QAMR: an Approximation-Based FullyReliable TMR Alternative for Area Overhead Reduction
Bastien Deveautour, Marcello Traiola, Arnaud Virazel, Patrick Girard

To cite this version:

HAL Id: lirmm-03035640
https://hal-lirmm.ccsd.cnrs.fr/lirmm-03035640
Submitted on 2 Dec 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
QAMR: an Approximation-Based FullyReliable TMR Alternative for Area Overhead Reduction

B. Deveautour  M. Traiola  A. Virazel  P. Girard
LIRMM, Univ. of Montpellier / CNRS
Montpellier, France
<lastname>@lirmm.fr

Abstract—In the last decade, Approximate Computing has become a trend topic for several error tolerant applications. In this context, the use of such paradigm for reducing the area and power cost of conventional fault tolerant schemes (i.e. Triple Modular Redundancy) has been investigated recently. Unfortunately, existing solutions cannot ensure the same level of reliability compared to conventional TMR. In this paper, we propose a novel fully robust approximation-based solution suitable for safety-critical applications that can reduce the cost compared to conventional TMR structures. This solution is based on the use of four approximate modules with an overall smaller area overhead compared to a TMR made of three precise modules. The main constraint is that, for a given output of the precise module, at least three approximate modules (among four) can feed the voter with the same output. In order to build our Quadruple Approximate Modular Redundancy (QAMR) structure, we use a simple and random process whose goal is to remove different outputs with the corresponding fan-in logic from each approximate module in such a way that the above constraint is satisfied. The majority voter and its mode of operation remains the same as in the TMR. To validate our approach, we conducted experiments and results demonstrate that it is possible to achieve the fault tolerance of a full TMR approach while reducing the area overhead up to 24.28%.

Keywords—Combinational circuits; fault tolerance; error correction; triple modular redundancy; approximate computing.

I. INTRODUCTION

During the lifespan of a system used in harsh (e.g. radiative) environment, its hardware is subject to various physical phenomena that may alter its performance or provoke errors [1]. More specifically, circuits manufactured with advanced nanometer technologies are prone to errors due to effects like aging or wear-out, leading to permanent fault and hence hard errors, and energetic charged particles striking, leading to transient faults and hence soft errors. This is a consequence of shrinking the transistor dimensions, augmenting the density of gates per chip, and of course the increased demand for safety-critical applications in harsh environments. When those faults propagate through the logic, they can be captured by a memory cell (e.g. a flip-flop) and stored as faulty value. In that case, a permanent or transient fault becomes a hard or soft error that may cause a malfunction of the system.

In response to the stress experienced by the circuits during their lifespan, several structures have been designed to maintain their accuracy. A well known existing structure capable of tolerating soft and hard errors is the Triple Modular Redundancy (TMR). A triplication of the circuit with a majority voter ensures an extreme logic error masking at a cost of a 200% area and power overhead. A TMR masks (tolerates) permanent or transient faults occurring in one or several modules (provided that they do not impact the same outputs if several modules are faulty) for any vector applied to its inputs.

Approximate Computing (AxC) is an emerging computation paradigm in which an inaccurate result rather than a guaranteed accurate one can be accepted at the cost of a reduced accuracy [2]. AxC has been applied to resilient applications, e.g. speech recognition, image encoding, etc., where an approximate result is sufficient for their purpose [3]. From the hardware standpoint, AxC enables the creation of circuits whose output values may differ from the original circuit for a certain set of input values [4].

In [5], AxC was applied to TMR, where two or even three of the modules are different approximations of the original circuit. Other proposals of a low cost TMR based on approximate computing were presented in [6] and more recently in [3] and [7]. Such AxC applications to TMR lead to both lower area overhead and power consumption. However, such advantages come at the expense of a reduced error-masking capability, which makes approximate TMR not suitable in safety-critical scenarios.

To overcome the above issue, we propose the Quadruple Approximate Modular Redundancy (QAMR). QAMR is a novel scheme to ensure a full logic masking (tolerance) of transient and permanent faults. Like TMR, QAMR masks all faults occurring in the modules and for which the voter still has a majority of correct responses. It achieves the same accuracy than the TMR while still benefiting from approximation advantages (i.e., smaller area and power overhead). To implement the QAMR, we use four approximate circuit replicas. The fundamental condition to respect is that, at a given time, at least three precise responses (i.e., non-approximated) must be delivered by the QAMR structure. In other words, the four Approximate Integrated Circuits (AxIC) must be approximated in a complementary manner.

