Evaluating a Low Cost Robustness Improvement in SRAM-based FPGAs

M. Ben Jrad, R. Leveugle
TIMA Laboratory (Grenoble INP, UJF, CNRS)
46 Avenue Félix Viallet - 38031 Grenoble Cedex - FRANCE
Mohamed.Ben-Jrad@imag.fr, Regis.Leveugle@imag.fr

***Abstract*— Soft errors in the configuration memory of SRAM-based FPGAs cause significant application disturbances. We demonstrate on Xilinx and Altera FPGAs the feasibility of a very low cost and automated mitigation approach and we evaluate its efficiency.**

Keywords—dependability; SRAM-based FPGA; soft errors; multiple errors

I. INTRODUCTION

An increasing number of applications rely on SRAM-based FPGAs but the large size of the configuration memory makes these devices very sensitive to soft errors. Soft errors in the configuration memory may modify the function of the circuit and remain until a reconfiguration of the device; they must therefore be mitigated to achieve the required reliability or availability level of many applications. We focus in this paper on such configuration errors. Also, many errors in the interconnection configuration have no effect on the application; we therefore focus on errors in the implemented logic functions, i.e. in the Look-Up Tables (LUT).

Many works have been devoted in the last decade to this problem. The state-of-the-art summary will be omitted due to the length restrictions. However, the focus is in general on how protecting the system against all possible soft errors, accepting very large overheads to achieve this goal. In this paper, we evaluate an approach based on the maximum exploitation of unused resources to increase the design robustness at no (or low) cost. The approach is therefore not intended for high-reliability or high-availability applications, but for cost-sensitive applications seeking the best dependability. The basic principles of the approach and a first case study on an AT40K target were presented in [1]. The main contributions of this paper are to demonstrate that the approach can be exploited on other FPGA targets (Xilinx Virtex V and Altera Stratix IV) and to report results on a set of benchmarks, thanks to automated procedures.

Section II briefly recalls the basics of the proposed approach and presents the automated design environments developed for the Xilinx and Altera targets. Results are presented and discussed in Section III.

II. EVALUATED APPROACH

The approach is based on inserting, at design time, a limited amount of redundancy so that effective errors occurring in the configuration can be detected and a signal can be sent to a system-level controller. The error detection is based on local duplication of selected LUTs, so that only logic resources that would actually not be used in the initial design are exploited. The comparisons and the combination of local error signals require extra LUTs but this extra logic is not necessarily an actual overhead if, due to the discrete size of the devices, enough logic resources were wasted in the initial implementation of the protected design. The approach can detect single-bit errors (SEUs) in the configuration memory but also many multiple-bit errors.

LUTs in Virtex V components have 6 inputs, but are in fact composed of two 5-input LUTs with connected inputs. In many cases (see column "Protected" in Table I), only one 5-input LUT is actually used after place and route. The idea is therefore to use the second 5-input LUT as a free replica of the function.

In the Stratix IV devices, combinatorial functions are implemented in blocks called ALMs (Adaptive Logic Modules). Each ALM has 8 inputs with a "fracturable" LUT that can be divided into two adaptive LUTs (ALUTs). In spite of the flexibility of this architecture, we have observed on our benchmarks that on an average 35% of the available resources remain unused, because no sub-function with the same inputs can be implemented for the original design in a partially-used LUT. In this case, the fracturability can be used to implement the duplica since the same inputs are required.

The generic design flow is illustrated in Figure 1 and is common to all targets, although the tools actually used at each step depend on the selected FPGA family. A specific tool has been developed to automate the insertion of the redundant logic and the generation of error signals, taking into account the constraints related to both architectures. In addition, this

This work is supported by French research funds in the frame of the CATRENE OPTIMISE project..

tool generates placement and routing constraints to optimize the implementation on the selected target.

