Highly-Reliable Approximate Quadruple Modular Redundancy with Approximation-Aware Voting

Mahmoud Masadeh
Department of Computer Engineering
Yarmouk University
Irbid 21163, Jordan
mahmoud.s@yu.edu.jo

Alain Aoun, Osman Hasan, and Sofiène Tahar
Department of Electrical and Computer Engineering
Concordia University
Montreal, Quebec, Canada
{a_alain, o_hasan, tahar}@ece.concordia.ca

***Abstract*—Redundancy has been a general method to produce a fault-tolerance system. The Triple Modular Redundancy (TMR) with majority voters covers 100% single fault-masking, where the minimum area overhead is 200%. On the other hand, approximate computing is suitable for applications that can tolerate errors and imprecision in their underlying computations. Thus, inexact results allow reducing the computational complexity and hardware requirements with increased performance and power efficiency. This work explains how approximate computing could provide low-cost fault-tolerant architectures with an enhanced system's reliability. In particular, we implement a novel Quadruple Modular Redundancy (QMR) designs using three identical approximate modules in addition to the exact module. Moreover, a two-steps magnitude-based voter is proposed to be able to tolerate approximation error. To validate our approach, we conducted experiments and the results showed the ability to achieve high fault tolerance, i.e., 99.88%, while reducing the probability of system failure by 15%, with 62% and 49.5% reduced area and power, respectively, compared to the traditional TMR.**

I. Introduction

Modern integrated circuits (ICs) are increasingly moving towards reduced feature sizes with high integration density. However, this consequently results in reduced noise margins and an increase in susceptibility of applied voltages to external effects caused by noise and radiation [1]. Such critical effects can cause permanent damage (*hard errors*) or transient faults (*soft errors*), that may lead to a faulty system. Therefore, error mitigation through *fault tolerance*, at both hardware and software levels, is proposed to solve this major issue, which significantly affects embedded and microprocessor-based systems.

Fault-tolerant computing can be done in *hardware*, *software* or in a *hybrid* fashion. In hardware-based systems, components redundancy techniques, such as double modular redundancy (DMR) and triple modular redundancy (TMR), are used for fault detection and correction, respectively [2]. For software-based systems, redundancy techniques include the addition of redundant code, without additional hardware components. Such technique is very attractive in designs based on commercial-off-the-shelf designs such as commercial microprocessors [3]. Moreover, hybrid solutions combine the benefits of both hardware and software techniques.

The addition of redundant components, either hardware or software or both, introduces a non-negligible implementation overhead in terms of size, power and performance. Such overhead can be reduced by adapting a *selective* or *partial* redundancy [4], to protect only the essential parts of the system while keeping the remaining non-essential parts unprotected. On the other hand, approximate computing with its design-efficiency has been proposed to minimize the overhead associated to fault tolerance [5].

Approximate computing (AC) has emerged as an enabler for enhanced performance and reduced power consumption [6]. It has been used in a large range of applications, e.g., multimedia, machine learning, big data and scientific applications, which are error-resilient, i.e., these applications are able to tolerate imprecision in their underlying computations, due to several factors including a noisy and/or redundant input data and user perceptual limitations [7]. Approximation accuracy depends on the application, the user and input data, where the real definition of good quality of results is flexible. For example, in image processing, the final result images are perceived by humans, which is subjective between different people. Image processing applications can accept up to 10% errors in an error-tolerant context [8]. Such an error margin for accepted results can be leveraged upon to improve execution time and energy. In this paper, we investigate the use of AC in designing efficient fault-tolerance systems.

Triple Modular Redundancy (TMR) is one of the most traditional fault-tolerance techniques, which consists of a triplicated module and a voter for masking the errors, which may affect one of the three redundant modules. The overhead of the TMR is a 200% increase in area, where the output of the system is compared with the gold execution to detect the occurrence of faults that cause errors. Approximate TMR (ATMR) [9] has been presented for hardware projects as a way to achieve a fault coverage as good as the traditional TMR without a huge area overhead. However, as stated in [10], the previously researched approaches of ATMR, depend on including different versions of the approximate module, where only one module could give an inexact output at each input vector (these modules are approximated in a complementary manner). This allows the majority voter, for any input vectors, to select two matching outputs out of three. However, this can be an unrealistic assumption, since different approximate modules could lead to different results with minor similarities between their outputs. To overcome such infeasible assumption, we propose a highly-reliable innovative scheme by using three identical approximate modules in addition to the exact one.

Moreover, we propose a two-step approximation-aware voter based on output magnitude, in order to accommodate for approximation-induced errors in the absence of soft-errors and being able to compare multi-bit values.

