EasyChair Preprint

№ 9361

Data and Fault Aware Routing Algorithm for NoC Based Approximate Computing

Ibrahim Krayem, Romain Mercier, Cédric Killian,
Angeliki Kritikakou and Daniel Chillet

November 24, 2022

Data and Fault Aware Routing Algorithm for NoC Based Approximate Computing

Ibrahim Krayem
Inria, IRISA, Univ Rennes
Lannion, France
ibrahim.krayem@irisa.fr

Romain Mercier
Inria, IRISA, Univ Rennes
Lannion, France
romain.mercier@irisa.fr

Cédric Killian
Inria, IRISA, Univ Rennes
Lannion, France
cÃľdric.killian@irisa.fr

Angeliki Kritikakou
Inria, IRISA, Univ Rennes
Rennes, France
angeliki.kritikakou@irisa.fr

Daniel Chillet
Inria, IRISA, Univ Rennes
Lannion, France
daniel.chillet@irisa.fr

ABSTRACT

Due to transistor shrinking and core number increasing in System-on-Chip (SoC), fault tolerance has become a critical concern. Given the amount of data communications on such architectures, Network-on-Chips (NoCs) lead a crucial role in terms of performance. Even if fault correction approaches have been developed, they cannot efficiently address several permanent faults on NoC, due to their high hardware costs and correction limitations. In parallel, Approximate Computing domain considers applications that can tolerate errors, hence allowing fault mitigation instead of correction. This latter brings the opportunity of low implementation cost techniques to improve the reliability of SoC.

In this work, we propose a routing technique which selects a path between cores according to data type and permanent fault positions. Error tolerant data are able to cross faulty paths by using a bit-shuffling error mitigation technique. Critical data circumvent faulty paths or are duplicated and shuffled in case there is no other correct path available. Results show that our routing technique allows to maintain all the communication paths within the NoC for a large amount of permanent errors. To further evaluate the behavior of the proposed technique, we performed a comprehensive analysis of the technique on the packet latency and saturation injection rate with respect to the number of faults and traffic type.

CCS CONCEPTS

• **Hardware → Network on chip**; **Fault tolerance**.

KEYWORDS

Network-on-Chip, Adaptive Routing Algorithm, Fault Mitigation, Approximate Computing, Bit-Shuffling

1 INTRODUCTION

Since several decades, technology evolution enabled high transistor density, reaching today billions of transistors per chip. While frequencies and chip density met the power limit, performance increase has been reached by inserting more cores into a single chip, giving birth to SoCs, multi- and many-cores. However, the increase in number of cores induced more data transfers, which cannot be handled by conventional communication means. To address this gap, NoC appeared as a scalable solution for communications between a large number of cores [19].

In this technology era, NoC became more sensitive to permanent faults. As the transistor size reaches 10 nm and below [2], the engraving thinness causes more and more hardware defects, due to manufacturing process [5]. During system operation, aging defects [10], like electromigration, Negative-Bias Temperature Instability (NBTI), Hot Carrier Injection (HCI), Time-Dependent Dielectric Breakdown (TDDB), and radiations [1] become additional sources of permanent faults. Manufacturing and aging defects are well known sources of Single Bit Upsets (SBUs), whereas radiations can lead to SBUs and Multiple Bit Upsets (MBUs) [14]. While SBUs are composed by several Single Event Upsets (SEUs), which affect several bits in the same word, MBUs, also called burst faults, are induced by a single SEU, which affects several adjacent bits of the same data word.

In this context, fault tolerant techniques are applied to the NoC to correct/mitigate permanent faults [9]. These techniques are based on i) mitigation through routing algorithms [8], ii) hardware reconfiguration through spare resources or default backup paths [6], iii) correction through circuit replication [5], and iv) information redundancy [11]. Although the aforementioned approaches are efficient for single permanent fault, they are less appropriate for multiple permanent faults.

