TECHNICAL REPORT

Timing Error Detection and Correction in Pipelines Constructed of Scan Flip-Flops

Andreas Floros, Yiorgos Tsiatouhas and Chrisovalantis Kavousianos

University of Ioannina
Department of Computer Science
Panepistimioupolis, P.O. Box 1186, 45110 Ioannina, Greece (Hellas)
e-mail: {afloros,tsiatouhas,kabousia}@cs.uoi.gr

Abstract

Timing errors, due to environmental or operating conditions or due to timing faults that may escape fabrication tests, are a real concern in nanometer technology, high complexity and high frequency circuits. In this work a concurrent error detection and correction circuit and technique are presented that are applicable in pipeline architectures constructed of scan Flip-Flops. In addition a pipeline error recovery mechanism is illustrated. The proposed circuit is characterized by low silicon area requirements, compared to earlier approaches, and negligible penalty on performance.

Index Terms – Timing error detection and correction, scan Flip-Flops, pipeline architectures, concurrent testing.

Contact Author:

Yiorgos Tsiatouhas
University of Ioannina
Department of Computer Science
Panepistimioupolis, P.O. Box. 1186
45110 Ioannina, Greece (Hellas)
Tel.: +30 26510 98853
Fax: +30 26510 98883
e-mail: tsiatouhas@cs.uoi.gr

***This work is** co-funded by the European Union - European Social Fund (ESF) & National Sources, in the framework of the program "Pythagoras II" of the "Operational Program for Education and Initial Vocational Training" of the 3rd Community Support Framework of the Hellenic Ministry of Education.*

Timing Error Detection and Correction in Pipelines Constructed of Scan Flip-Flops

Andreas Floros Yiorgos Tsiatouhas Chrisovalantis Kavousianos
Dept. of Computer Science

University of Ioannina
Ioannina, Greece
{afloros, tsiatouhas,kabousia}@cs.uoi.gr

***Abstract*—Timing errors, due to environmental or operating conditions or due to timing faults that may escape fabrication tests, are a real concern in nanometer technology, high complexity and high frequency circuits. In this work a concurrent error detection and correction circuit and technique are presented that are applicable in pipeline architectures constructed of scan Flip-Flops. In addition a pipeline error recovery mechanism is illustrated. The proposed circuit is characterized by low silicon area requirements, compared to earlier approaches, and negligible penalty on performance.**

I. Introduction

As modern CMOS nanometer technologies scale down and the complexity of integrated circuits increases, an ongoing difficulty to achieve adequate reliability levels and keep the cost of testing within acceptable bounds is reported [1-2]. The device size scaling, the increase of the operating frequency and the power supply reduction affect circuits' noise margins and reliability. The probability of transient faults generation increases and many times it is hard to achieve the soft error rate (SER) specifications.

Timing related transient faults due to crosstalk, power supply disturbance or ground bounce are known mechanisms for timing error generation. Important problems also arise due to timing faults. The increased path delay deviations, due to process variations, and the manufacturing defects that affect circuit speed may result in timing errors that are not easily detectable (in terms of test cost) in high frequency and high device count ICs. Considering also the huge number of paths in modern circuit designs along with the complexity of testing, it is easy to realize that a significant number of defective ICs may pass the fabrication tests. Furthermore, modern systems running at multiple frequency and voltage levels may suffer from timing errors generated by numerous environmental and process related (and many times data dependent) variabilities that can affect circuit performance [3]. Consequently, concurrent testing techniques for timing error detection and correction are becoming mandatory in order to achieve acceptable levels of error robustness and meet reliability requirements.

Duplication and triplication techniques and self-checking design [4] are widely used to achieve systems reliability. In addition, soft and/or timing error detection techniques have been proposed in the open literature [5-8] that are based on the temporal nature of the transient faults or the delayed response of timing faults to provide error tolerance using time redundancy.

A pipeline architecture (named *Razor*) with timing error detection/correction for low power operation of systems exploiting dynamic voltage scaling has been introduced in [3]. According to this architecture for every system Flip-Flop in the design, an assistant shadow latch, a multiplexer (MUX) and a XOR gate as comparator are added. Recently, an advanced and low cost architecture (under the name *Blade*) with the same error detection/correction capabilities has been proposed in [9]. The new architecture requires only a multiplexer and a XOR gate per system Flip-Flop reducing drastically the silicon area cost.

