2014 20th IEEE International Symposium on Asynchronous Circuits and Systems

Metastability in Better-Than-Worst-Case Designs

Salomon Beer* Marco Cannizzaro†, Jordi Cortadella‡, Ran Ginosar* and Luciano Lavagno†

*Electrical Engineering Department
Technion, Israel

†Dipartimento di Elettronica
Politecnico di Torino, Italy

‡Department of Software
Universitat Politecnica de Catalunya, Spain

Abstract

Better-Than-Worst-Case-Designs use timing speculation to run with a cycle period faster than the one required for worst-case conditions. This speculation may produce timing violations and metastability that result in failures and non-deterministic timing behavior. The effects of these phenomena are not always well understood by designers and researchers in this area. This paper analyzes the impact of timing speculation and the reasons why it is difficult to adopt this paradigm in industrial designs.

I. Introduction

In the last few years there has been a proliferation of research around a paradigm that has been often referred to as *Better-Than-Worst-Case Design*, *Resilient Circuits* or *Timing Speculation*. The first work following this direction was Razor [6], which led to other variations along the same line [8], [5], [1].

Circuits usually run at a clock frequency that covers all potential delay variations under any possible operating condition. However, a significant part of the cycle period is often wasted due to the conservative margins to cover these variations. The strategy consists of running at a faster frequency, at the expense of sporadically experiencing runtime errors as a consequence of the setup and hold time violations. These errors are corrected with the aid of special circuitry. Hence the system can achieve better performance by operating at nominal rather than worst-case conditions. The strategy can be considered as an aggressive over-clocking [3], with a robust management of timing failures.

The basic scheme for error detection and recovery is the use of a pair of flip-flops (or latches) to capture the logic signals at the end of the critical paths twice: first by the main flip-flop after a clock period, next by the second flip-flop some fraction of the clock cycle later (to account for worst-case operation). If the captured values are identical, no error occurred and the computation proceeds using the "speculated" value sampled in the main flip-flop. If a mismatch is detected, the circuit is stopped and restarted from the correct value, which is stored in the secondary sampling element.

II. Metastability

Very often, there has been a misunderstanding about the behavior of the main flip-flop when a timing violation is produced. This issue is not a minor aspect of the approach since it has led to the proposal of incorrect mechanisms. Some of the common mistakes come from the lack of knowledge about the metastability phenomenon [7], with wrong assumptions like these:

- A setup violation means that the logic value at the output of the flip-flop is unknown, but is either a 0 or a 1.
- Metastability at the flip-flop is resolved in negligible time.
- Metastability is resolved by putting two flip-flops after the signal.

By disregarding metastability, simple approaches for error recovery may be devised (e.g., 1-cycle recovery procedures) thus leading to overly optimistic conclusions about the benefits of timing speculation [4], [9].

Metastability of a signal is an unstable state in which the signal is neither a logic 0 nor a logic 1. The resolution time is the time the signal takes to settle into a stable state (either 0 or 1). Known facts about metastability are:

- The resolution time is unpredictable and unbounded. Hence, no assumptions can be made about the required time to fully settle a metastable signal.
- The resolution time can be modeled as a random point process with a probability distribution that decreases exponentially with the allowed settling time.

This second fact is cardinal for the design of circuits that must live with metastability. It indicates that metastability can be resolved "in practice" if the circuit waits for long enough. The typical concept to measure the practicality of the waiting time is the *Mean Time Between Failures* (MTBF), that depends, among others, on two behavioral parameters: frequency of the clock and switching probability of the data. Typically, a system is considered to be robust when MTBF is several years (e.g., more than one hundred years).

A common rule of thumb to deal with metastability is the recommendation to put a few flip-flops after a signal to settle its value. This chain of flops is usually known as a *synchronizer*. This rule has often contributed to create another misunderstanding: a chain of flops helps to resolve metastability. Again, this is a wrong assumption. What contributes to resolve metastability is the waiting time that lets the signal to settle. In current technologies, the waiting time to achieve an acceptable MTBF can be 2, 3, 4 cycles or more (depending on the technology and the clock and data frequency). The chain of flops is used as a mechanism to *pipeline* the signal propagation while the circuit is waiting for metastability to be resolved. The number of flops is determined by the number of cycles required to achieve the desired MTBF.