Besides correction techniques, approximate computing, that considers applications able to tolerate errors, brings the possibility to mitigate faults instead of correcting them. It offers new opportunities to make circuits more resilient, with solutions that are less area hungry, and the possibility of still using the hardware with the presence of faults. For instance, a run-time approach, named Bit-Shuffling (BiSu) [12], reduces the impact of multiple permanent faults in NoC datapaths, by shuffling the bits inside a flit and transferring the faults on the Least Significant Bits (LSBs) of each data. However, the impact on the latency has not been evaluated for this technique neither the impact on the performance with respect to traffic type, e.g. critical or error tolerant data.

To deal with both critical and error-tolerant data in a cost-effective way, we propose a routing algorithm for NoC-based computing architectures. The proposed algorithm, named BiRoPro, runtime decides the transmission path based on the data type and the position of faults. As long as the data are error-tolerant, they are routed on faulty paths applying bit-shuffling to reduce the fault impact. Critical data use an adaptive circumventing path without faults. In case such a path does not exist, the critical data are spread

on duplicated flits and shuffled for a correct transmission. Moreover, we propose a comprehensive study on the latency with respect to the permanent fault density within the NoC and the traffic type.

The rest of the paper is organized as follows: Section 2 presents the related work. Section 3 describes the required background and BiRoPro technique. Considered architecture and experimenal setup are presented in Section 4, while the evaluation results of the proposed approach are presented in Section 5. Hardware costs are given in Section 6, and finally, Section 7 concludes this work.

2 RELATED WORK AND POSITIONING

Regarding correction/mitigation of permanent faults in NoC, several approaches have been proposed. Existing methods are based on reconfiguration using spare resources [9] or default-backup paths with re-transmission [6]. However, under multiple permanent faults, the number of required spare resources and re-transmission latency is prohibitively increased. Other techniques use spatial redundancy, i.e., Triple Modular Redundancy (TMR) [5], and information redundancy, i.e., Error-Correcting Code (ECC) [16], to protect NoC architectures. When multiple permanent faults need to be addressed, their induced hardware costs are drastically increased.

Regarding NoC routing algorithms, deterministic approaches, such as XY, provide high performance in terms of latency and bandwidth. However, they cannot handle permanent faults, potentially leading to a high number of unavailable paths between Intellectual Properties (IPs). As depicted in Fig. 1a, there is no XY available path for communication from IP_3 to IP_5 and from IP_0 to IP_2 (permanent faults are illustrated by red lightnings and unavailable paths with red dotted arrow), while communication from IP_3 to IP_7 is routed through the fault-free path (depicted by the green arrow). Therefore, fault-tolerant adaptive routing algorithms [7] are proposed to circumvent faulty paths in NoC architectures, using only the healthy resources [8]. This behavior is depicted by the fault-free path (green arrow) which circumvents the permanent fault in the communication from IP_3 to IP_5 in Fig. 1b. Typical techniques include rules to avoid congestion and deadlock during packet transmissions. Although these methods handle permanent faults with lower hardware costs, the latency, throughput and network congestion increase and IPs can become unreachable, such as IP_2 in Fig. 1b.

Last, but not least, fault tolerant methods based on approximate computing recently offered techniques to protect communications on the NoCs. A bit-shuffling hardware mechanism, called (BiSu) [12], exploits the position of permanent faults and changes the bit organization inside flits to reduce the fault impact. This opens new opportunities in a faulty NoCs, with respect to the error tolerance of data, hence calling for specific routing algorithm. However, for application domains where not only error tolerant data, but also critical data are required, such approaches that reduce the fault impact, without correcting the faults, are less appropriate.

To deal with this limitation, we propose a Bit-shuffling based Routing Protocol (BiRoPro) for NoCs dedicated for applications with both critical and error-tolerant data. The main idea is to associate bit-shuffling technique with adaptive algorithm offering the possibility to route critical data on fault-free paths. Error-tolerant data are routed on a faulty path following a bit-shuffling approach to reduce the fault impact, e.g., through the faulty path illustrated with a dashed blue arrow from IP_3 to IP_5. On the contrary, critical data will benefit from fault-free paths, highlighted in green. Note that, the proposed algorithm allows to maintain every path on the architecture, even from IP_0 to IP_2, hence outperforming classic adaptive routing algorithms. In the case of isolated IP, error-tolerant data benefit from the mitigation technique, while critical data are sent with double flits as illustrated with a double yellow arrow [12]. Our method is more detailed in the next section.