In this work, we present a modified version of the Blade topology to be applied in pipeline architectures that are based on the use of scan Flip-Flops as system Flip-Flops (that is the majority of cases). The new topology provides concurrent timing error detection and correction capabilities with the minimum error detection latency and down to a single clock cycle penalty for error recovery. Moreover, this topology does not insert any delay in the critical paths of the circuit.

I. THE SCAN-BLADE BASED PIPELINE ARCHITECTURE

A. *Error detection and correction*

Fig. 1 illustrates a pipeline stage register constructed by standard scan Flip-Flops. When the *Scan_EN* signal is "high" the circuit is in the scan mode of operation, for testing purposes, and the scan Flip-Flops are fed by the *Scan_IN* inputs. Aiming to provide the circuit with the ability of concurrent timing error detection and correction the scan Blade Flip-Flop, to be used in the register of a pipeline stage, is proposed and presented in Fig. 2. The new Flip-Flop consists of the standard scan Flip-Flop plus a two input multiplexer (MUX) and a two input XOR gate. This extra hardware cost is very small compared to the Razor scheme where an additional shadow latch is used. The XOR gate directly compares the data at the *M* input and the *Q* output of the Main Flip-Flop for error detection, while the feedback from the *M* line to the input of the extra MUX forms the required memory element for error correction. The main characteristic and the advantage of the proposed topology is that we do not insert any circuitry in the critical path from the *D* input of the Flip-Flop to its *Q* output. Thus, the delay penalty is negligible and dedicated only to the small extra parasitic capacitances of the MUX input that is fed by the *M* line as well as this of the XOR input that is driven by the *Q* output.

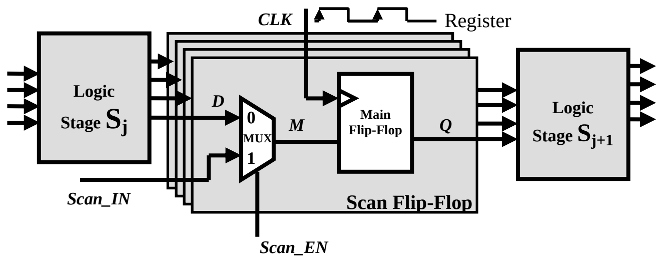


Figure 1. The standard scan Flip-Flop

The additional memory element that is required in the Blade topology is constructed by the two MUXs and the feedback path from the *M* line, as we mentioned earlier. Its memory state is activated by the *Capture* signal which in the error free case is controlled by the *Cap_CLK* signal, a delayed version of the clock signal *CLK* with a lower duty cycle. An OR gate is used to provide the register error indication signal *Error_R*$_j$ from the local error signals (*Error_L*) of the XOR gates in the Blade Flip-Flops of a register. Finally, the error indication signal *Error_R*$_j$ is captured in a single Flip-Flop (Error Flip-Flop) at the falling edge of the *Cap_CLK* signal. When the *Cap_CLK* signal is high the

Capture signal is activated (turns also to high) and the MUXs latch enters the memory state; else the MUXs latch is transparent. The time interval that the *Capture* signal is active must coincide with the time interval where the *D* inputs of the Blade Flip-Flops, in all stage registers, change values due to an earlier evaluation of the pertinent logic stages according to the circuit specifications. Any signal transition at the *D* inputs of the Blade Flip-Flops, outside this time interval, is considered as violation of the timing specifications and must be detected. Obviously, the deactivation of the *Capture* signal (its falling edge), and consequently of the *Cap_CLK* signal, must take place before the Main Flip-Flop's setup time plus the delay of the scan Flip-Flop MUX so that valid data are present at the inputs *M* of the Main Flip-Flops at the triggering edge of the clock *CLK*.