The rest of the paper is structured as follows. Section II introduces the related work. Next, Section III explains our proposed methodology to reduce the overhead and enhance the reliability of TMR by means of AC. Section IV provides the experimental results. Finally, Section V concludes the paper and highlights the future work.

II. Related Work

Research efforts in the field of AC-based fault-tolerance systems can be mainly classified in terms of targeting functional modules [3], [5], [9]–[12] or the voter component [13]–[16]. For instance the authors of [3] used AC at the software level and designed a reduced-overhead software-based fault-tolerant system. In [5], the authors investigated the use of data precision reduction approximation technique for low-cost fault-tolerant architecture. Using approximate logic modules for TMR to reduce area overhead was initially proposed in [10]. Moreover, [11] proposed a novel technique for efficient utilization of acceptable error rate threshold to perform logic masking of soft errors. Similarly, the authors of [9] presented a low-cost approximate TMR based on loop perforation combined with data size and precision reduction. Recently, [12] proposed a low-cost solution based on four different approximate modules. However, each approximate module has a specific eliminated output, which is inapplicable for all modules. Generally, all of the previous approaches rely on having non-identical approximate modules, where only one of them can differ from the original for each input vector while other approximate modules have exact result. Such assumption is too idealistic. Therefore, our proposed solution is different by having three identical approximate modules in addition to the exact one.

Regarding the voter component, [13] presented a novel compact voter for approximate TMR using pass transistors and quadded transistors level redundancy for high fault masking. In [14], the authors presented a novel fault-tolerant voter circuit which itself can tolerate a fault and give error-free outputs by improving the overall system's reliability. Considering simultaneous fault occurrences at both the functional module and the voter, [15] proposed a fault-tolerant majority voter which is robust to faults in the presence of faults occurring internally or externally to the voter. Also, [16] proposed a simple and effective fault-tolerant voter circuit with high reliability and low cost. However, [14]–[16] targeted exact TRM not ATMR. Moreover, [13]–[16] consider a bit-level output which is unsuitable for modules with mult-bit or magnitude-based results. Thus, our proposed solution is novel in terms of having an approximation-aware magnitude-based voter.

III. Proposed QMR and Voter

We propose an architecture for Quadruple Modular Redundancy (QMR), which consists of three identical approximate modules, exact module and a two-steps voter, i.e., *Voter1* with the classical bit-based design and *Voter2* with a new magnitude-based design, as shown in Figure 1. The outputs of the approximate modules feed the *Voter1* component. Then, the output of the exact module and the result of *Voter1*, i.e., $Result_A$, are the inputs to the *Voter2* component, which is an approximation-aware magnitude-based design.

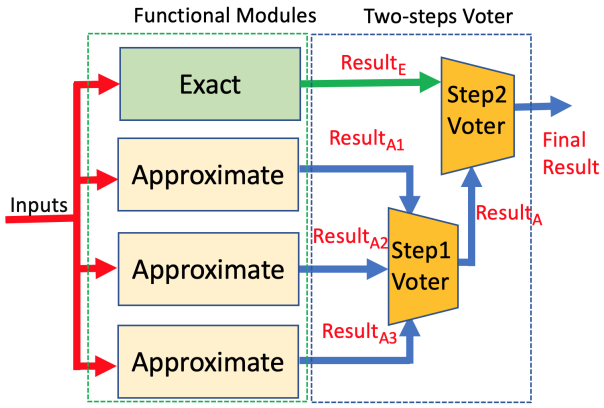

Figure 1: The Proposed Quadruple Modular Redundancy (QMR) and Two-Steps Voter

The proposed QMR has to deal with reduced accuracy which is intrinsic to approximation. Thus, we have to assure that: i) an approximation induced difference, between the exact and approximate modules, is not considered as an error in the absence of faults; and ii) a fault induced result, for the exact or approximate module, could be tolerated if its magnitude is still less than the acceptable approximation error, i.e., Error Distance (ED). Therefore, we propose to use an *acceptable threshold*, where the QMR considers an error if the final ED is higher than such threshold.

Generally, ATMR is a new notion with a few works targeting the voter design. In this work, we present an innovative QMR with a two-step voter under the assumption that *the voting circuitry does not have redundancy and does not fail*. However, the failure of a voter means the failure of the whole system even if all modules are functioning accurately. Due to approximation, the output of *Voter1* component for fault-free approximate modules is expected to differ from the exact output. Furthermore, the proposed *Voter2* compares the inputs based on their total magnitude rather than their binary representation, which is suitable for multi-bit outputs.