3 PROPOSED APPROACH

In this section, we present the Bit-shuffling Routing Protocol (BiRoPro). First, Section 3.1 explains the target domain and assumptions of the proposed method. Then, the principle of the BiSu technique is presented in Section 3.2. Finally, the proposed BiRoPro routing protocol is detailed in Section 3.3.

3.1 Target Domain and Assumptions

BiRoPro mitigates multiple permanent faults, which can occur in NoC, and especially on the datapath part of the interconnect affecting data flits which transit on the NoC. As illustrated by the red lightnings in Fig. 1 and Fig. 2, faults can be located in i) the interconnections between routers, and ii) the buffers and the crossbar within each router. Due to nanoscale technologies and power scaling, devices and components become more susceptible to multiple permanent faults [14]. As the buffers and the crossbar are the largest components of a router [4], they have higher probability of accumulating faults due to radiation effects, manufacturing defects or other intrinsic failures. For the same reasons, interconnections are often impacted by permanent faults, usually expressed as stuck-at, short or bridge faults [3]. Such faults are managed in a similar way by the proposed method.

We assume that the positions of the faults are known and provided by methods such as Built-In Self-Test (BIST) techniques [13], which diagnose faults in interconnections and routers using Test Pattern Generator (TPG) and Output Response Analyzer (ORA) blocks. Other techniques available in the literature can be used to diagnose the permanent faults in a NoC [18]. As these techniques are largely studied in the literature, they are not detailed further. We consider a source routing guaranteeing the routing algorithm to be deadlock and live-lock free.

3.2 Error mitigation technique:Bit-Shuffling (BiSu)

The BiSu technique extends a NoC router with extra hardware blocks, i.e. Shuffler (S) and De-shuffler (D) blocks, as shown in Fig. 2-(a). The flits of size S_F bits, which transit through the NoC, are divided into N_{SF} blocks of bits, called Subflit (SF). Each SF consists of S_{SF} bits, which are re-organized by shuffler blocks to minimize fault impacts.

For instance, in Fig. 2, packets are organized with three 8-bit flits ($SF = 8$), and each flit is decomposed in 2-bit subflits ($S_{SF} = 2$). If two faults appear on bits 6 and 7 of the router, each flit will be corrupted and the impact on data (e.g. on an 8 bits uchar) will be very high. Before crossing the faulty router, the Shuffler block (S) re-organizes the flits by shuffling subflits SF_0 and SF_3 to reduce the

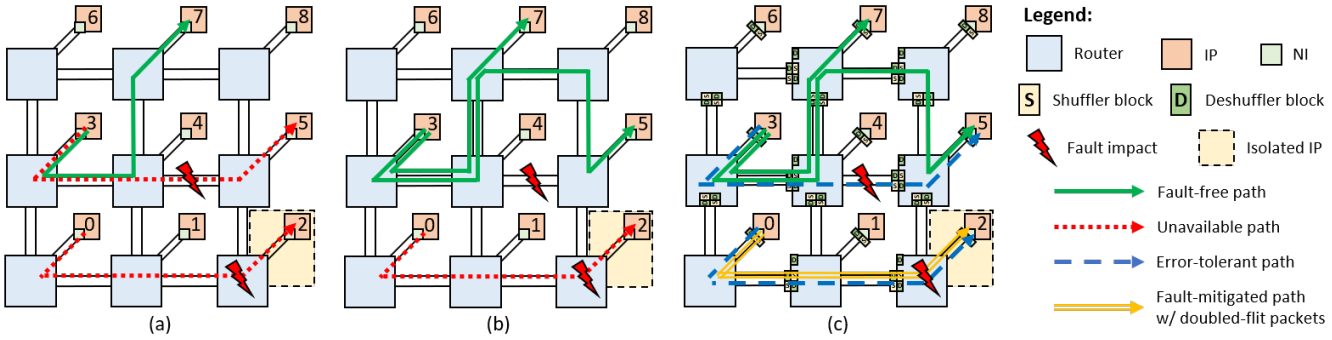
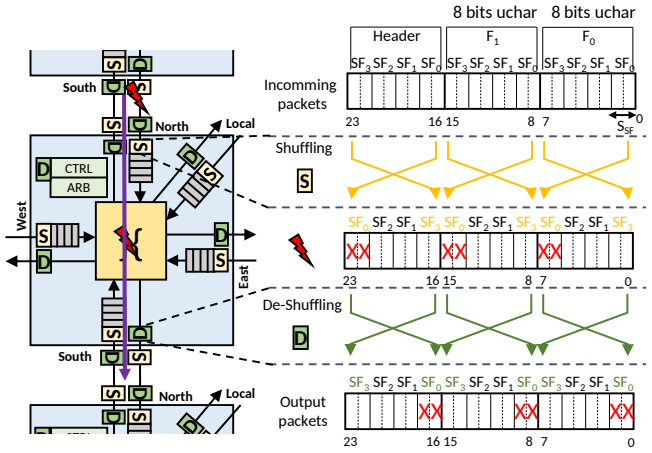


