filter was 7388 ps and the operation with 8209 ps cycle time (10% larger) was assumed. Similarly, the cycle time of 5940 ps was assumed for FFT. The numbers of paths in FIR filter and FFT are 1761 and 440. All paths were assumed to be sensitizable for path delay test. In this setup, 285 ms is necessary to test all the paths for FIR filter and 71 ms for FFT. The computational time to prepare configuration information necessary to identify a pair of BEs is not considered in the experiments, since this process becomes active very infrequently, and it can be concealed as explained in Sect. 3.3.

4.2 Setup and Evaluation Metric

The most important mission of the proposed procedure is to find potential faults. If the threshold slack is set to be a large value, the probability that actual faults are missed becomes lower and lower. On the other hand, in this case, many paths are predicted to be faulty and many BEs must be replaced even though those paths still have timing slack. This means that the threshold of the path slack for BE replacement must be carefully determined considering this trade-off.

Figure 5 illustrates a situation in which timing errors may happen in the proposed procedure. In this example, 150 paths are selected for slack assessment, and the threshold slack value for BE replacement is 10 ps. In the first standby time of S1, path #1 was tested at time t_1, and the slack was 11 ps at this moment. Then, then path #1 was not selected for BE replacement at time t_1. During this standby time of S1, paths #1 to #100 were tested. Paths #101 to #140 were tested in the second standby time of S2, and the next test for path #1 is scheduled at t_3 in the next standby time of S3. However, before the next test at t_3, the delay increase from the previous test at t_1 exceeded 11 ps, and at a certain timing of t_2 ($t_1 < t_2 < t_3$), path #1 was activated and induced a timing error. This example indicates that the delay increase that possibly arises during the time interval between slack assessments (in this example, t_1 to t_3) must be smaller than the threshold slack value.

Another situation of missing timing error is as follows: to reduce the amount of memory storing configuration information for slack assessment, the number of paths for slack assessment is reduced. On the other hand, a path that is not included in the set may cause a timing error. Manufacturing variability makes this situation happen easily even while timing critical paths are selected for slack assessment according to the timing information in design time.

Therefore, we evaluate the probability that no timing errors happen in 10 years before the slack assessment triggers BE replacement as a metric called "success probability", where a success means that no actual timing errors arise prior to the timing error prediction. We change the threshold slack, the lengths of active time and standby time, and the number of paths for slack assessment. The lengths of active time and standby time were assumed to fluctuate according to normal distributions. In experiments, we varied the average times, but the standard deviations were fixed to 30% of the average times. In addition, to consider manufacturing variability, each gate delay varied according to normal distribution. The average delay was extracted from the logic synthesis result and the standard deviation was set to 5% of the average. In evaluating the metric, 1,000 devices with manufacturing variation were evaluated with temporally fluctuating active and standby times. The success probability was evaluated assuming that aging effects linearly increased the gate delay by 30% in 10 years, which was referred to 11% in 3 years in [19]. This aging speed corresponds to 3.4×10^{-4}% per hour and 8.2×10^{-3}% per day. A simulator to evaluate the success probability was implemented with C++.

In this setup, we discuss the overhead to implement the proposed method. The proposed procedure requires an external memory to store configuration information for testing. 356 kB and 89 kB storages are necessary for FIR and FFT, respectively. In addition, a computational resource, such as an embedded processor, to generate test configurations for identify a pair of BEs. When an embedded processor on the same chip can be temporally used for this purpose, the overhead can be minimized. In this case, a memory to store the program for test configuration generation is necessary.

4.3 Results

Figures 6 and 7 show relations of FIR filter between success probability and threshold slack with average active time of 1 hour and 1 day, respectively. As the threshold slack increases, the success probability increases to 100%. When the average standby time is 1 second (Fig. 6), the necessary threshold slack to attain 100% success probability is 0.001% of the clock cycle time, whereas the delay increase in an hour is 0.00034%. Taking into account the variation of the

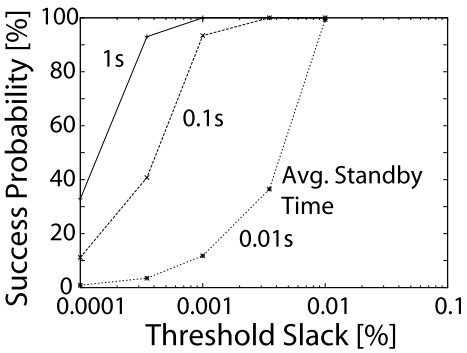

Fig. 6 Success Probability versus Threshold Slack (FIR filter, Average Active Time 1 Hour, All Paths Tested).

