A PREDICTIVE DELAY FAULT AVOIDANCE SCHEME FOR COARSE-GRAINED RECONFIGURABLE ARCHITECTURE

Toshihiro Kameda[1], *Hiroaki Konoura*[1], *Dawood Alnajjar*[1], *Yukio Mitsuyama*[2], *Masanori Hashimoto*[1] *and Takao Onoye*[1]

[1]Dept. Information Systems Engineering, Osaka University & JST, CREST
[2]School of Systems Engineering, Kochi University of Technology & JST, CREST

ABSTRACT

A scheme for avoiding delay faults with slack assessment during standby time is proposed in this paper. The proposed scheme performs path delay testing and checks if the slack is larger than a threshold value using selectable delay embedded in basic elements (BE) on a coarse-grained reconfigurable device. If the slack is smaller than the threshold, a pair of BEs to be replaced, which maximizes the path slack, is identified. Experimental results 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.

1. INTRODUCTION

Life-time deterioration due to aging effects is becoming a serious concern in nano-scale VLSI design. Aging effects increase propagation delay, which eventually results in timing failure and make 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 design margin involves significant overhead in circuit speed and area, and degrades the performance of a designable circuit.

For coping with aging-induced delay increase, two approaches are studied; suppressing aging effects (e.g. [1]) and eliminating faulty modules [2, 3]. Though aging suppression is effective for extending device life-time, it cannot stop the aging effects completely and its efficacy is limited.

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 BE [2, 3], where BE is basic element composing a reconfigurable device. To ensure the successful replacement, we need to identify which BE should be eliminated and which BE should 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. [4]), 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 happens when delay increase accumulated along with the path consisting of several BEs wastes 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.

Tests for reconfigurable devices can be classified into two categories. The first category is manufacturer test aiming to make sure the BEs on a chip satisfy the given functional and speed specifications under all the possible reconfiguration options (e.g. [5]). 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. This user test can be regarded as a subset of the first manufacturer test, since unused functionalities are not tested. In addition, the speed specifications of BEs used for non-critical paths can be relaxed without degrading the application performance. For our purpose, the second user test should be extended so that it can identify a pair of faulty and healthy BEs for delay fault avoidance in field.

This paper proposes a scheme for identifying a pair of faulty BE and healthy BE to avoid setup delay faults. The proposed scheme for coarse-grained reconfigurable architecture adds a small circuitry to each BE to estimate slack in path delay test, which enables delay fault prediction. The proposed scheme 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.

2. PROPOSED FAULT AVOIDANCE SCHEME

2.1. Requirements

The goal of this work is to extend the life-time of application circuits mapped on reconfigurable device without outputting errors. To extend the life-time by replacing aged faulty BEs without errors, the requirements for testing are listed.

- User test is good enough for ensuring the mapped circuit behavior. Manufacturer test could be over-testing in most cases and shorten the life-time.
- Delay faults must be predicted before timing errors happen. Error detection requires error recovery system (e.g. Razor [6]) and it is expensive.
- Testing needs to guide faulty BE elimination. Just identification of faulty BE is not enough.

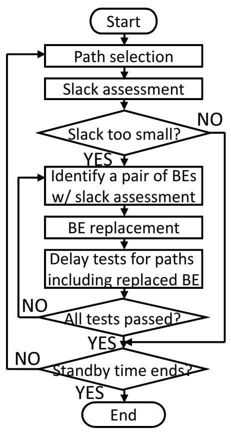

Fig. 1. Proposed Procedure of Fault Avoidance.

- During faulty BE elimination, other circuits on the same device should continue their operation. For this, clock signal manipulation is not allowed, and the system clock must be used for testing.

2.2. Fault Avoidance Procedure

Figure 1 shows the proposed procedure for eliminating faulty BE. When the circuit enters in standby mode, fault prediction starts. 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 time 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 up a BE on the path, and find a spare cell that can be used for BE replacement. We then replace them and assess the path slack. This assessment is repeated for all the possible pairs of the BE on the path and replaceable spare BE. Once the slack values of all the possible pairs are available, we select the pair that maximizes the path slack and replace them. After that, the paths that goes through the replaced BE are tested. If all the tests pass, the BE replacement succeeds. If not, we select 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, for example 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 this paper, and we will study that in the near future extending [2].

With this procedure, we can replace a faulty BE that will start to induce timing errors in the near future with a spare

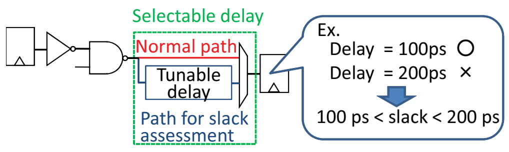

Fig. 2. Slack Assessment with Selectable Delay.