Figure 1: Illustration of the behavior of state-of-the-art routing algorithms and the proposed BiRoPro approach on a faulty NoC highlighting 3 communications: IP_3 to IP_7, IP_3 to IP_5 and IP_0 to IP_2. (a) XY routing algorithm. (b) Adaptive routing algorithm. (c) Proposed BiRoPro approach.



a) Original NoC extended with the BiSu method.

b) Illustration of the BiSu method. (S_F = 8, S_{SF} = 2)

Figure 2: NoC extended with the BiSu method.

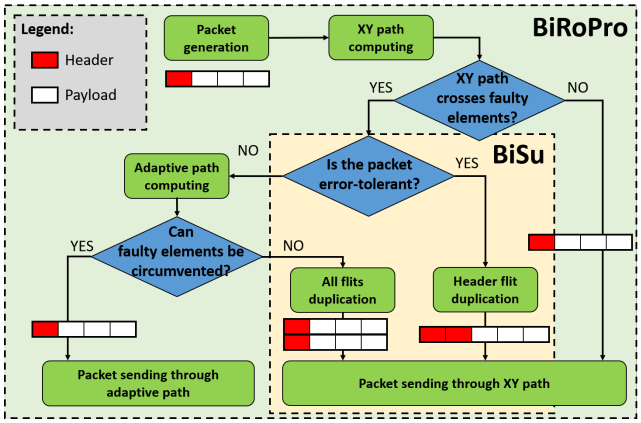


Figure 3: BiRoPro diagram.

impact of the faults, as only LSBs will be affected. The de-shuffler block (D) brings back the initial order of the SFs, as illustrated in Fig. 2-(b).

The BiSu technique is applied to mitigate errors on the interconnection bus and inside the router. One extra D block is added in the routing controller (CTRL) to read the routing information of the shuffled header and forward the current packet towards the expected output.

Header flits are critical, i.e., cannot tolerate any fault. If sufficient number of bits are unused in the header flit (localized on the LSBs positions), then BiSu can exploit these unused bits to protect the header bits. Otherwise, a spreading technique is used where the header is divided in two parts. Each part is placed into the Most Significant Bits (MSBs) on a flit. In this way, unused bits are created and thus BiSu is able to protect the header bits. The flit-spreading technique could also be used for critical data, e.g., instructions, where each flit is split on two flits, at the cost of more buffer occupancy.

3.3 Bit-shuffling Routing Protocol (BiRoPro)

Fig. 3 presents the overview of BiRoPro. First, the proposed approach determines if the XY path to reach the destination is impacted by any fault. If the path is fault-free, the packet is sent through the NoC using XY routing algorithm. This behavior is illustrated through the communication from IP_3 to IP_7 in Fig. 1-(c). If the path is faulty, BiRoPro determines the nature of the data contained in the packet. If the data is not critical (error-tolerant), then the packet is sent through the NoC following XY path applying the BiSu approach, where the header is spread on two flits to mitigate faults. This behavior is depicted in Fig 1-(c) for the communication from the IP_3 to IP_5 depicted by the dashed blue arrow.