In this paper, we present the QAMR approach and a simple circuit approximation method developed to demonstrate its advantages. This approximation method is based on complementarily cutting outputs (and the related fan-in internal logic) from each circuit replica composing the QAMR. This is done in such a way that three replicas of the same output are always available. Consequently, we are able to use the same majority voter as in TMR schemes. To validate our approach experimentally, we used publicly available combinational circuits to implement the QAMR scheme. Experimental results show promising results that encourage a deeper exploration. Indeed, for several benchmarks, QAMR achieves a smaller area overhead than the TMR, while still providing the same reliability level.
II. STATE-OF-THE-ART ON AXC BASED FAULT TOLERANCE

**TMR** is a fault tolerant scheme made of three identical instances of a circuit connected to a majority voter. **TMR** protects against faults (permanent or transient) occurring in one or several modules (provided that they do not impact the same outputs if several modules are faulty), for any input vector. This fault tolerant solution requires a 200% area overhead due to the two extra circuit instances. Moreover, we must add the voter area that depends on the number of circuit outputs.

**AxC** is able to target different layers of computing systems, from hardware to software [2]. In this work, we focus on **AxICs**, which are the outcome of **AxC** application at hardware level, specifically on **Integrated Circuits** (ICs). Approximate hardware is a design concept in which the designer selectively relaxes non-critical specifications to improve chip area, power and/or run time. Non-critical specifications usually refer to resilient applications where the level of accuracy can be lowered without impacting the system integrity.

Some authors have developed different strategies to create approximate combinational hardware circuits. These strategies can be grouped into the three main approaches:

- **Ad-Hoc approximate circuits**, which usually involves a different handling for each approximation case. The **Ad-Hoc** approach is a necessary choice in the case the designer needs to implement or remove specific functionalities to the original circuit. For example, authors in [8] and [9] propose an accuracy-configurable adder and multiplier, respectively, to reduce power consumption if the application is resilient. Although they can be efficient, **Ad-Hoc** approaches are usually very resource-demanding when applied to large circuits.

- **Automatic approximate circuit synthesis methodologies**, which assist the designer in reducing the area of a circuit while minimizing the impact on the accuracy. Authors in [10] present an algorithm created to design general inexact circuits able to achieve a certain **Quality of Resilience (QoR)**. A quality function determines if the circuit meets the QoR requirements. In [11], authors developed an evolutionary technique based on genetic codes to approximate circuits until they reach a state in which they are considered too far from their original circuit. Larger circuits can benefit from these synthesis methodologies.

- **Hardware neural accelerators to implement approximate functions**. Neural Networks (NNs) offer a significant parallelism capability and can be efficiently accelerated by dedicated hardware to gain in performance/energy at the expense of accuracy. For example, in [12], authors propose NN-based accelerators to approximate Transcendental Functions (i.e. cos, exp, log, pow, and sin).

In the literature, several proposals have been made to reduce the **TMR** area overhead by using **AxC**. This scheme is known as **Approximate TMR** (**ATMR**) [5] and its extension as **Full ATMR** (**FATMR**). The **ATMR** scheme uses two **AxICs** and a precise one as replicas, while **FATMR** uses three **AxICs**. In these implementations, only one **AxIC** can give an erroneous answer at a time. In other words, each approximate module has its own unique domain of approximation. However, producing such a low cost **TMR** may suffer from severe limitations in term of reliability.

Let us resort to Figure 1 to illustrate the above mentioned issue. A green block in the figure refers to an original or approximate replica of the circuit that provides a correct output when the vector $x$ is applied at its input. Conversely, a yellow block refers to an approximate replica of the circuit that provides a wrong output (because of the approximation) when the vector $x$ is applied at its input. The figure shows the different possible scenarios in the case of a single fault. Figures 1.a and 1.b show the **ATMR** scheme in two different scenarios. In both scenarios, a **Single Event Transient (SET)** occurs in one of the three replicas. The outcome, however, is very different depending on the nature of the faulty replica. In the case of Figure 1.a, the **SET** occurs in the replica that gives an approximate wrong response for the input vector $x$. In this case, the voter will deliver a precise output because the two remaining replicas are giving a precise response for the same input vector $x$. Conversely, in the case of...
Figure 1.b, the SET occurs in one replica that might have delivered a correct response. In this case, the voter will deliver an incorrect output. Figures 1.c and 1.d show the same scenarios. The only difference is that the FATMR scheme has only approximate replicas. If the SET occurs in a replica delivering a wrong approximate response for the input vector \( x \) (Figure 1.c), the voter will deliver a correct output. However, if the SET occurs in a replica that should deliver a correct response for the input vector \( x \), the voter will deliver an incorrect output. In summary, input vectors when only two out of three replicas compute correctly are vulnerable to SET. The authors refer to those vectors as unprotected.

