Is aproximate computing suitable for selective hardening of arithmetic circuits?

Bastien Deveautour, Arnaud Virazel, Patrick Girard, Serge Pravossoudovitch, Valentin Gherman

2018 13th International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS)

Is Aproximate Computing Suitable for Selective Hardening of Arithmetic Circuits?

B. Deveautour, A. Virazel, P. Girard,
S. Pravossoudovitch
LIRMM, Univ. Montpellier, CNRS
161 rue Ada, 34000 Montpellier, France
Email: {deveautour, virazel, girard, pravo}@lirmm.fr

V. Gherman
CEA – LIST, Laboratoire Fiabilité et Intégration Capteurs
PC 172, 91191 Gif sur Yvette, France
Email: valentin.gherman@cea.fr

***Abstract*—Selecting the ideal trade-off between reliability and cost associated with a fault tolerant architecture generally involves an extensive design space exploration. In this paper, we address the problem of selective hardening of arithmetic circuits by considering a duplication/comparison scheme as error detection architecture. Different duplication scenarios have been investigated: i) a full duplication, ii) a reduced duplication based on a structural susceptibility analysis, iii) a reduced duplication based on the logical weight of the arithmetic circuit outputs and iv) a reduced duplication based on an approximated structure from a public benchmark suite. Experimental results performed on adder and multiplier case study circuits demonstrate the interest of using approximate circuits to improve the mean time to failure while keeping a low area and power overhead and reduced error probability and error magnitude.**

Keywords—Arithmetic circuit; fault tolerance; selective hardening, error detection, duplication scheme, approximate computing.

I. Introduction

From the safety point of view, selecting the ideal trade-off between reliability improvement and cost associated with a fault tolerant architecture employed for hardening mainly depends on the critical level of the application, the environment radiation levels and the technology used. It is important to create a balance that best suits the design cost- budget and the acceptable error rate constraint [1]. Selective hardening is a technique that creates this balance by allowing the design to move between the two edge cases of no- hardening and fully hardened, and optimizes it by selecting the most sensitive circuit parts to be hardened. This provides improvement in error rate at an acceptable area and power overheads [2].

Selective hardening is done in two steps. Firstly, the most sensitive circuit parts are identified based on their contribution to the overall circuit error rate. Secondly, a fault tolerance scheme is applied on these selected circuit parts [3]. Most of the approaches of selective hardening in the literature focus on improving the vulnerability analysis methodology and use existing fault tolerant architectures for hardening. Thus, they are better classified on the basis of the former criteria. Circuit-level vulnerability estimation methods generally consider the three masking effects (logical, electrical and latching-window) that prevent transient pulses from getting latched in Flip-Flops (FFs). The use of accurate models that consider all the three masking effects is impractical because of the immense amount of computational effort needed to simulate or to deal with these models. Therefore, some techniques like [4] and [5] rely on approximate abstract models while considering all three masking effects, whereas others like [1], [6], [7] and [8] resort to only one or two of them to identify circuit elements with the highest impact on soft error rate.

Recently, we have proposed a very fast reliability analysis methodology for logic circuits, called structural susceptibility analysis, that helps selecting the best candidates and identifying the degree of hardening necessary to respect the design cost (in terms of area and power) and soft error reliability constraints [9]. Based on this structural analysis we have also proposed a selective hardening technique using the Hybrid Transient Fault Tolerant (HyTFT) architecture presented in [10]. This selective hardening approach reduces the number of Combinational Logic (CL) output nodes to be compared with a full version of the circuit for error detection in a vulnerability-aware manner. Reducing the number of comparison points not only reduces the size of the comparator but also significantly reduces the size of the duplicated CL copy since only the logic cones responsible for generating the selected outputs need to be synthesized to create this partially duplicated copy.