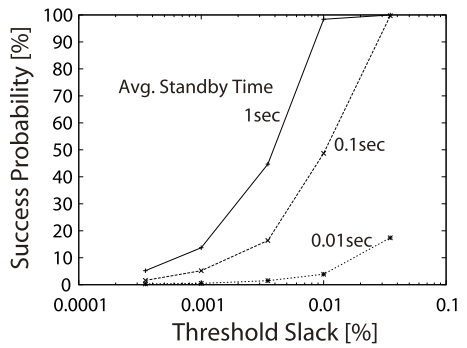

Fig. 7 Success Probability versus Threshold Slack (FIR filter, Average Active Time 1 Day, All Paths Tested).

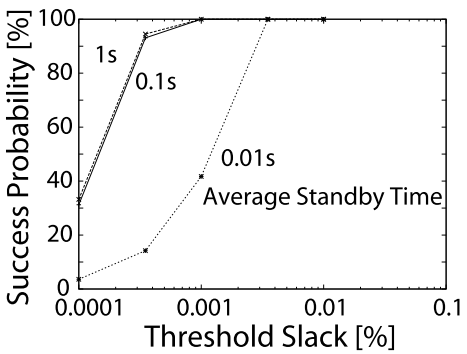

Fig. 8 Success Probability versus Threshold Slack (FFT, Average Active Time 1 Hour, All Paths Tested).

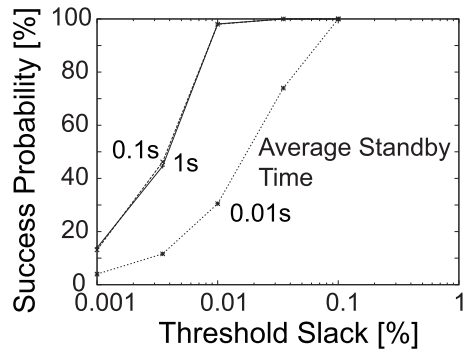

Fig. 9 Success Probability versus Threshold Slack (FFT, Average Active Time 1 Day, All Paths Tested).

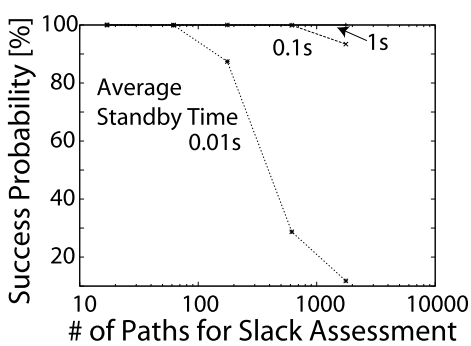

Fig. 10 Success Probability versus # of Paths for Test (FIR filter, Average Active Time 1 Hour, Threshold Slack 0.001%).

active time, this result is reasonable. A similar discussion is applicable to the 1 day case of Fig. 7.

On the other hand, the necessary threshold slack to attain 100% success probability is different depending on the average standby time. When the average standby time is 0.01 seconds, typically 29 standby times are necessary to perform slack assessment for all the paths, since 285 ms is necessary to test all the paths. This means the average interval of path slack assessment becomes the average active time multiplied by 29. On the other hand, the active and standby times are fluctuating. To overcome short standby time and time fluctuation, the necessary threshold slack becomes ten times larger in Fig. 6.

Figures 8 and 9 show the relation between success probability and threshold slack in FFT, where the average active times are 1 hour and 1 day, respectively. In FFT, the number of existing paths is smaller than that of FIR, and hence 0.1 seconds is long enough to complete all the paths. Therefore, the results of 1 and 0.1 seconds are almost the same.