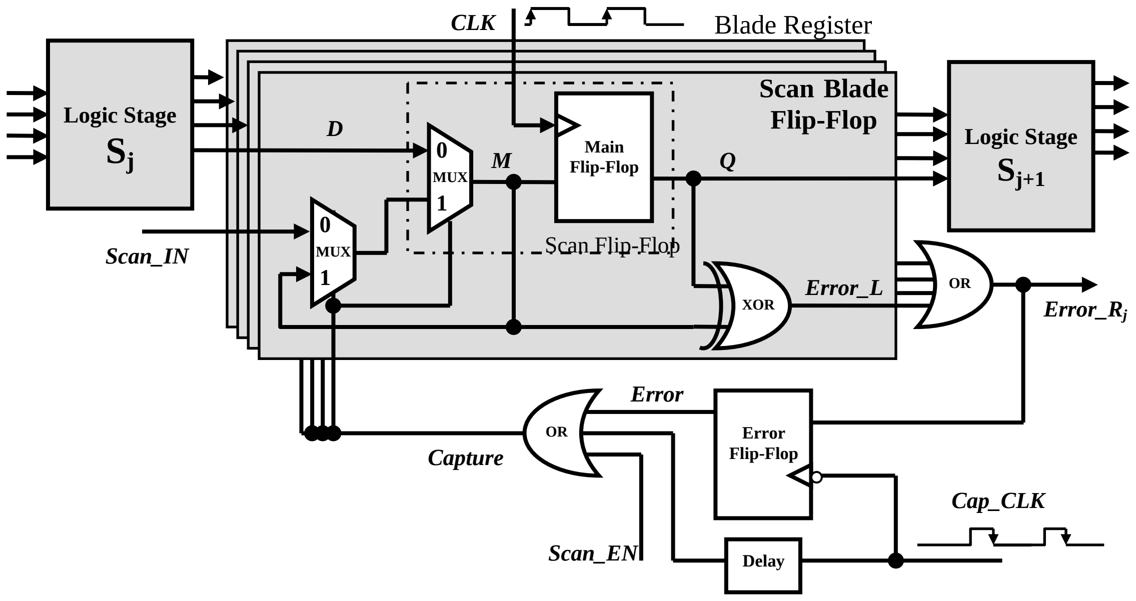


Figure 2. The Scan Blade Flip-Flop.

Note that the extra silicon area cost of the OR gate at the output of a Blade register is very small, one gate per register and especially when a Domino gate is used, while the Error Flip-Flop with the OR gate at its output is a one time cost for the whole pipeline and thus an insignificant cost.

In Fig. 3 the operation of the scan Blade Flip-Flop is presented. Since we refer to the normal mode of operation the *Scan_EN* signal is "low". In clock cycle i, the response of the logic stage S_j is within the timing specifications of the circuit. Consequently, after the triggering edge of the clock *CLK* both the data input *M* and the output *Q* of the Main Flip-Flop will carry the same value up to the falling edge of the *Capture* signal. Thus, the XOR output *Error_L* as well as the subsequent signal $Error\_R_j$ will be both zero at the triggering edge of the *Cap_CLK* signal on the Error Flip-Flop. In that case, the pipeline's operation remains unaltered (*Error*=low). In the next cycle i+1 a timing fault occurs which induce a delayed response of the stage S_j. Thus, a timing error is generated at the next triggering edge of the clock *CLK* and a transition occurs at the *D* input of a Blade Flip-Flop, inside cycle i+2, after this triggering edge and before the activation of the *Capture* signal. Since the MUXs latch is transparent during this time interval, this transition passes to the *M* line. In that case, the comparison by the XOR gate of the MUXs latch valid data with the erroneous data stored in the Main Flip-Flop turns the local error signal *Error_L* to “high” and generates a register error indication signal *Error_R$_j$*. Thus, the triggering (falling) edge of the *Cap_CLK* signal captures a “high” value at the output of the Error Flip-Flop. This “high” value extends the active duration of the *Capture* signal beyond this falling edge, keeping all MUXs latches in the memory state. At this point the error has been detected. In addition, all the MUXs latches hold the correct (valid) responses of the S_j logic state for the cycle i+1. The new responses of S_j at the cycle i+2 are blocked at the *D* inputs of the Blade Flip-Flops. Entering the next cycle i+3, the triggering edge of the clock *CLK* moves the valid data of the MUXs latch to the Main Flip-Flop and makes them available to the next pipeline stage S_{j+1}. Consequently, the error is corrected.