Although effective in reducing area and power consumption compared to a full duplication, this selective hardening technique is not suitable for arithmetic circuits since it does not consider any error metrics when it is applied. In this paper, we analyze in details the impact of this selective hardening technique on error metrics used in the Approximate Computing (AxC) context: Worst Case Error (WCE) and Error Probability (EP) [12]. Then, we compare different duplication scenarios to build an error detection architecture for arithmetic circuits: i) a full duplication scheme, ii) a reduced duplication scheme based on the structural susceptibility analysis presented in [9], iii) a reduced duplication scheme based on the logical weight of the arithmetic circuit outputs and iv) a reduced duplication scheme based on an approximated structure from a public benchmark suite [11] which is composed of arithmetic circuits. Experimental results on two case studies (adder and

This work has been partially funded by the French government under the framework of the PENTA HADES ("Hierarchy-Aware and secure embedded test infrastructure for Dependability and performance Enhancement of integrated Systems") European project.

multiplier) demonstrate the interest of using approximate structures as duplication scheme since both area overhead and power consumption are reduced compared to a full duplication scheme, while maintaining good levels on error metrics.

The paper is organized as follows. Section II surveys our previous works related to selective logic hardening. Section III presents the different scenarios we consider to selectively harden arithmetic circuits. Experimental results are discussed in Section IV. Finally, conclusions are given in Section V.

II. Previous Work on Selective Logic Hardening

The structural susceptibility analysis methodology proposed in [9] is based on the fact that not all the outputs of a CL block have the same susceptibility to Single Event Transient (SET) effects and assumes that their susceptibility is a function of the number of nodes in their fan-in logic cone. It exploits the structural properties of the output fan-in cone to get their relative susceptibility estimates. The outputs are ranked on the basis of their relative susceptibility and the best candidates are selected for error detection. The number of output candidates selected defines the reliability improvement and cost trade-off and the cost-aware selection allows us to optimize this trade- off.

Algorithm 1 shows the pseudo-code of the susceptibility analysis. The algorithm starts by reading the pre-place-and-route netlist of the design. Then it forms groups F_j of all fan-in cells for each CL output O_j. Once the groups are formed the weight W_j of each fan-in cone is calculated by adding together the weights of all the cells in the corresponding fan-in cone group. According to the hypothesis that forms the basis of this method, cell weight is the number of inputs and outputs of that cell. Ranks are assigned to each output on the basis of their fan-in cone weight using a sort function shown in line 15 of Algorithm 1.

```
1  read(netlist);
2  // Group all fan-in cone cells together for each
      output node
3  foreach $O_j$ do
4      $F_j \leftarrow O_j$.get_fanin();
5  end
6  // Get weight of fanin cone of each output
7  foreach $O_j$ do
8      foreach $C_i$ do
9          if $C_i \in F_j$ then
10             $W_j \leftarrow W_j + C_i$.get_pins();
11         end
12     end
13 end
14 // Sort outputs on the basis of their fanin cone
      weight
15 sort($O_j$,$F_j$,$W_j$);
```

Algorithm 1. Structural susceptibility analysis

The algorithm is further explained by its application to a simple example circuit shown in Figure 1. The shaded regions mark the boundaries of the two output fan-in cones. The weight parameter (W_i) is given on the top of each gate. The fan-in cones weight (S_j) given on the right of corresponding output is found to be 14 and 12 for *O1* and *O2* respectively. According to these figures, the output *O1* is more susceptible to SETs than output *O2*. In other words, having a SET detection mechanism placed on *O1* can better improve the reliability of the circuit when compared to having it placed on *O2*. Deeper analyses and validations of this structural susceptibility analysis can be found in [9].

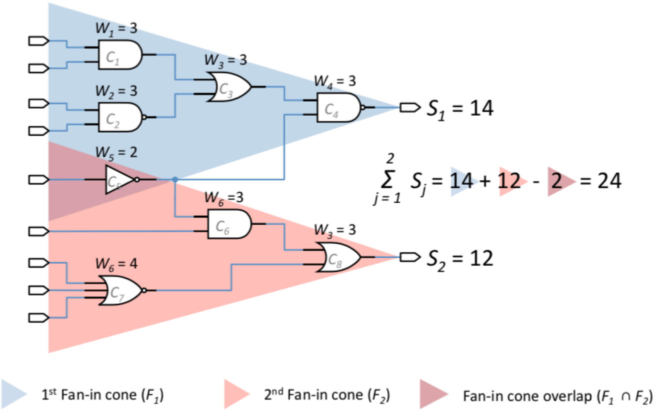

Figure 1. Application of the structural susceptibility analysis

III. Selective Error Detection for Arithmetic Circuits

An error detection architecture must be capable of detecting transient, permanent and delay faults that may occur in an arithmetic circuit. The error detection scheme we evaluate employs duplication and comparison to detect faults. Since the architecture relies on duplication of the arithmetic block and the use of comparator, its implementation incurs an overhead of more than 100% in terms of area and power.