III. Simple analysis of Razor

Figure 1 depicts a simple scheme similar to the one proposed in [6]. For simplicity in the analysis, let us assume that d_1 and d_2 are chains of non-inverting buffers, d_1 is one of the critical paths of the circuit and d_2 is much shorter than d_1. Since d_1 is critical, a Razor flip-flop is used between d_1 and d_2. The Razor flip-flop contains an auxiliary flip-flop that captures the incoming data with a delayed clock. Hence, *Qaux* has the correct value that should be captured in Q in the absence of error. The XOR gate compares Q and *Qaux* and generates signal E that indicates whether an error has occurred.

1522-8681/14 $31.00 © 2014 IEEE
DOI 10.1109/ASYNC.2014.21

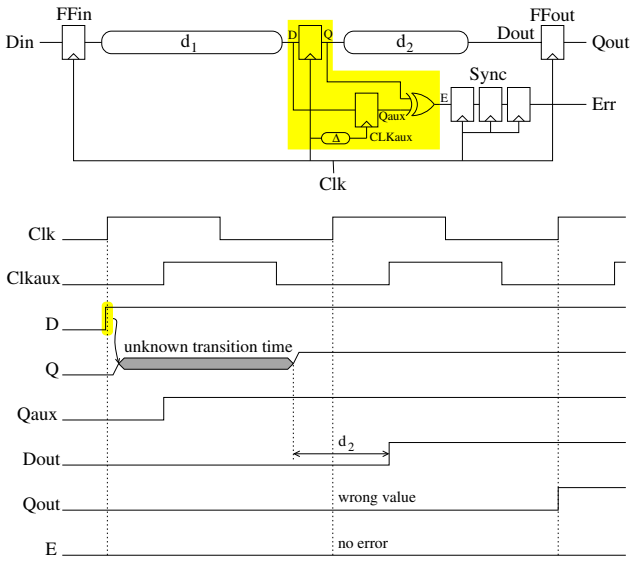

Fig. 1. Basic Razor scheme and timing diagram with a false positive.

Two principal facts must be understood about the behavior of the previous scheme:

- Signal Q may transition at an unknown time when signal D changes around the rising edge of the clock due to some internal metastability in the flip-flop. This non-deterministic value may produce metastability in the flip-flop after signal E. For this reason, a certain time must elapse for signal E to settle. This is the reason why a synchronizer is included between E and *Err*.
- Metastability produces time non-determinism, i.e., no assumptions can be made about the propagation time to *Dout* across delay d_2 since the delay of Q is unknown. The only reliable assumption is that *Dout* becomes stable d_2 time units after Q becomes stable.

These two facts have serious implications on the technical viability of this scheme.

On the one hand, signal *Err* is read n cycles after the error has occurred, where n is the number of flops of the synchronizer. This implies that the effect of an erroneous calculation is propagated during n cycles across the circuit. To rescue the circuit from this catastrophic effect, the execution must be rolled back for n cycles. Thus the scheme has only been proposed for advanced microprocessors with mechanisms to restart the pipeline at some previous checkpoint.

On the other hand, the non-deterministic timing of metastable signals may lead to *false positives*. To illustrate this phenomenon, we will resort to the timing diagram in Fig. 1. We observe that the first rising edge of *Clk* captures signal D with a setup violation. This may produce a late switching on signal Q. In parallel, *Qaux* captures the correct data value. Let us assume that metastability is resolved some time before the next rising edge of *Clk* and the resulting value is the correct one. In this scenario, signal E will report "no error". However, the propagation from Q to *Dout* may occur later than expected and *FFout* may capture an incorrect value. Therefore, signal *Err* will not report the error on *Qout*.