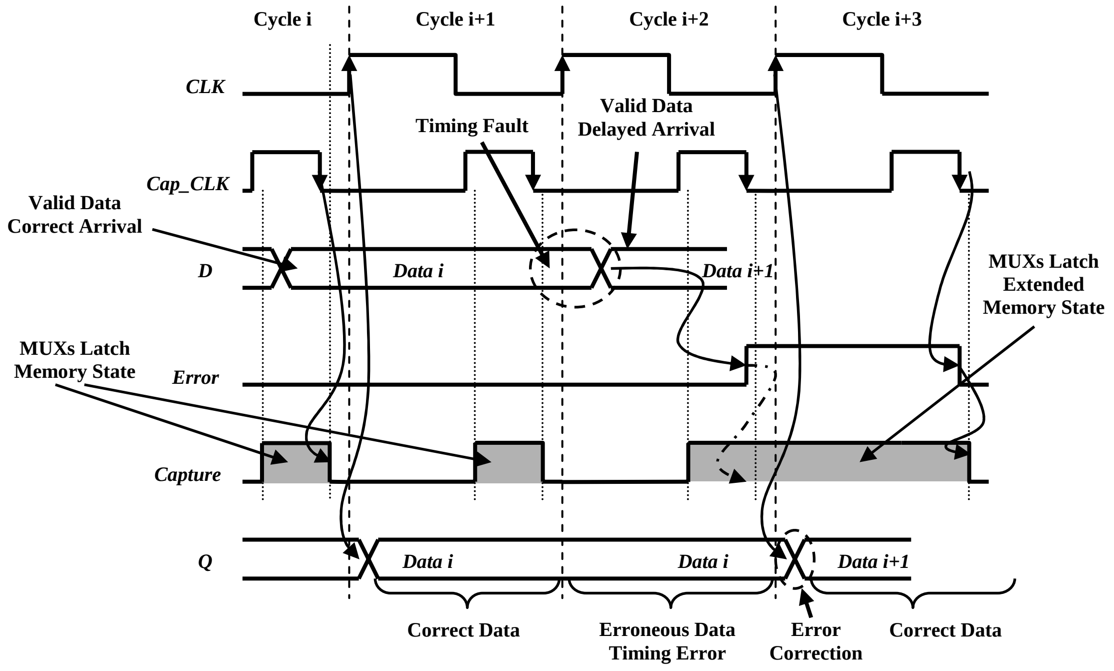


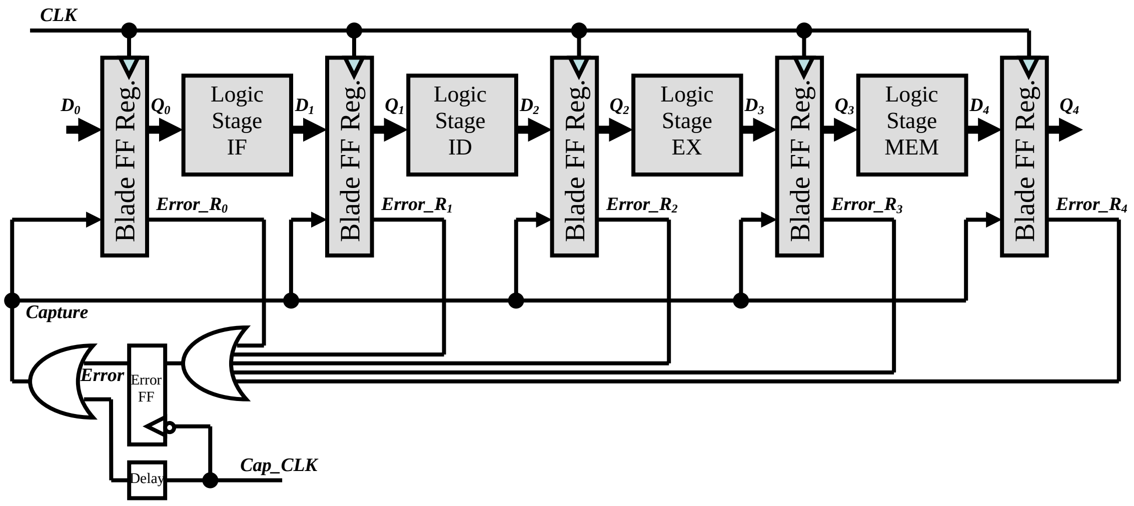
Figure 3. Blade Flip-Flop operation with a timing fault in cycle i+1 and recovery in cycle i+3.