A practicable way of providing the designer the freedom to control the cost and reliability improvement of error detection architecture implementation is to cleverly select the functions to be duplicated. It allows reducing the overhead in terms of area and power with duplication and comparison at a cost of the fault tolerance capability. Figure 2 shows a simplified scheme of the considered error detection architecture. It can be seen that the Reduced Copy Block (RCB) only implements a part of the arithmetic functions of the original Arithmetic Block (AB). A comparator represented by the block labeled as '==?' allows the fault detection.

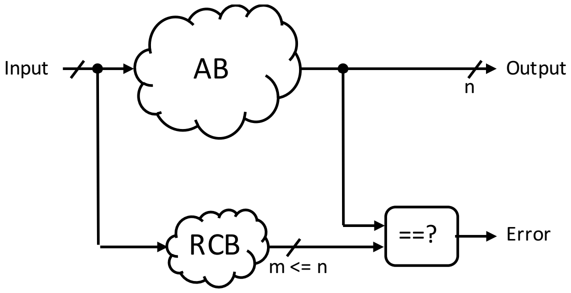

Figure 2. Error detection architecture

Next sub-sections address the different duplication scenarios we consider:

- a full duplication scheme;

- a reduced duplication scheme based on the susceptibility analysis;
- a reduced duplication scheme based on the logical weight of the arithmetic circuit outputs;
- a reduced duplication scheme based on an approximated structure.

Note that the comparator shown in Figure 2 must be adapted to the various duplication scenarios.

A. Scenario 1 (S1) – Full duplication scheme

This scenario represents the ideal case of the error detection architecture. In fact, when full duplication is used, the error detection architecture is able to detect all faults (transient, permanent and delay faults) that may occur in the arithmetic circuit. For this scenario, the comparator is a full comparator able to produce an error signal when it receives different binary values on its inputs.

B. Scenario 2 (S2) – Reduced duplication scheme based on the structural susceptibility analysis

Here, we use the structural susceptibility analysis to build a number of reduced copies of the arithmetic circuit. Each copy is created by selecting a set of outputs ranked by descending order of their weight S_j obtained by Algorithm 1. Consequently, the smallest copy corresponds to the logic cone driving the output having the highest weight S_j while the biggest copy corresponds to a copy of the circuit truncated from its logic cone driving the output having the lowest weight S_j. For this scenario, the comparator is reduced since the duplication has less outputs to compare to the original arithmetic circuit.

C. Scenario 3 (S3) – Reduced duplication scheme based on the logical weight

Since the structural susceptibility analysis only takes into account the circuit structure, here we consider the possibility to duplicate the arithmetic circuit by using a functional metric. We assume the arithmetic circuit to produce output words ranked form *LSB* (Less Significant Bit) to *MSB* (Most Significant Bit). The idea is to build the reduced copies of the arithmetic circuit based on logic cones driving the *MSB* up to the *LSB*. In that case, the smallest copy corresponds to the logic cone driving only the *MSB* output while the biggest copy corresponds to a copy of the arithmetic circuit truncated from its logic cone driving the *LSB* output. As for *S2*, the comparator is also a reduced comparator since the duplication has less outputs.

D. Scenario 4 (S4) – Reduced duplication scheme based on an approximate structure

The use of *S2* or *S3* to build the duplication scheme leads to an error detection architecture able to detect only faults affecting the common (structural/functional) duplicated area. Hence, a remaining set of faults will be not detected by these duplication schemes. These faults will affect the function of the arithmetic circuit by providing wrong answers. Consequently, we thus must understand the impact of the undetected faults on the application in order to determine if the outputs are still acceptable or not by the user. This characterization is usually done in the AxC context.