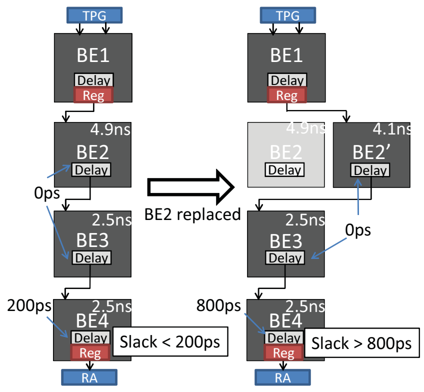

Fig. 3. Slack Assessment in Reconfigurable Device.

cell without causing any timing errors originating from the replacement. The threshold value of the slack is experimentally investigated in Section 3.

2.3. 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 given. To predict path delay faults, we add selectable delay just in front of 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, when 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 100ps and 200ps. A similar idea of this selectable delay is found in timing error predictive flip-flop [7].

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 go through at least one selectable delay. In this example, the cycle time is assumed to be 10ns, and pipeline registers in BE2 and BE3 are disabled. This path under test does not pass the test when the selectable delay is 200ps in BE4. In this case, the path slack is smaller than 200ps and BE replacement is invoked in this example. The right figure is the slack assessment when BE2 is replaced

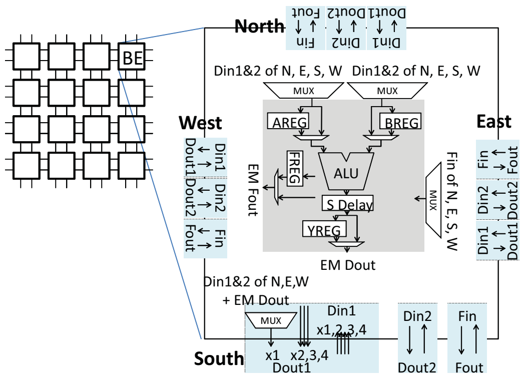

Fig. 4. Assumed Architecture.

with BE2′. The test has passed even when the selectable delay is set to 800ps, which means the replacement improves the slack from “ <200ps ” to “ > 800ps ”.

3. EXPERIMENTAL RESULTS

3.1. Assumed Architecture

Figure 4 illustrates the target architecture for this study, referring to a coarse grained dynamically reconfigurable architecture in [8]. In this architecture, identical BEs are aligned repeatedly in a 2-D 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. 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. Here, a path from YREG to AREG or BREG of the next BE does not include the selectable delay, but this path is very short and slack assessment is unnecessary. For better routability, 2x-, 3x- and 4x-long interconnects are implemented.

The above architecture was implemented with Verilog-HDL and a 4x4 array was synthesized with an industrial 65nm 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 data for slack assessment. The number of bits to configure a BE is 101 bits. The following experiments assumed that the configuration data was given using a 1-bit bus with 10MHz clock.

On this 4x4 array, we implemented FIR filter and FFT. Without manufacturing variability, the critical path delay of FIR filter was 7,388 ps and the operation with 8,209 ps cycle time (10% larger) was assumed. Similarly, the cycle time of 5,940 ps was assumed for FFT. The numbers of paths in FIR filter and FFT are 1,761 and 440. All paths were assumed to be sensitizable for path delay test. In this setup, 28ms is necessary to test all the paths for FIR filter and 7ms for FFT.

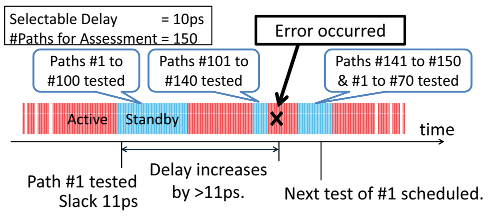
Selectable Delay = 10ps
#Paths for Assessment = 150
Error occurred
Paths #1 to #100 tested
Paths #101 to #140 tested
Paths #141 to #150 & #1 to #70 tested
Active
Standby
time
Path #1 tested
Slack 11ps
Delay increases by >11ps.
Next test of #1 scheduled.

Fig. 5. A situation that a timing error happens.

3.2. Setup and Evaluation Metric

The most important mission of the proposed scheme is to find out all potential faults. If a larger threshold slack is selected, 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 exemplifies a situation that timing errors may happen in the proposed scheme. 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, the slack of path #1 was 11 ps. Then, then path #1 was not selected for BE replacement. During this standby time, paths #1 to #100 were tested. Paths #101 to #140 were tested in the second standby time, and the next test for path #1 is scheduled in the next standby time. However, before the next test, the delay increase from the last test exceeded 11 ps, and at a certain timing, path #1 was activated and induced an error. This example indicates that the delay increase that possibly arises during the time interval between slack assessments 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 data 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 easily happen even while timing critical paths are selected for slack assessment according to the timing information in design time.