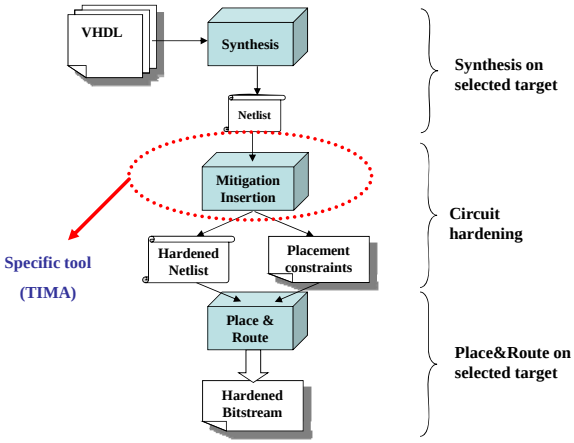

Figure 1. Generic design flow

III. Implementation Results

Experiments presented here have been carried out on ITC'99 benchmarks. Implementation results on Virtex V are summarized in Table I. Additional resources are only due to the error signal generation. In many cases, the percentage of additional resources is less than 50% of the coverage of single and multiple errors in LUTs.

TABLE I. Implementation Results in Virtex V Platforms

Benchmark	Protected LUTs	Add. Res.
b01	90%	40%
b02	100%	50%
b03	57.89%	26.32%
b04	72.81%	29.82%
b05	47.83%	19.57%
b06	100%	50%
b07	60.61%	25.25%
b08	47.83%	21.74%
b09	83.67%	32.65%
b10	50%	23.68%
b11	55%	24 %
b12	36.86%	15.29%
b13	71.43%	30.61%
b14	51.85%	20.84%

Implementation results on Stratix IV are summarized in Table II. The number of ALMs obtained for each benchmark after synthesis is recorded, as well as the percentage of ALUTs that can be duplicated without actual overhead (column "Coverage"). Additional resources are recorded after placement and routing, with or without placement constraints. The results show that the placement constraints allow noticeable savings in most cases. b03 can be protected at more than 80% without any overheads (with or without placement constraints); this demonstrates the feasibility, in some cases, to significantly increase the intrinsic robustness of an application for free (including the alarm generation). But unfortunately, on an average, the ratio between the coverage and the overheads is less than the one obtained on Virtex V in spite of the stronger placement and routing constraints that had to be generated for this target.

TABLE II. Implementation Results in Altera Stratix IV Platforms

Bench.	Initial number of ALMs	Coverage	Add. Res.	
			Without P&R const.	With P&R const.
b01	9	70%	33.33%	22.22%
b02	7	87.50%	42.86%	42.86%
b03	17	81.82%	0%	0%
b04	53	26.58%	35.85%	24.53%
b05	74	19.33%	20.27%	13.51%
b06	8	36.36%	25%	25%
b07	36	5.26%	8.33%	0%
b08	14	40%	42.86%	28.57%
b09	20	52.63%	25%	20%
b10	19	25.81%	26.32%	21.05%
b11	59	66.18%	32.20%	32.20%
b12	163	14.77%	12.88%	9.82%
b13	33	31.25%	33.33%	24.24%
b14	528	22.19%	22.73%	13.64%

IV. Conclusion

The results presented in this paper demonstrate the feasibility to improve with low (or no) overheads the robustness of an application implemented on SRAM-based FPGAs by taking advantage of unused resources available after placement and routing of the design. The proposed approach, initially evaluated on an AT40K platform, has proved to be general enough to be ported on Xilinx and Altera platforms. Furthermore, we have demonstrated that the approach can be automated and has therefore the potential to be exploited in a standard design flow. Experiments also showed that some platforms (or their associated development tools) lead to more difficult implementations of the approach. In particular, many constraints had to be taken into account to succeed in applying the approach to Stratix FPGAs. In addition, the results are less interesting than on other platforms.

References

[1] J. B. Ferron, L. Anghel, R. Leveugle, "Towards low-cost soft error mitigation in SRAM-based FPGAs: a case study on AT40K", 3rd IEEE Latin American Symposium on Circuits and Systems (LASCAS), Playa del Carmen, Mexico, 2012