IV. Results and Discussion

In order to validate and evaluate our proposed methodology, we intially carried out a case study where the exact module is an 8-bit array multiplier. The approximate module is an 8-bit approximate array multiplier, where the full adders contributing to the lowest 9 significant bits of the results are utilizing approximate mirror adder 5 (AMA5) [17], i.e., *Design19* of [18]. Next, we evaluate area and power, accuracy, and reliability of the proposed QMR to shows its benefits.

A. Area and Power Assessment

Compared to a traditional TMR with 3 exact modules and a voter, the proposed QMR has an exact module, 3 approximate

Table I: The Percentage (%) of Fault Detection in the Exact Module

	Number of Clusters															
	2	**4**	**8**	**16**	**32**	**64**	**128**	**256**	**512**	**1024**	**2048**	**4096**	**8192**	**16384**	**32768**	**65536**
Run#1	56.28	56.82	57.43	58.43	58.51	59.87	61.10	63.64	65.82	68.91	72.20	77.11	80.55	87.82	93.78	99.81
Run#2	54.60	55.26	55.97	57.17	57.76	58.58	60.57	62.55	65.31	68.13	72.50	76.97	80.18	88.75	94.62	99.90
Run#3	52.20	52.44	53.39	54.63	55.08	55.92	57.83	60.35	63.03	65.77	69.59	75.66	78.32	87.50	95.21	99.93
Average	54.36	54.84	55.60	56.74	57.12	58.12	59.83	62.18	64.72	67.60	71.43	76.58	79.69	88.03%	94.54	99.88

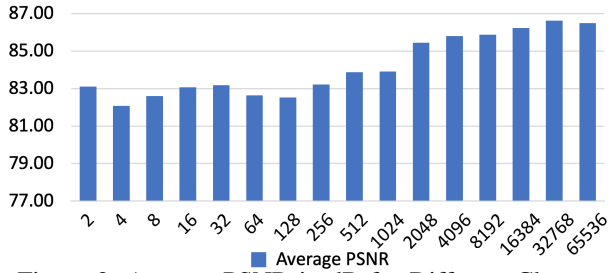

Figure 2: Average PSNR in dB for Different Clusters

modules, a traditional voter and an approximation-aware voter. For the analysis of design metrics of the proposed QMR, we utilized an XC7VX485T FPGA, from the Xilinx Virtex-7 family. Conventional TMR and proposed QRM exhibited a power consumption of 14.347W and 7.24W, respectively. Similarly, design area was 79 and 39 LUT for the same designs. This shows the approximation results in 62% and 49.5% savings in area and power, respectively, when compared to traditional TMR.

B. Accuracy Assessment

Design redundancy aims for fault correction, by having - at least- two out of three modules with the same output. The proposed QMR is able to detect a single-event upset (SEU) that could flip any input or output bit of the four modules. As shown in Figure 1, the approximate results, i.e., $Result_{A1}$, $Result_{A2}$ and $Result_{A3}$, are fed to the bit-based *Voter1*, which gives $Result_A$. Thus, a single faulty approximate module could be tolerated.

Two approximate modules could fail simultaneously with a the same fault magnitude. Then, the output of *Voter1*, i.e., $Result_A$ which is a faulty approximate value, could be closer to the exact result based on its magnitude. Therefore, *Voter2* tolerates such faults that keep the approximation error within the acceptable threshold. Similarly, a SEU on the exact module updates its magnitude. Accordingly, the output of the faulty exact module has an error magnitude ($Result_E$), which may be within the acceptable approximation error.

We exhaustively validated the accuracy of the proposed QMR, where the basic design module is an 8-bit multiplier. Thus, each module has $2^8 \times 2^8$ = 65536 input combinations. On average, the QMR was able to detect 99.88% of the faults in the exact module. Then, based on that, the approximate module is used to assess the final result.

Table I shows the percentage of faults detection in the exact module, based on the number of used clusters (C), where C represents the number of times to execute the proposed QMR. The number of adjacent inputs for each execution, is called cluster size, which is $\frac{65536}{C}$. For each cluster, *Voter2* of the proposed QMR checks the difference between the faulty output of the exact module and the output of *Voter1*. For example, if the number of clusters is 65536, then the cluster size equals one input. So, for every single input the faulty output of the exact module is checked to see if it is within the acceptable *error threshold*. For the example of 32 clusters, the size of each cluster is 2048 of adjacent inputs. Thus, every input is checked based on its magnitude to specify the corresponding cluster. Then, the associated error is checked, based on its magnitude, by *Voter2* to specify if the error magnitude is within the range of the 2048 corresponding inputs.