Hardware design of fault tolerance circuits for safety-critical applications is a crucial task. Realizing it by using AxC-based schemes raises some important challenges. Specifically, it is mandatory to know the workload of such application. For FATMR schemes, it implies that input vectors that are not protected by the structure must not be critical for the application. Such design requirements can be challenging and not always achievable, even for resilient applications.

### III. PROPOSED QAMR SCHEME

The goal of our approach is to achieve the TMR reliability level while reducing area and power costs. We propose to make use of AxC in a quadruple duplication scheme. As with a classic TMR, the goal is to protect the whole structure function for all input vectors against permanent and transient faults occurring in one or several modules (provided that they do not impact the same outputs if several modules are faulty). Such fault-masking coverage will be suitable for safety-critical applications even when the workload is unknown.

![Figure 2: TMR, FATMR and QAMR single fault masking](image)

Figure 2 shows the principle of the proposed QAMR scheme. In this case, if we consider the precise modules in TMR having precision domains \((D1, D2, D3, D4)\) triplicated, we cover each precision domain against a single fault scenario. FATMR covers most of their precision domains but a single fault may affect the outcome. For example, the structure would mask a fault on \(D1\) (as \(D1\) appears in all three modules of FATMR structure) whereas a fault on \(D2\) or \(D3\) could not be masked. Instead, QAMR offers a complete coverage since we triplicate all the precision domains. By doing so, we can reach the same TMR reliability level. At the same time, the four AxICs enable the opportunity to achieve efficiency gains in terms of reduced area and power consumption. The underlying insight is that a good AxC technique achieves more gains than it reduces the system accuracy.

Hereafter, we formalize the conditions required to generate a QAMR structure: i) the circuit must have at least four outputs in order to obtain four AxICs, ii) for each AxIC with missing accuracy for a specific set of input vectors, all other AxICs must tolerate this deficiency and provide a correct output response.

To implement our QAMR approach, we developed a circuit approximation method to produce the AxICs respecting the conditions mentioned above. The goal of this work is to simply demonstrate the QAMR feasibility and the opportunities it creates for fault-tolerant architectures. The design of optimal approximation techniques is left out for future works. Our preliminary method is illustrated in Figure 3.

![Figure 3: QAMR scheme](image)

For a given circuit, we remove one group of outputs from the original circuit to obtain a first AxIC module, and we repeat this process as many times as needed to form four different AxICs. Meanwhile, we also remove the fan-in logic belonging to the group of outputs removed for each different AxIC. To respect the above-mentioned conditions (complete coverage of the precision domains), it is important that an output is removed from only one of the four approximate replicas. The advantage of such a structure is the use of the same voter than the TMR scheme. Indeed, for any input vector, the voter will always deal with three bits for each output of the circuit. For better clarity, Figure 3 shows an example of a 4-bits output circuit in the QAMR scheme. Since each AxIC has only one missing output, the voter is able to execute the majority vote just like in a classic TMR scheme.

### IV. QAMR DESIGN FLOW

Exploring all possibilities of iteratively removing one group of outputs for each circuit is not possible when using circuit with a large number of outputs. We applied our method to
benchmarks with up to 245 outputs. The number of complementary groups of outputs for QAMR for a circuit having 245 outputs is $3.19e+63$. To avoid such a huge and impractical exploration, we arbitrarily generate complementary groups of outputs in a random manner to obtain one QAMR version, then evaluate the area variance between the generated QAMR version and the previous ones, and finally iterate the process until the variance becomes lower than a threshold defined by the user. QAMR version with the lowest area is the final (best) QAMR.

Figure 4 sketches the flow of the proposed circuit approximation method. Starting from the netlist of the original circuit, a direct synthesis allows creating TMR by adding a majority voter for further comparison with QAMR. In parallel, we arbitrarily generate complementary groups of outputs in a random manner to create four distinct approximate versions of the original netlist.

![Figure 4: QAMR design flow](image)

Once the algorithm has provided four approximate versions with the corresponding complementary groups of outputs, we perform a logic synthesis and obtain the four AxICs (modules) of the QAMR structure. Each module lacks one group of outputs and their respective fan-in logic. We create the QAMR by adding a majority voter.

