An Automated Design and Verification Flow for Fault-Tolerant ASICs

Anselm Breitenreiter and Milos Krstic
IHP, Im Technologiepark 25, 15236 Frankfurt (Oder), Germany
{breitenreiter|krstic}@ihp-microelectronics.com

***Abstract*—The paper proposes an automated flow for the design and verification of fault-tolerant ASICs. This should include the analysis of fault-sensitivity of a given circuit, the implementation of fault-tolerance mechanisms and the verification of the obtained fault-tolerance level.**

I. Introduction

With decreasing feature size of transistors soft and hard errors not only play a role in space but also in terrestrial applications. This increases the demand for fault-tolerant (FT) circuits. To face current challenges in VLSI design, there is a range of additional uniform flows as for example static timing analysis (STA), design for testability (DFT) or low-power implementation. For fault-tolerance there is no design and verification flow available and designs rather rely on individual solutions.

This work aims at the investigation of a universal flow and the respective methods for the design and verification of fault-tolerant systems. The FT flow should have three major aspects:

1) Analysis of fault-sensitivity
2) Implementation of fault-tolerance mechanisms
3) Verification of the obtained fault-tolerance level

In the following these steps are described in more detail.

II. Analysis of fault-sensitivity

First the design needs to be analyzed for all possible faults and their effects on the system. This could be done on the post-synthesis netlist by either analytic methods [1]–[3] or by simulation [4], where the procedure is very similar to the verification of fault-tolerance. The effect of a fault can be investigated in two different ways. On the one hand on the logic layer, which determines if the fault results in an error, if that error propagates to the outputs and if the system automatically returns to a correct state. This information is inherent to the system. On the other hand there is the functional layer, which determines up to what extent an erroneous system can still fulfill the intended function and produce a useful output. This information is non-inherent to the system and needs to be specified. For this *fault-tolerance constraints* (FTC) are foreseen. The result of this step is a list of all possible faults with a rating for their logic as well as their functional effect.

III. Implementation of fault-tolerance mechanisms

The following step should be the implementation of fault-tolerance. Based on the results from the analysis and the FTC, parts of the system should be selected to insert certain mechanisms of fault-tolerance. The selection of system components to be protected should be an automated process, which aims at a minimum of additional costs while guaranteeing the correct operation of the system under the specified conditions. The implementation of fault-tolerance can be done for instance by structural redundancy, by replacing of cells with their fault-tolerant equivalents or by adding cells [5]. The flow should offer an open interface for the integration of additional fault-tolerance insertion mechanisms. The result is a netlist of the fault-tolerant system.

Table I
Cost of fault-tolerance for a space Ethernet PHY

Implementation	Area in mm^2	Area in %
Normal	1.899	100
Fault-tolerant	3.445	181

IV. Verification of fault-tolerance

Finally, it needs be to verified again that the system still meets the previous functional and non-functional (e.g. area, power, timing) requirements and that the required level of fault-tolerance was reached.

The verification of fault-tolerance should be done by fault-injection in simulation, where it should be compatible to UVM [6] to seamlessly integrate into a modern verification process.

V. Design case: Space Ethernet PHY

The benchmark circuit for this flow is an Ethernet PHY for space applications. Table I shows a comparison of the area costs of the non-fault-tolerant and fault-tolerant implementation. Fault-tolerance is achieved on gate-level by using only fault-tolerant sequential cells. This results in 181% of the original area, which illustrates the wide space for solutions which compromise between full and no protection, while guaranteeing the correct system operation under the specified conditions.

VI. Conclusion

This paper presents the idea of a uniform and automated flow for the design and verification of fault-tolerant ASICs. It should employ formal methods as well as simulation. The goal is, outgoing from this early introductory work, the

establishment of a methodology which could be easily added to the standard digital design flow.

ACKNOWLEDGEMENT

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 64024.

REFERENCES

[1] S. Weidling, "Neue ansätze zur verbesserung der fehlertoleranz gegenüber transienten fehlern in sequentiellen schaltungen," Ph.D. dissertation, University of Potsdam, Germany, 2016.

[2] L. Chen, M. Ebrahimi, and M. B. Tahoori, "Cep: correlated error propagation for hierarchical soft error analysis," *Journal of Electronic Testing*, vol. 29, no. 2, pp. 143–158, 2013.

[3] S. S. Mukherjee, C. Weaver, J. Emer, S. K. Reinhardt, and T. Austin, "A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor," in *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*. IEEE, 2003, pp. 29–40.

[4] A. Simevski, R. Kraemer, and M. Krstic, "Automated integration of fault injection into the asic design flow," in *2013 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*, Oct 2013, pp. 255–260.

[5] M. Nicolaidis, "Design for soft error mitigation," *IEEE Transactions on Device and Materials Reliability*, vol. 5, no. 3, pp. 405–418, 2005.

[6] *Universal Verification Methodology (UVM) 1.2 User's Guide*, Accellera Systems Initiative (Accellera), 2015.