Regarding critical packets, i.e., packets that cannot tolerate faults, an adaptive routing algorithm is used to forward them until the destination IP by circumventing faulty elements in the NoC. This behavior is depicted in Fig 1-(c) through the communication from the IP_3 to IP_5 represented by the green arrow. In the case where the destination IP is isolated from the source IP, the packets are sent through the NoC following the XY path using BiSu approach to protect data. This is illustrated in Fig. 1-(c) through the communication from IP_0 to IP_2 represented by the doubled yellow arrow.

As these flits contain critical data, they are spread on two flits in order to enable data protection.

4 EXPERIMENTAL SETUP

4.1 Architecture and fault model

For our experiments, we consider an 8×8 NoC with a mesh topology. The router service time is set to 2 clock cycles, and the router is considered without virtual channel. Packets include 64 bytes of data organized in 16 flits plus one header flit of 32 bits, and are injected according to different traffic models, such as random, perfect shuffle and transpose traffic patterns. The proposed BiRoPro approach is implemented and is compared to i) the XY routing algorithm and ii) an adaptive routing algorithm searching for the shortest fault-free path between source and destination IPs [7].

Faults are randomly injected in the NoC datapath, i.e., routers and interconnections. To avoid fault masking due to data values, we consider a permanent fault by simulating a bit-flip on the data. In our experiments, we express the number of injected faults by using the fault density metric, which corresponds to the average number of faults per router. For example, 10 injected faults within an 8×8 NoC correspond to a fault density of 0.16. Note that it is possible to have several faults on a same router.

4.2 Simulator

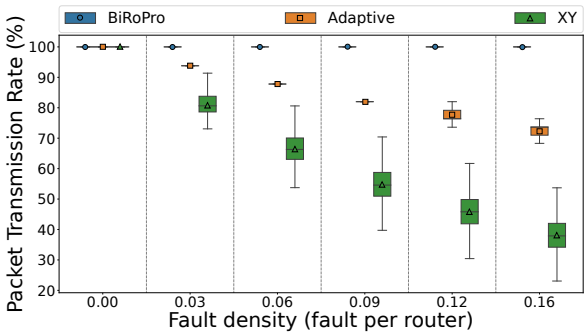
In this paper, given the large exploration space regarding the fault positions, we use an analytical model based on queuing theory, offering effective NoC-performance analyzing [15]. This model provides different performance metrics, i.e., average buffer utilization and average packet latency. The latter enables to compute the saturation packet injection rate used to evaluate our contribution. Each experiment is performed 10,000 times and box plots are used to represent the results through the minimum, median and maximal values and the first and third quartiles.

5 RESULTS

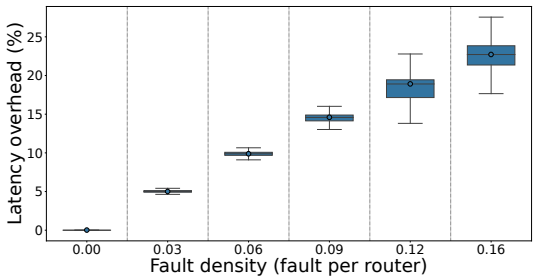
5.1 Correct Packet Transmission and Latency

We study the packet transmission rate and latency of the proposed BiRoPro method. We first compute the percentage of packets that are transmitted through the NoC without error, considering a random traffic pattern. Results are compared with XY and adaptive routing algorithms. Then, we compute the latency overhead by taking as reference the best case, i.e., a XY routing algorithm with fault free NoC, i.e. the shortest path. Results are respectively presented in Fig. 4a and Fig. 4b according to the NoC fault density.

In Fig. 4a, we can observe that the XY and adaptive routing algorithm drastically reduce the transmission rate of correct packets, when the fault number increases in the NoC. On the contrary, the proposed BiRoPro technique maintains the transmission rate constant even in presence of multiple faults. For example, considering a fault density equal to 0.09 fault per router, i.e., 6 faults in the 8×8 NoC, the percentage of correct packet transmission is, on average, 54.98% and 82.64%, for XY and the adaptive routing algorithm, respectively. BiRoPro maintains the correct packet transmission at 100%. Moreover, we can deduce that the percentage of correct packet transmissions decreases, when the fault density increases.



(a) Correct transferred packet rate.