We therefore evaluate the possibility that no timing errors happen in 10 years as a metric called success probability. 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. Also, 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 ef-

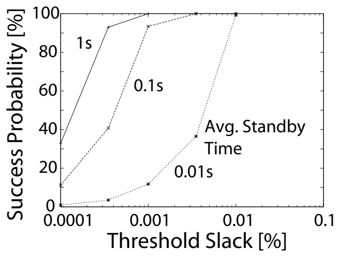

Fig. 6. Success Probability vs. Threshold Slack (FIR filter).

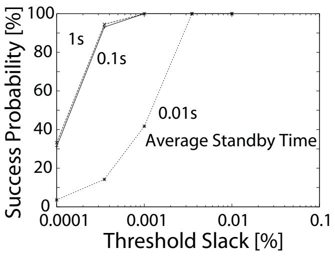

Fig. 7. Success Probability vs. Threshold Slack (FFT).

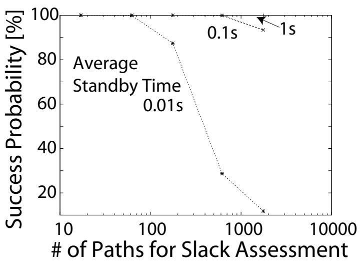

Fig. 8. Success Probability vs. # of Paths for Test (FIR filter, Threshold Slack 0.01%).

fects linearly increased the gate delay by 30% in 10 years as an example. 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++.

3.3. Results

Figures 6 shows relations of FIR filter between success probability and threshold slack with average active time of 1 hour. As the threshold slack increases, the success probability increases to 100%. When the average standby time is 1 s (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 active time, this result is reasonable. A similar discussion is applicable to other experiments with longer average active times.

On the other hand, the necessary threshold slack to attain 100% success probability depends on the average standby time. When the average standby time is 0.01 s, typically three standby times are necessary to perform slack assessment for all the paths, since 28 ms is necessary to test all the paths. This means the average interval of path slack assessment becomes the average active time multiplied by three. Furthermore, 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.

Figure 7 shows the relation between success probability and threshold slack in FFT. The average active time is 1 hour. In FFT, the number of paths is smaller than that of FIR, and hence 0.1 s is long enough to test all the paths. Therefore, the results of 1 s and 0.1 s are almost the same.

Figure 8 presents success probability when the number of paths for slack assessment was reduced in FIR filter. When the number of paths for slack assessment decreases, the possibility that other paths would cause timing errors may increase. On the other hand, the standby time needed to test the paths becomes shorter. Thus, there are two aspects; the path reduction may improve or degrade the success probability. Besides, Fig. 8 indicates that if the standby time is short (0.001 s), reducing the number of paths for slack assessment is helpful to improve the success probability. In the case of longer standby time (1 s and 0.1 s), the success probability was unchanged from Fig. 6, which means the success probability was not degraded by the path reduction in this test case.

These results reveal that the small threshold slack can attain 100% success probability. In case of FIR filter with 1 hour active time and 1 s 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.

4. CONCLUSION

This paper presented a scheme for avoiding delay faults with slack assessment in coarse-grained reconfigurable device. The proposed scheme 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. Future works include detailed evaluation on implementation overhead.

5. ACKNOWLEDGEMENT

This work is partly supported by NEDO.

6. REFERENCES

[1] D. R. Bild et al., "Minimization of NBTI Performance Degradation Using Internal Node Control," *Proc. DATE,* pp.148–153, 2009.

[2] H. Konoura et al., "Implications of Reliability Enhancement Achieved by Fault Avoidance on Dynamically Reconfigurable Architectures," *Proc. FPL,* pp.189 – 194, 2011

[3] M. G. Parris et al., "Progress in autonomous fault recovery of field programmable gate arrays," *Proc. ACM Computing Surveys,* vol.43, Issue 4, Oct. 2011.

[4] T. Inoue et al., "A Self-Test of Dynamically Reconfigurable Processors with Test Frames," *IEICE Trans. Information and Systems*, vol.E91-D, No. 3, pp. 756 – 762, Mar. 2008.

[5] J. S. J. Wong et al., " Self-Measurement of Combinatorial Circuit Delays in FPGAs, " *ACM Trans. Reconfigurable Technology and Systems*, vol. 2, no. 2, pp. 1-22, Jun. 2009.

[6] S. Das et al., "A self-tuning DVS processor using delay-error detection and correction," *IEEE Journal of Solid-State Circuits*, vol.41, pp.792–804, Apr. 2006.

[7] H. Fuketa et al., "Adaptive Performance Compensation with In-Situ Timing Error Predictive Sensors for Subthreshold Circuits," *IEEE Trans. VLSI Systems*, vol. 20, no. 2, pp. 333–343, Feb. 2012.

[8] D. Alnajjar et al., "Coarse-Grained Dynamically Reconfigurable Architecture with Flexible Reliability," *Proc. FPL*, pp. 186–192, 2009.