The AxC paradigm is based on the intuitive observation that rather than a perfect result, inner operations of a computing system can be selectively inaccurate for providing gains in efficiency (i.e., less power consumption, less area, higher manufacturing yield) [12, 13, 14]. An AxC structure is generally qualified by error metrics.

In this paper, we consider as error metrics to evaluate the different duplication scenarios the Worst-Case Error (WCE) metric defined by Equation 1.,

$$WCE = \max_{\forall i}\left|O_{approx}^{(i)} - O_{prec}^{(i)}\right| \tag{1}$$

where $O_{approx}^{(i)}$ ($O_{prec}^{(i)}$) is the ith output of the approximate (precise) implementation. We also use the Error Probability (EP) metric defined by Equation 2.,

$$EP = \frac{\#Faulty\ response}{2^n} \tag{2}$$

where *n* is the number of outputs of the arithmetic circuit.

Scenario *S4* consists in using as reduced duplication an approximate version from a public benchmark suite [11] of the arithmetic circuit. The approximate version is selected based on its area and timing properties compared to the original precise version. For this last duplication scenario, the comparator must provide an error signal when the arithmetic circuit processes a response larger than the *WCE* value of its approximate version used as duplication. More details on the design of such a comparator can be found in [15].

IV. Experimental results

The fourth duplication scenarios are compared using two case study precise arithmetic circuits: 8-bits adder *add8_001* and 8-bits multiplier *mul8_012* from the publicly available EvoApprox8b benchmark suite [11]. Table I provides an overview of the specifications of these two arithmetic circuits in terms of function and available versions (precise and approximate). To compare the different scenarios (*S1*, *S2*, *S3* and *S4*), we expose the results we obtained in terms of area and power consumption overhead with respect to *S1* as well as *EP* and *WCE* metric values. All netlists were obtained using the CAO tool Design Compiler from Synopsys [16] with the NanGate 45nm Open Cell Library [17]. The *S1* is implicitly shown in every figure where the area overhead is 100% as it is a full duplication.

Figures 3.a and 3.b respectively presents the results of *S2* with each possible reduced duplication of *add8_001* and *mul8_012* respectively. The different points in the figures correspond to the different reduced duplications produced by the considered scenario. For each reduced duplication cases (i.e. area overhead), we show the power consumption overhead, *EP* and *WCE* metrics. Figures 3.c and 3.d present the same result achieved when using *S3* as duplication scenario.

TABLE I. CASE STUDY SPECIFICATIONS

Circuit name	Functionality	Versions
add8_001	8 bits adder	56 precise
		449 AxC
mul8_012	8 bits multiplier	34 precise
		471 AxC

These results show that while the power consumption and the *EP* of these scenarios have a similar behavior, the *WCE* is lower for most of the versions of the duplications using *S3*. This behavior is explained by the fact that *S3* duplicates the arithmetic circuit with functional constraints while *S2* only consider its structure.

For better visualization and comparison of the previous results with respect to *S4*, we illustrate power consumption overhead, *EP* and *WCE* metric values of each scenario separately. In the approximate benchmark suite, we chose the 8-bits adder *add8_189* and the 8-bits multiplier *mul8_013* as approximate versions. The criterion for this choice were to have an area and a longest path timing in the range to the precise versions for a fair comparison.

For each scenario, Figures 4.a and 4.b show the Power consumption overhead of the 8-bits adder and 8-bits multiplier respectively. Results show that the power consumption overhead for *S4* is in the trend of *S3*. For the multiplier case study, the power consumption overhead achieved with *S4* is reduced about 3% compared to the best values obtained with *S2* or *S3*.

Figures 4.c and 4.d show the *EP* metric values of the adder and multiplier respectively. The *EP* achieved with an approximate version of the arithmetic circuit as duplication is lower than the ones obtained with the other scenarios; more than 25% in case of the adder and about 12% for the multiplier.

Finally, Figures 5.e and 5.f show the *WCE* metric values of the adder and multiplier respectively. The *WCE* of the selected approximate adder is of 3 and below the one obtained with *S3* for an equivalent area overhead. In the case of the multiplier, the *WCE* of the approximate version is 1424 which is 23 times lower than the *WCE* that *S2* and *S3* imply for a similar area overhead (*WCE* = 32767).