Figure 10 presents success probability when the number of paths for slack assessment was reduced in FIR filter. As explained in Sect. 4.2, when the number paths for slack assessment decreases, the possibility that other paths would cause timing errors may increase. On other hand, the standby time needed to test the paths for slack assessment becomes shorter. Thus, there are two aspects; the path reduction may improve the success probability or may degrade the success probability. On the other hand, Fig. 10 indicates that if the standby time is short (0.01 seconds), reducing the number of paths for slack assessment is helpful to improve the success probability. Looking at the case of longer standby time (1 and 0.1 seconds), the success probability is unchanged compared to Fig. 6, and degradation of success probability was not caused by the path reduction in this test case. Figure 11 shows the FFT result, and the similar tendency is observed in the figure.

These results in this work reveal that the small threshold slack can attain 100% success probability. In case of

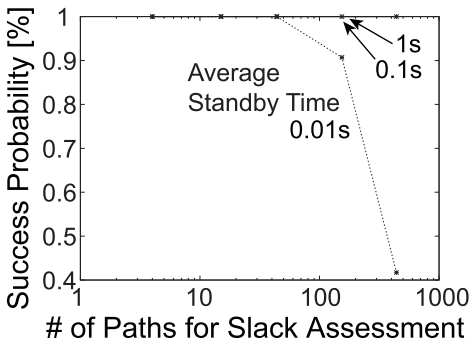

Fig. 11 Success Probability versus # of Paths for Test (FFT, Average Active Time 1 Hour, Threshold Slack 0.001%).

FIR filter with 1 hour active time and 1 second standby time, 0.001% threshold slack corresponds to less than 1 ps. This small threshold is highly desirable, since only the paths whose slack is less than 1 ps are judged to be faulty and to be replaced. Smaller number of replacement can save the number of spares, improve life-time extension and reduce area overhead.

5. Conclusion

This paper presented a procedure for avoiding delay faults with slack assessment in coarse-grained reconfigurable architectures. The proposed procedure predicts timing faults before errors happen using selectable delay embedded in BEs, guides and ensures BE replacement without new faults originating from BE replacement. Experimental results using two application circuits show that small threshold slack for BE replacement is enough to successfully predict timing faults.

Acknowledgements

This work is partly supported by NEDO.

References

[1] D.R. Bild, G. Bok, and R.P. Dick, "Minimization of NBTI performance degradation using internal node control," Proc. DATE, pp.148–153, 2009.
[2] A. Calimera, E. Macii, and M. Poncino, "NBTI-aware sleep transistor design for reliable power-gating," Proc. GLSVLSI, pp.333–338, May 2009.
[3] H. Konoura, Y. Mitsuyama, M. Hashimoto, and T. Onoye, "Implications of reliability enhancement achieved by fault avoidance on dynamically reconfigurable architectures," Proc. FPL, pp.189–194, 2011.
[4] Matthew G. Parris, Carthik A. Sharma, and Ronald F. Demara, "Progress in autonomous fault recovery of field programmable gate arrays," Proc. ACM Computing Surveys (CSUR), vol.43, Issue 4, Oct. 2011.
[5] T. Inoue, T. Fujii, and H. Ichihara, "A self-test of dynamically reconfigurable processors with test frames," IEICE Trans. Inf. & Syst., vol.E91-D, no.3, pp.756–762, March 2008.
[6] A. Shibayama, H. Igura, M. Mizuno, and M. Yamashina, "An autonomous reconfigurable cell array for fault-tolerant LSIs," ISSCC Dig. Tech., pp.230–231, 1997.
[7] Z.E. Rakosi, M. Hiromoto, H. Ochi, and Y. Nakamura, "Hot-swapping architecture extension for mitigation of permanent functional unit faults," Proc. FPL, pp.578–581, 2009.
[8] A. Doumar, S. Kaneko, and H. Ito, "Defect and fault tolerance FPGAs by shifting the configuration data," Proc. DFT, pp.377–385, 1999.
[9] N.J. Macias and L.J.K. Durbeck, "Adaptive methods for growing electronic circuits on an imperfect synthetic matrix," Biosystems, vol.73, no.3, pp.173–204, 2004.
[10] L. Shang, M. Zhou, Y. Hu, and E. Yang, "A Domain partition model approach to the online fault recovery of FPGA-based reconfigurable systems," IEICE Trans. Fundamentals, vol.E94-A, no.1, pp.290–299, Jan. 2011.
[11] J.M. Emmert, C.E. Stroud, and M. Abramovici, "Online fault tolerance for FPGA logic blocks," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.15, no.2, pp.216–226, Feb. 2007.
[12] M. Abramovici, C.E. Stroud, and J.M. Emmert, "Online BIST and BIST-based diagnosis of FPGA logic blocks," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.12, no.12, pp.1284–1294, Dec. 2004.
[13] I.G. Harris, P.R. Menon, and R. Tessier, "BIST-based delay path testing in FPGA architectures," Proc. ITC, pp.932–938, 2001.
[14] M. Abramovici and C. Stroud, "BIST-based delay-fault testing in FPGAs," Proc. International On-Line Testing Workshop, pp.131–134, 2002.
[15] S. Das, D. Roberts, L. Seokwoo, S. Pant, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, "A self-tuning DVS processor using delay-error detection and correction," IEEE J. Solid-State Circuits, vol.41, no.4, pp.792–804, April 2006.
[16] H. Fuketa, M. Hashimoto, Y. Mitsuyama, and T. Onoye, "Adaptive performance compensation with in-situ timing error predictive sensors for subthreshold circuits," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.20, no.2, pp.333–343, Feb. 2012.
[17] D. Alnajjar, Y. Ko, T. Imagawa, H. Konoura, M. Hiromoto, Y. Mitsuyama, M. Hashimoto, H. Ochi, and T. Onoye, "Coarse-grained dynamically reconfigurable architecture with flexible reliability," Proc. FPL, pp.186–192, 2009.
[18] D. Alnajjar, H. Konoura, Y. Ko, Y. Mitsuyama, M. Hashimoto, and T. Onoye, "Implementing flexible reliability in a coarse grained reconfigurable architecture," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., in press.
[19] K. Kang, H. Kufluoglu, M.A. Alam, and K. Roy, "Efficient transistor-level sizing technique under temporal performance degradation due to NBTI," Proc. ICCD, pp.216–221, 2006.