Then, if the area of the QAMR is smaller than the area of the TMR, we compare the new synthesized QAMR to a former best QAMR version stored in a database. Each time the area of a new QAMR is smaller than the area of the best QAMR version, it is saved and the former best QAMR candidate is overwritten in the database. If the area of the QAMR is larger than the area of the TMR or of the best QAMR, the QAMR version is discarded. Besides, each time a new QAMR version is synthesized, an area report is generated to update the area variance calculated by using Equation 1:

$$V_A = \frac{1}{n} \sum_{i=1}^{n} (A_i - \bar{A})^2$$

where $V_A$ represents the area variance, $A_i$ is the area of the QAMR version obtained during the $i^{th}$ iteration of the process described in Fig. 4, and $n$ is the total number of iterations. Once a new QAMR version has been generated, compared, and saved or discarded, the variance is compared to a threshold defined by the user. If it is higher than the threshold, the process is re-run and an additional iteration will provide a new QAMR version. Otherwise, the algorithm stops and the best QAMR version saved in the database is considered as the final QAMR.

Note that, the two values used during the first process iteration to calculate the variance are the following: i) the area of the first QAMR generated version and ii) the TMR area to which we compare our QAMR area.

V. EXPERIMENTAL RESULTS

A. Experimental Setup

Experiments have been done by using the Combinational Multi-Level and Two-Level circuits from the publicly available LGSynt91 benchmark suite [13]. For each circuit, we obtained the classic TMR by using three precise versions of the circuit and a voter. In a same way, we composed the QAMR with four approximate versions of the circuit and the same voter. We used the principle described in Section III to create the four approximate versions for each circuit. We used Design Compiler of Synopsys [14] for circuit synthesis, using the NanGate 45nm Open Cell Library [15].

From the LGSynt91 benchmark suite, we select only circuits that have five or more outputs. This is important, as the first step after identifying the logic function of a given circuit is to form random groups of outputs. In the case a circuit has less than four outputs, the creation of four approximate modules is impossible. Note that with four outputs, the random selection of output groups will always give the same combinations.

B. Results Analysis

To fairly compare our results with those obtained with the TMR scheme, we use a Relative Area Gain (RAG) metric. Note that results do not consider the voter area since it is the same in both schemes. This means that we consider the TMR as our baseline with 0% of area gain. Thus, the higher the RAG, the better the QAMR area performance with respect to the TMR.

Equation 2 shows how RAG is calculated. $A_p$ represents the area of each precise circuit in the TMR. $A_m$ represents the area of each approximate module in the QAMR.

$$RAG = \frac{3 \cdot A_p - (A_{x1} + A_{x2} + A_{x3} + A_{x4})}{3 \cdot A_p}$$

(2)

Figure 5 shows RAG achieved for all benchmark circuits used in our experiments. Results indicate that 19 out of 52 circuits have a lower RAG when using the QAMR scheme. Some circuits like Apex1 or e64 have a RAG above 20%. On the contrary, RAG from circuits presenting area loss is generally staying above -10%. Peculiarly, results for Alu4, however, are really poor with our QAMR approach, with an additional area cost of 40%.

With results having such heterogeneity, we analyzed which parameters could help in determining what type of circuit would be suitable for a QAMR approach. We came to the following observation. Our approximation method consists in removing outputs and their associated fan-in logic without removing the logic shared with the other (preserved) outputs. One key characteristic of such method is therefore the number of nodes in the circuit that lead to more than one output. This number...
divided by the total number of nodes gives the so-called Shared Logic Rate (SLR) of the circuit.

Let us assume that the success of the QAMR approach remains in removing output’s cones that share most of their logic with other cones. The removal of a cone with such a high SLR will allow the synthesis tool to remap the circuit in a configuration that was not necessarily interesting before. Indeed, synthesis tools rely on heuristics to perform the best possible technology mapping. It is reasonable to assume that a synthesis tool can perform better with a circuit that has been simplified by removing logic from it. On the contrary, the synthesis tool will not be able to remap and optimize the remaining logic that was shared with a low SLR output cone since the remapping options are more limited.