This set of comparisons with different duplication scenarios show that the use of approximate circuit as reduced duplication could be a good alternative to build an error detection scheme of arithmetic circuits. In fact, this duplication scenario offers similar values in terms of area and power overhead while reducing drastically the error metrics compared to the more conventional *S2* and *S3* duplication scenarios.

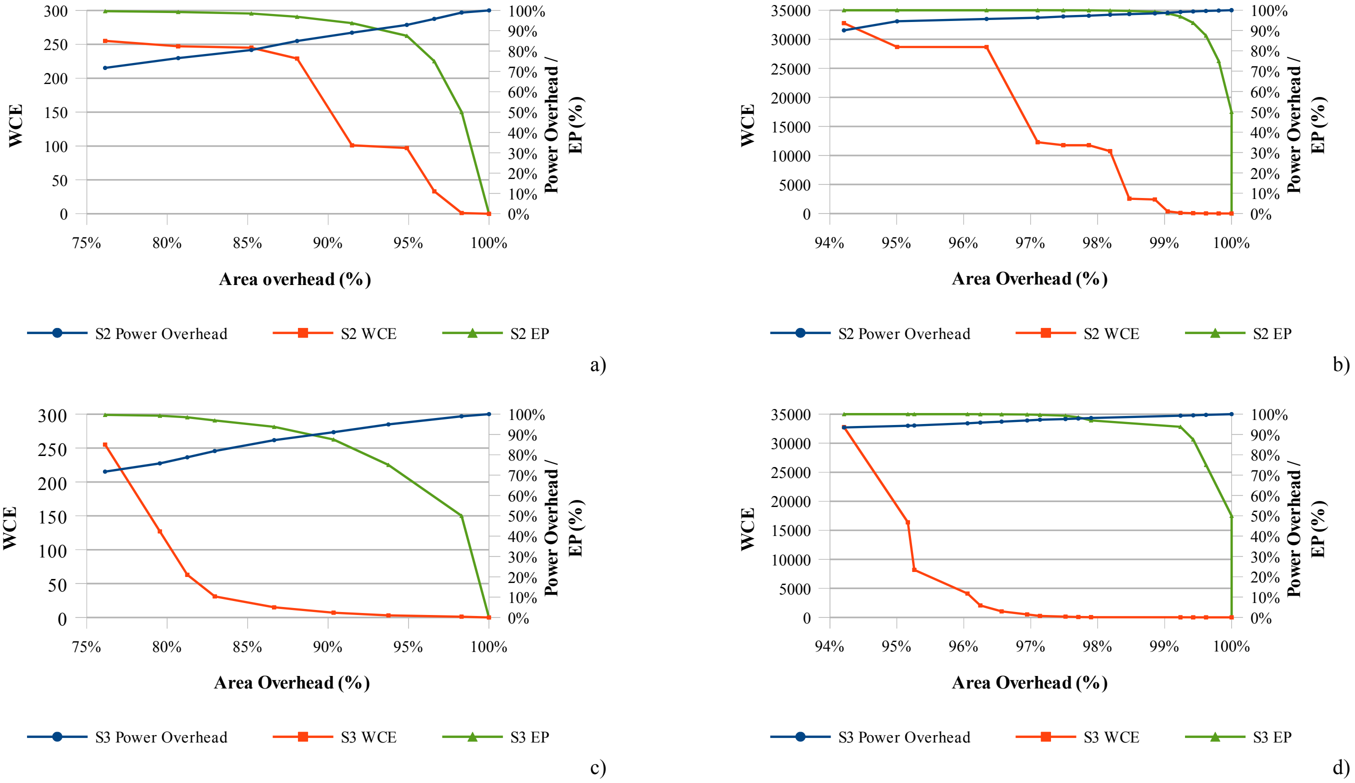

Figure 3. Scenario comparisions with respect to S1 with a) S2 for the add, b) S2 for the mult, c) S3 for the add and d) S3 for the mult