Toshihiro Kameda received the B.E. and M.E. degrees in Information Systems Engineering from Osaka University, Osaka, Japan, in 2010 and 2012, respectively. He is currently with NTT communications. His research interest includes testing and fault avoidance for reconfigurable devices.

Hiroaki Konoura received the B.E. and M.E. degrees in Information Systems Engineering from Osaka University , Osaka, Japan, in 2009 and 2011. He is currently pursuing the Ph.D. degree in the Department of Information Systems Engineering at Osaka University. His major interests include mitigation of circuit degradation and fault tolerance techniques by using partial reconfiguration. He is a student member of IEEE.

Dawood Alnajjar received the B.E. degree from the University of Jordan, Amman, Jordan in 2006. He received the M.E. degree in Information Systems Engineering from Osaka university, Osaka, Japan in 2010. He is currently pursuing his Ph.D. degree in the Department of Information Systems Engineering at Osaka University. His interest includes reconfigurable computing, computer architecture, and reliable systems.

Yukio Mitsuyama received the B.E., M.E., and Ph.D. degrees in Information Systems Engineering from Osaka University, Japan, in 1998, 2000, and 2010, respectively. He was an Assistant Professor in Graduate School of Engineering, Osaka University. Since 2011, he has been an Assistant Professor in the School of Engineering, Kochi University of Technology. His research interests include reconfigurable architecture and its VLSI design. He is a member of IEEE and IPSJ.

Masanori Hashimoto received the B.E., M.E. and Ph.D. degrees in Communications and Computer Engineering from Kyoto University, Kyoto, Japan, in 1997, 1999, and 2001, respectively. Since 2004, he has been an Associate Professor in Department of Information Systems Engineering, Graduate School of Information Science and Technology, Osaka University. His research interest includes computer-aided-design for digital integrated circuits, and high-speed circuit design. Dr. Hashimoto served on the technical program committees for international conferences including DAC, ICCAD, ASP-DAC, DATE, ISPD, ICCD and ISQED. He is a member of IEEE and IPSJ.

Takao Onoye received the B.E. and M.E. degrees in Electronic Engineering, and Dr.Eng. degree in Information Systems Engineering all from Osaka University, Japan, in 1991, 1993, and 1997, respectively. He is currently a professor in the Department of Information Systems Engineering, Osaka University. His research interests include media-centric low-power architecture and its SoC implementation. He is a member of IEEE, IPSJ, and ITE-J.