Figure 2 shows the average of the obtained *PSNR* for an image processing application, utilizing the same cluster configuration as in Table I. The average of the obtained PSNR is 84.16 dB with a minimum of 82.07 dB and a maximum of 86.61 dB. Generally, all obtained results have an acceptable quality with high PSNR.

Table II shows the percentage of detecting faults in the approximate modules with the Exact module being flawless, i.e., only $Result_A$ is faulty. For such scenario, $Result_E$ should be carried out to the final result. Hence it is recommended to keep the number of clusters less than 512, since it offers a balance between the cases shown in Tables I and II.

C. Reliability Assessment

According to Figure 1, the output of *Voter1* is considered valid if at least two approximate modules are fault-free. The relative failure of the output for *Voter1*, with respect to the failure of approximate modules, follows the rule of a binomial distribution. The probability of faulty *Voter1* output *Pf* is obtained using Equation 1 with *p* being the module's probability of failure. Furthermore, the output of *Voter2* would fail to deliver an acceptable quality if the exact module is faulty, and less than two approximate modules are flawless. The probability of failure *Pf* for *Voter2* follows the theory of

Table II: The Percentage (%) of Fault Detection in the Approximate Module

	Number of Clusters															
	2	**4**	**8**	**16**	**32**	**64**	**128**	**256**	**512**	**1024**	**2048**	**4096**	**8192**	**16384**	**32768**	**65536**
Run#1	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.12	0.43	6.19	14.38	31.47	49.67	69.20	80.19	96.04
Run#2	0.00	0.00	0.00	0.00	0.00	0.00	0.04	0.15	0.83	5.82	18.98	35.23	48.58	74.63	85.18	96.36
Run#3	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.13	0.58	6.14	17.25	39.99	49.56	74.29	88.21	96.31
Average	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.14	0.61	6.05	16.87	35.56	49.27	72.71	84.53	96.23

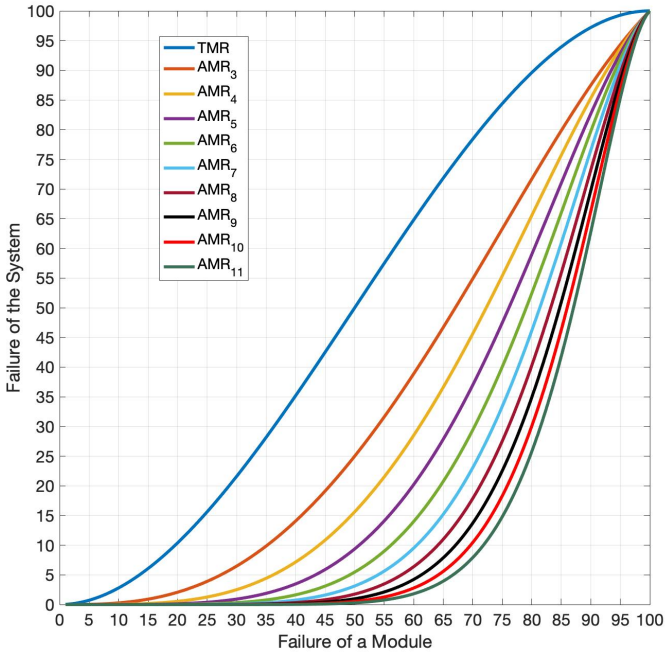

Figure 3: The Relative Probability of System Failure vs Component Failure for Different Module Redundancy Circuits

two independent events, given by Equation 2, with *P(A)* and *P(B)* representing the failure probability for the exact module and the output of *Voter1*. For the sake of simplification, all modules, i.e., exact and approximate, are considered to have the same failure rate. Figure 3 depicts the system's relative failure in terms of the module's failure for the conventional TMR and approximate module redundancy with *n* approximate modules and one exact module, i.e., AMR_n. The proposed QMR has an average of 15% fewer chances of failure compared to the regular TMR.

$$Pf = \binom{1}{n}(1-p)(p)^{n-1} + \binom{0}{n}(p)^{n} \quad (1)$$

$$Pf = P\left(A \cap B\right) = P(A) \times P(B) \quad (2)$$

Clearly, we notice that as the number of approximate modules increases, the probability of system's failure decreases. However, more approximate modules means a higher area and power consumption along with additional complexity in the design of *Voter1*. Thus, for a specific area and power budget, the maximum number of approximate modules to integrate together should be specified. For the stand-alone approximate modules of 8-bit multipliers, we found that the power consumption of 3 exact modules equals the power consumption of 11 approximate modules, while the area of 3 exact modules equals the area of 8 approximate modules. Thus, the benefits of design approximation are clear.