Other variants have been proposed to have more robust schemes when dealing with metastability. In [8], [2], the main flip-flops were substituted by latches to avoid the propagation of metastability in the datapath. In [8], timing violations were identified by a transition detector that detects whether a transition in the D signal occurs after the rising edge of the clock. The transition detector may also suffer metastability. In [2], errors are detected with an auxiliary flip-flop that captures the speculated value and compares with the value at the main latch.

None of the previous approaches describes a formal timing analysis procedure that can be used to guarantee timing correctness with the presence of metastability. In fact, it is widely believed that this is totally impossible within the synchronous domain (only its probability of occurrence can be reduced). The authors of [2] proposed the safest approach, which confines meta-stability to the control path, and hence is immune to the undetected timing errors described above, that plague the other Razor-like approaches. However, even they can only claim to extend the MTBF to a value that is comparable to that due to Single-Event Upsets in traditional synchronous circuits.

IV. Conclusions

Metastability is a phenomenon that cannot be neglected when using timing speculation. The need for synchronization and the timing non-determinism introduced while metastability is resolved are two important aspects that must be considered. In fact, metastability failures are much more critical for Better-Than-Worst-Case-Designs than for standard synchronous designs, because they are deliberately driven towards synchronization failures. Their control is constantly striving to compute with some errors, and some correct results, and hence is bound to hit the "gray" metastability zone that is unavoidably present between these two answers. We believe these are strong reasons why industrial exploitation has not succeeded for those schemes using timing speculation.

References

[1] K. A. Bowman, J. W. Tschanz, N. S. Kim, J. C. Lee, C. B. Wilkerson, S. Lu, T. Karnik, and V. De. Energy-efficient and metastability-immune timing-error detection and instruction-replay-based recovery circuits for dynamic-variation tolerance. In *IEEE ISSCC*, pages 402–403, 2008.

[2] Keith A. Bowman, James W. Tschanz, Shih-Lien L. Lu, Paolo A. Aseron, Muhammad M. Khellah, Arijit Raychowdhury, Bibiche M. Geuskens, Carlos Tokunaga, Chris B. Wilkerson, Tanay Karnik, and Vivek K. De. A 45 nm resilient microprocessor core for dynamic variation tolerance. *IEEE Journal of Solid-State Circuits*, 46(1):194–208, January 2011.

[3] B. Colwell. The zen of overclocking. *Computer*, 37(3):9–12, Mar 2004.

[4] J. Cong and K. Minkovich. Logic synthesis for better than worst-case designs. In *Int. Symp. on VLSI Design, Automation and Test (VLSI-DAT)*, pages 166–169, April 2009.

[5] S. Das, D. Roberts, Seokwoo Lee, S. Pant, D. Blaauw, T. Austin, K. Flautner, and T. T. Mudge. A self-tuning DVS processor using delay-error detection and correction. *Solid-State Circuits, IEEE*, 41:792–804, 2006.

[6] D. Ernst, S. Nam Sung Kim, Das, S. Pant, R. Rao, Toan Pham, C. Zieslera, D. Blaauw, T. Austin, K. Flautner, and T. Mudge. Razor: a low-power pipeline based on circuit-level timing speculation. In *IEEE Int. Symp. Microarchitecture (MICRO-36)*, pages 7–18, 2003.

[7] R. Ginosar. Metastability and synchronizers: A tutorial. *Design Test of Computers, IEEE*, 28:23–35, 2011.

[8] S.Das, C. Tokunaga, S. Pant, W.-H. Ma, S. Kalaiselvan, K. Lai, D.M. Bull, and D.T. Blaauw. RazorII: In situ error detection and correction for PVT and SER tolerance. In *IEEE J. Solid-State Circuits*, pages 32–48, 2009.

[9] Yinan Sun, Yongpan Liu, Xiaohan Wang, Hongliang Xu, and Huazhong Yang. Design methodology of multistage time-domain logic speculation circuits. In *IEEE Int. Symp. on Circuits and Systems (ISCAS)*, pages 1944–1947, May 2011.