According to the above analysis, if a timing error occurs in a pipeline stage S_j during a particular clock cycle, then the data in the subsequent stage S_{j+1} are incorrect, during the next clock cycle, and must be flushed from the pipeline. However, the MUXs latch contains the correct data without the need to re-execute the operation in the S_j stage. Thus, the S_{j+1} stage re-executes the operation using the correct input data with only one-cycle penalty.

B. *Pipeline recovery*

In case of error detection a pipeline state recovery action must follow. Fig. 4 illustrates the pipeline recovery mechanism. The event of a timing error in a logic stage (lets say ID stage) generates an error indication signal *Error_R2* at the following Blade register. This means that the results of the next stage are incorrect (as indicated in Fig. 4b) since its input data are not valid.

The error indication signal is latched by the Error Flip-Flop and the *Capture* signal remains high keeping all the MUXs latches of the Blade Flip-Flops in all stage registers in the memory state. Thus, in the next clock cycle every stage is allowed to re-compute its result using the correct data stored in the MUXs latches. Note here that there is no need for the failing stage to re-compute the response of the cycle where the failure occurred since the correct responses are already available in the following MUXs latches. The Blade Flip-Flop based pipeline architecture can tolerate any number of errors in a clock cycle since all stages re-evaluate their results with correct data at their inputs. In case that one or more stages fail in each clock cycle, the pipeline will continue to run at half of the normal speed.



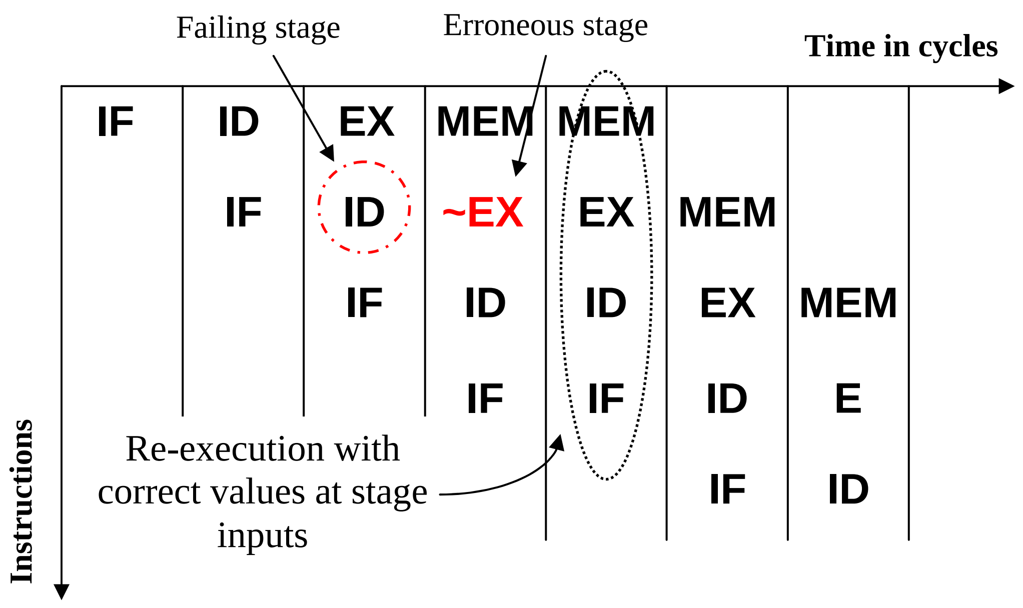


Figure 4. a) Pipeline organization and b) Pipeline recovery.

Referring to the analysis of the Blade architecture, there is no need to apply main clock gating to accomplish pipeline recovery. Moreover, although the counterflow pipeline approach [10] can be also applied in the Blade architecture as in the Razor case, there is no need to proceed with it. This is due to the fact that the pipeline performance is not affected by the recovery mechanism since there is not any prohibitive delay in the feedback path from the error indication signal generation to the redirection of the MUXs' inputs. The MUXs latches in the Blade Flip-Flops of every register are set, by the *Capture* signal, to the memory state independently of the error signal generation. Thus, at the time an error indication signal (*Error*="high") is latched in the Error Flip-Flop, the *Capture* signal is already active ("high") keeping the MUXs latches in the memory state. The error indication signal simply extends the active state of the *Capture* signal until the next triggering (falling) edge of the *Cap_CLK* signal, that is for a time duration equal to a clock period. Consequently, the following triggering edge of the clock *CLK* injects the correct data from the MUXs latch into the pipeline, allowing the "swerved" instruction to continue. Later instructions inside the pipeline are not flushed and continue to run after recovery. Hence, only a single cycle is required in the Blade architecture for pipeline recovery as it is shown in Fig. 4b.