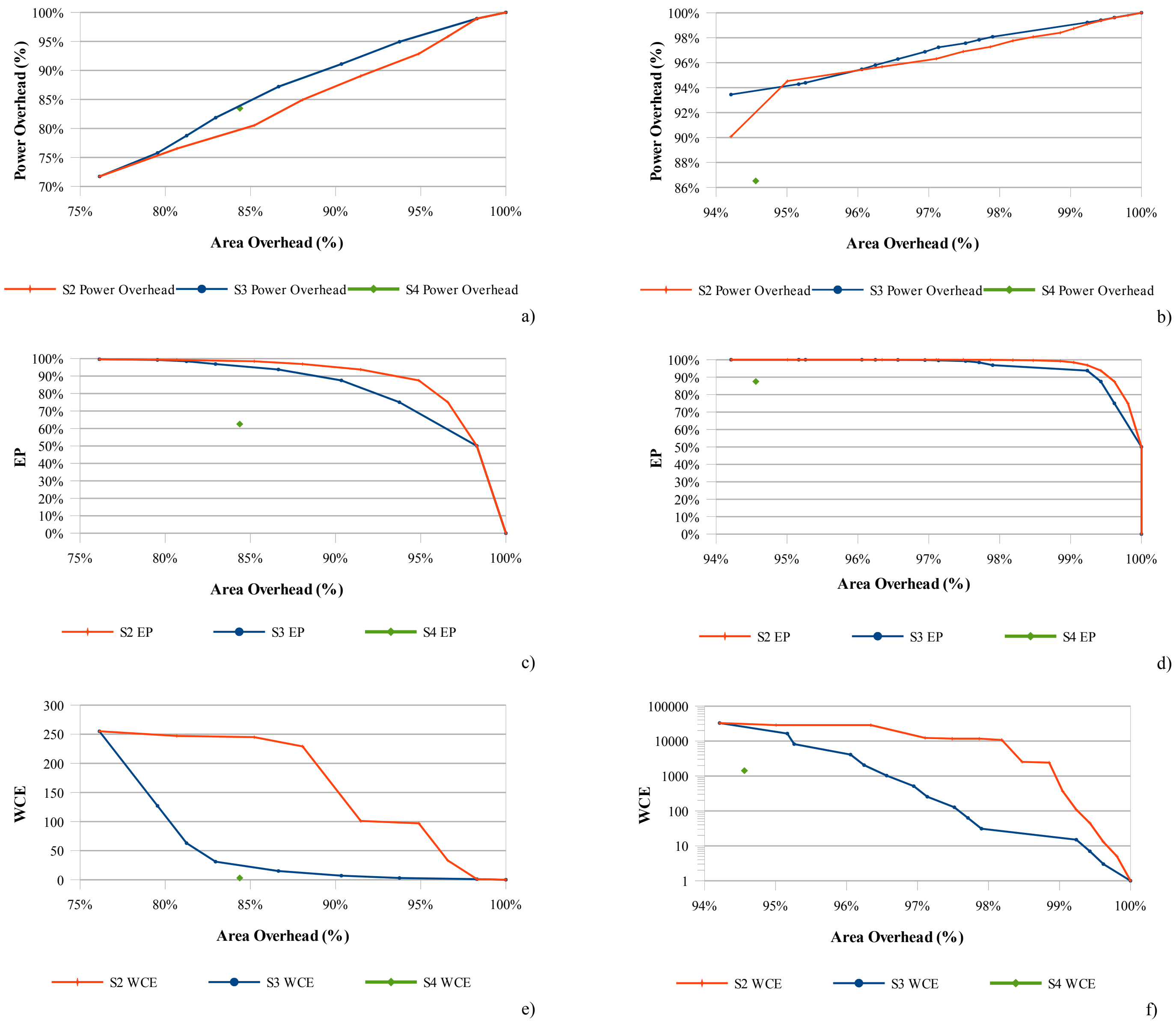

Figure 4. All scenario comparisions with respect to S1 with a) power overhead for the add, b) power overhead for the mult, c) EP for the add, d) EP for the mult, e) WCE for the add and f) WCE for the mult

V. CONCLUSION

In this paper, we have addressed the problem of selective hardening of arithmetic circuits. We have considered a duplication/comparison scheme as error detection architecture with different duplication scenarios. Experimental results have shown the interest of using approximate structures as duplication scheme since it provides much better WCE and EP rates than the structural susceptibility and weight-based methods with the same amount of hardware cost.