Let us assume that the success of the QAMR approach remains in removing output’s cones that share most of their logic with other cones. The removal of a cone with such a high SLR will allow the synthesis tool to remap the circuit in a configuration that was not necessarily interesting before. Indeed, synthesis tools rely on heuristics to perform the best possible technology mapping. It is reasonable to assume that a synthesis tool can perform better with a circuit that has been simplified by removing logic from it. On the contrary, the synthesis tool will not be able to remap and optimize the remaining logic that was shared with a low SLR output cone since the remapping options are more limited.

Figure 6 shows results obtained previously, this time ranked from circuits with the highest SLR to circuits with the lowest. We observe that below 20% of SLR, our QAMR scheme underperforms the TMR most of the time. We can see that only 6 out of 23 circuits with an SLR lower than 20% have a positive RAG. On the other hand, 13 out of the 29 circuits with an SLR higher than 20% outperform the TMR versions by achieving a positive RAG.

![Figure 5: Area gained by QAMR compared with TMR](image1)

![Figure 6: QAMR area gains ordered by SLR](image2)
In addition to area gain results, we obtained relative power and timing gains as well. Power consumption values are estimated values (rather than exact values) given by the commercial synthesis tool. Nonetheless, they are relative between every TMR and QAMR we explored, making so that a fair comparison of the power consumption is finally obtained.

Timing values are based on the timing of the longest path in TMR and QAMR schemes. Relative Power Gain (RPG) obtained is positive for nearly 60% of the studied circuits with 14% of the circuits above 20% RPG (with a best case of 46% RPG for K2 circuit). On the contrary, RPG for circuits presenting power loss is generally above -10% RPG and represents 24% of the circuits (with a worst case of -95% RPG for e5 circuit). Note that 7% of the circuits have a 0% RPG. Regarding the Relative Timing Gain (RTG), 63% of the circuit have a positive RTG, with 30% above 10% RTG (with a best case of 75% RTG for e64 circuit). Only 5% of the experimented circuits have a negative RTG and 19% present 0% RTG. Those results also confirm the interest of the presented design exploration.

VI. CONCLUSION AND DISCUSSION

In the context of error-tolerant applications, approximate computing trades off some computing accuracy with increased performance, decreased area footprint and/or power efficiency. In this context, studies in the literature proposed to relax reliability constraints to achieve gains in circuit area and power consumption. Despite the efficiency optimization opportunities brought by this kind of techniques, reliability still represents a key requirement in most advanced safety-critical computing systems: sacrificing reliability could result in the production of more cost-efficient systems, but also in endangering human lives. In particular, previous works on approximation-based TMRs presented the advantage of reducing its area cost compared to the standard TMR. However, such advantage comes at the expense of a reduced fault tolerance, preventing the Approximate TMR to be used in safety-critical applications.

In this paper, we have presented a solution to profit from the benefits brought by AxIC without sacrificing the reliability requirements. We proposed the novel Quadruple Approximate Modular Redundancy (QAMR) to reduce the standard TMR area cost without sacrificing the offered QoR.

To investigate the feasibility of the approach, we used a simplistic method based on the removal of a random portion of output’s cones for each one of the AxIC. Despite that, we managed to obtain very promising results showing that it is possible to use AxIC to reduce area costs without sacrificing reliability requirements. Obtained results clearly indicate that QAMR offers a cheaper alternative to the standard TMR scheme for safety-critical applications. Further studies now are needed to establish enhanced approximation techniques to fully exploit AxIC opportunities in safety-critical scenarios. A first possibility is to enhance the approach used in this work, by smartly selecting the output groups to remove for each AxIC. Understanding which cones are determinant to the simplification process should allow new uneven combinations of outputs and thus, enhance the area gains.

Although we developed the QAMR scheme by using a structural approach to prove its feasibility, other approaches must be explored. Among them, we can exploit the fact that circuits can also be approximated from a functional point of view. For example, representing a circuit as Sums of Products (SOP) could allow a designer to remove specific and different min ters for each AxIC. In that case, each AxIC would have a precision domain depending on input vectors rather than output cones. The challenge of such a functional approach resides in the design of a majority voting logic. Regardless of the functional technique utilized, it must discriminate an input vector from another to determine which AxIC will deliver an approximate response. This approach will explored in the near future.

In general, more sophisticated and efficient logic synthesis techniques are required to fully profit from AxIC opportunities, when it comes to safety-critical scenarios. In particular, advanced mathematical model are needed to turn an abstract specification of a desired QAMR behavior into an actual gate-level implementation. Synthesis tools based on such models will offer to designers a cheap alternative to the TMR scheme, still perfectly suitable in safety-critical contexts.

REFERENCES