Note that the use of the delayed *Cap_CLK* signal may result in the corruption of the data in the MUXs latch due to short paths in the combinational logic. To prevent data corruption a minimum path delay constraint is considered in the design. In order to meet this constraint, buffers may be added (like in the Razor case) and/or gates constructed of minimum size plus high-threshold voltage transistors may be used during logic synthesis to slow down fast paths. The minimum path delay constraint is equal to the delay of the *Capture* signal, with respect to the system clock *CLK*, plus the hold time of the MUXs latch. However, a trade-off arises. A large value for the minimum path delay constraint may increase the number of the required buffers in the design and consequently the silicon area penalty. On the other side, a small value for this delay constraint reduces the error tolerance due to the reduction of the maximum detectable signal delay.

II. CONCLUSIONS

In this work we present a new scan Flip-Flop that incorporates timing error detection/correction capabilities. In addition a pipeline architecture is discussed that exploits this scan Flip-Flop for pipeline recovery after a timing error occurrence. This approach is characterized by low silicon area requirements, negligible performance penalty and the minimum cost, of only one clock cycle, for pipeline recovery. Although the proposed technique has been illustrated for pipeline architectures, it can be applied in general to any sequential circuit. The proposed topology can be used to support architectures that exploit dynamic voltage scaling for low power operation as in [3].

REFERENCES

[1] R. Wilson and D. Lammers, "Soft Errors Become Hard Truth for Logic," *EE Times*, 3 May 2004 (available at http://www.eetimes.com/ news/latest/showArticle.jhtml?articleID=19400052).

[2] S. Mitra, N. Seifert, M. Zhang, Q. Shi and K. S. Kim, "Robust System Design with Built-In Soft-Error Resilience," *IEEE Computer*, vol. 38, no. 2, pp. 43–52, 2005.

[3] T. Austin, D. Blaauw, T. Mudge and K. Flautner, "Making Typical Silicon Matter with Razor," *IEEE Computer*, vol. 37, no. 3, pp. 57–65, 2004.

[4] M. Nicolaidis and Y. Zorian, "On-Line Testing for VLSI – A Compendium of Approaches," *Journal of Electronic Testing: Theory and Applications*, vol. 12, no. 1-2, pp. 7-20, 1998.

[5] C. Metra, R. Degiampietro, M. Favalli and B. Ricco, "Concurrent Detection and Diagnosis Scheme for Transient, Delay and Crosstalk Faults," *5th IEEE On-Line Testing Workshop*, pp. 66-70, 1999.

[6] M. Nicolaidis, "Time Redundancy Based Soft-Error Tolerance to Rescue Nanometer Technologies," *VLSI Test Symp.*, pp. 86-94, 1999.

[7] L. Anghel and M. Nicolaidis, "Cost Reduction and Evaluation of Temporary Faults Detecting Technique," *Design Automation and Test in Europe Conference*, pp. 591-598, 2000.

[8] S. Matakias, Y. Tsiatouhas, A. Arapoyanni, and Th. Haniotakis, "A Circuit for Concurrent Detection of Soft and Timing Errors in Digital CMOS ICs," *Journal of Electronic Testing: Theory and Applications*, vol. 20, no. 5, pp. 523-531, 2004.

[9] A. Floros, Y. Tsiatouhas, A. Arapoyanni and Th. Haniotakis, “A Pipeline Architecture Incorporating a Low-Cost Error Detection and Correction Mechanism”, *IEEE Int. Conf. on Electronics Circuits and Systems*, pp. 692-695, 2006.

[10] R.F. Sproull, I.E. Sutherland and C.E. Molnar, “The Counterflow Pipeline Processor Architecture,” *IEEE Design and Test of Computers*, vol. 11, no. 4, pp. 48-59, 1994.