As future study, we intend to analyze deeper the use of approximate circuits in a duplication/comparison scheme in order to build a complete fault tolerant scheme for arithmetic circuits.

REFERENCES

[1] S. N. Pagliarini, L. A. B. Naviner, and J.-F. Naviner, “Selective Hardening Methodology for Combinational Logic,” *13th Latin American Test Workshop*, 2012, pp. 1–6.

[2] I. Polian, and J. Hayes, “Selective Hardening: Toward Cost-Effective Error Tolerance,” *Design Test of Computers*, vol. 28, no. 3, pp. 54–63, May 2011.

[3] C.Zoellin, H.Wunderlich, I.Polian, and B.Becker,“Selective Hardening in early Design Steps,” *13th European Test Symposium*, 2008, pp. 185–190.

[4] M. Fazeli, S. Ahmadian, S. Miremadi, H. Asadi, and M. Tahoori, “Soft Error Rate Estimation of Digital Circuits in the Presence of Multiple Event Transients,” *Design, Automation Test in Europe*, 2011, pp. 1–6.

[5] K. Mohanram and N. Touba, “Cost-Effective Approach for Reducing Soft Error Failure Rate in Logic Circuits,” *International Test Conference*, 2003, pp. 893–901.

[6] I. Polian, S. Reddy, and B. Becker, “Scalable Calculation of Logical Masking Effects for Selective Hardening Against Soft Errors,” *Annual Symposium on VLSI*, 2008, pp. 257–262.

[7] M. Maniatakos and Y. Makris, “Workload-Driven Selective Hardening of Control State Elements in Modern Microprocessors,” *VLSI Test Symposium*, 2010, pp. 159–164.

[8] C. Bottoni, B. Coeffic, J.-M. Daveau, L. Naviner, and P. Roche, “Partial Triplication of a Sparc-v8 Microprocessor using Fault Injection,” *Latin American Symposium on Circuits and Systems*, 2015, pp. 1–4.

[9] I. Wali, B. Deveautour, A. Virazel, A. Bosio, L. Dilillo, and P. Girard, “An Effective Hybrid Fault-Tolerant Architecture for Pipelined Cores,” *20th European Test Symposium*, 2015, pp. 1–6.

[10] I. Wali, A. Virazel, A. Bosio, L. Dilillo, P. Girard, and M. Sonza Reorda, “A Low-Cost Reliability vs. Cost Trade-Off Methodology to Selectively Harden Logic Circuits,” *Journal of Electronic Testing*, vol. 33, no. 31, pp. 25–36, February 2017.

[11] V. Mrazek, R. Hrbacek, Z. Vasicek, and L. Sekanina, “Evoapprox8b: Library of Approx Adders and Multipliers for Circuit Design and Benchmarking of Approximation Methods,” *Design, Automation Test in Europe Conference*, 2017, pp. 258–261.

[12] S. Mittal, “A Survey of Techniques for Approximate Computing,” *ACM Comput. Surv.*, vol. 48, no. 4, pp. 62:1–62:33, Mar. 2016.

[13] Q. Xu, T. Mytkowicz, and N. S. Kim, “Approximate Computing: A survey,” *IEEE Design Test*, vol. 33, no. 1, pp. 8–22, Feb 2016.

[14] J. Han and M. Orshansky, “Approximate Computing: An Emerging Paradigm for Energy-Efficient Design,” *18th IEEE European Test Symposium (ETS)*, 2013, pp. 1–6.

[15] M. Traiola, A. Virazel, P. Girard, M. Barbareschi, and A. Bosio, “Testing Integrated Circuits for Approximate Computing Applications,” *4th Workshop On Approximate Computing*, January 2018.

[16] Design Compiler. [Online]. Available: https://www.synopsys.com/

[17] NanGate. Nangate 45nm open cell library. [Online]. Available: http://www.nangate.com/?page id=2325.