V. Conclusion

Several works have been proposed to reduce the overhead, i.e., area, energy and performance, associated with fault-tolerant systems. Approximate computing has shown a great promise to improve the efficiency of error-resilient applications. In particular, approximate arithmetic functional units provide promising results regarding generating TMR implementations with a reduced overhead while maintaining an acceptable accuracy. In this work, we investigated the impact of approximate computing on improved system reliability with reduced area and power consumption. Specifically, we showed that it is possible to use approximate computing to implement efficient fault-tolerant architectures. For future work, similar to Figure 3, we plan to explore the advantages of integrating various numbers of approximate modules, i.e., 4 to 10 approximate modules.

References

[1] P. Shivakumar, M. Kistler, S. W. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," in *International Conference on Dependable Systems and Networks*, 2002, pp. 389–398.

[2] A. Aponte-Moreno, A. Moncada, F. Restrepo-Calle, and C. Pedraza, "A review of approximate computing techniques towards fault mitigation in HW/SW systems," in *Latin-American Test Symposium*, 2018, pp. 1–6.

[3] A. Aponte-Moreno, C. Pedraza, and F. Restrepo-Calle, "Reducing Overheads in Software-based Fault Tolerant Systems using Approximate Computing," in *Latin American Test Symposium*, 2019, pp. 1–6.

[4] X. Vera, J. Abella, J. Carretero, and A. González, "Selective replication: A lightweight technique for soft errors," *ACM Transactions on Computer Systems*, vol. 27, no. 4, 2010.

[5] Rodrigues, G. S. and Fonseca, J. and Benevenuti, F. and Kastensmidt, F. and Bosio, A., "Exploiting approximate computing for low-cost fault tolerant architectures," in *Symposium on Integrated Circuits and Systems Design*, 2019, pp. 1–6.

[6] M. Masadeh, O. Hasan, and S. Tahar, "Comparative study of approximate multipliers," in *Great Lakes Symposium on VLSI*. ACM, 2018, pp. 415–418.

[7] S. Venkataramani, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Approximate computing and the quest for computing efficiency," in *Design Automation Conference*, 2015, pp. 1–6.

[8] A. Rahimi, A. Ghofrani, K. Cheng, L. Benini, and R. K. Gupta, "Approximate associative memristive memory for energy-efficient GPUs," in *Design, Automation & Test in Europe*, 2015, pp. 1497–1502.

[9] G. Rodrigues, J. Fonseca, F. Kastensmidt, V. Pouget, A. Bosio, and S. Hamdioui, "Approximate TMR based on successive approximation and loop perforation in microprocessors," *Microelectronics Reliability*, vol. 100-101, 2019.

[10] I. A. Gomes, M. G. Martins, A. I. Reis, and F. L. Kastensmidt, "Exploring the use of approximate TMR to mask transient faults in logic with low area overhead," *Microelectronics Reliability*, vol. 55, no. 9, pp. 2072 – 2076, 2015.

[11] T. Arifeen, A. S. Hassan, H. Moradian, and J. A. Lee, "Probing Approximate TMR in Error Resilient Applications for Better Design Tradeoffs," in *Euromicro Conference on Digital System Design*, 2016, pp. 637–640.

[12] B. Deveautour, M. Traiola, A. Virazel, and P. Girard, "QAMR: an Approximation-Based Fully Reliable TMR Alternative for Area Overhead Reduction," in *European Test Symposium*, 2020, pp. 1–6.

[13] T. Arifeen, A. Hassan, and J. Lee, "A fault tolerant voter for approximate triple modular redundancy," *Electronics*, vol. 8, no. 3, p. 332, 2019.

[14] R. Kshirsagar and R. Patrikar, "Design of a novel fault-tolerant voter circuit for TMR implementation to improve reliability in digital circuits," *Microelectronics Reliability*, vol. 49, no. 12, pp. 1573 – 1577, 2009.

[15] P. Balasubramanian and K. Prasad, "A fault tolerance improved majority voter for TMR system architectures," pp. 102 – 122, 2016.

[16] T. Ban and L. A. de Barros Naviner, "A simple fault-tolerant digital voter circuit in TMR nanoarchitectures," in *International NEWCAS Conference*, 2010, pp. 269–272.

[17] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 1, pp. 124–137, 2013.

[18] M. Masadeh, A. Aoun, O. Hasan, and S. Tahar, "Decision Tree-based Adaptive Approximate Accelerators for Enhanced Quality," in *International Systems Conference*, 2020, pp. 1–5.