(b) Latency overhead compared to a fault-free XY routing algorithm.

Figure 4: NoC performance evaluations using BiRoPro.

In the following we only highlight results on BiRoPro which is the only capable of routing any pair of IPs in presence of faults. Indeed, as shown in Fig. 4a, in presence of faults, XY and adaptive routing algorithms are not able to maintain all the paths and some IPs are isolated. This will lead to less injected packets as our method. Moreover, it will also reduce the amount of available IPs, hence decreasing the system performance. This latter is out of scope or the proposed paper.

As the proposed BiRoPro is the only approach to achieve 100% under multiple faults, we further analyse its behavior regarding latency overhead in Fig. 4b. The critical packets circumvent the faulty routers or are doubled in case of an isolated IP, which naturally impacts the latency compared to a fault-free execution. Note that, section 5.3 further analyses the latency overhead considering error-tolerant packets. From the obtained results, we observe that with higher fault density, the gap between the best and worst case latency increases. For instance, with a fault density of 0.16 (i.e., 10 faults), the latency overhead varies from 17% to 27% with an average of 23%. Such variation is due to the fault positions, potentially leading to long paths that circumvent the faulty routers.

5.2 Saturation Packet Injection Rate

In this part, we study the congestion induced by the proposed method considering i) Random, ii) Transpose and iii) perfect Shuffle traffic patterns, considering only critical packets. In Random pattern, the source and the destination are randomly generated. In Transpose pattern, some pairs of cores have a higher communication probability than others, whereas the rest of the traffic is

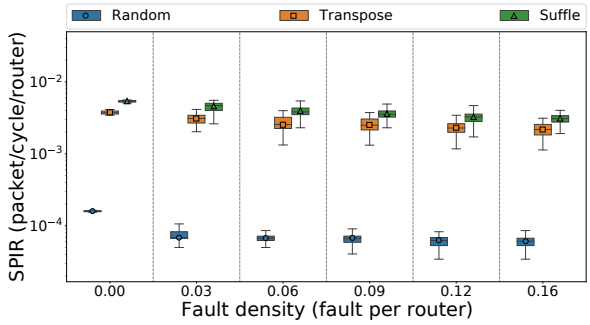


Figure 5: Saturation Packet Injection Rate of BiRoPro technique for different traffic patterns.

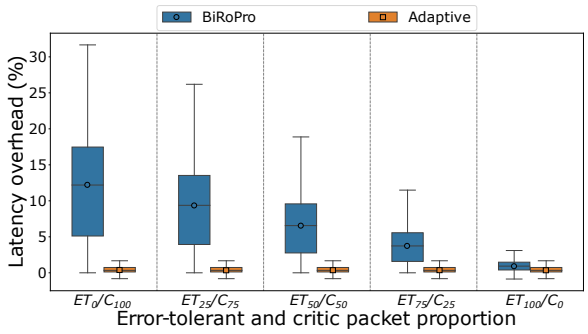
uniform random. Given a vector of N elements, the perfect Shuffle pattern is a communication pattern that divides the elements of this vector into two equal pieces, and then, combines these pieces by shuffling them [17]. The values of the Saturated Packet-Injection Rate (*SPIR*) are computed according to the analytical model presented in Section 4.2 with respect to the fault density in the NoC. The *SPIR* is considered reached for a *PIR* providing an average latency higher than 10× the latency of a zero-load NoC, with a very low packet injection rate $1.562 \times 10^{-5} packet/cycle/router$.

The obtained results are shown in the Fig. 5. We can first observe that the Random traffic pattern provides more congestion in the NoC, i.e., the packet injection rate that provokes the saturation of the NoC is lower. Transpose and Shuffle patterns are almost similar regarding the *SPIR*. For instance, with a fault-free NoC, the *SPIR* are 1.588×10^{-4}, 5.344×10^{-3}, and 3.778×10^{-3} $packet/cycle/router$, respectively for Random, Transpose and perfect Shuffle traffic patterns. In terms of impact on the *SPIR*, the perfect Shuffle traffic pattern is the less sensitive to fault, with only a *SPIR* reduction equal to 41% for a fault density of 0.16 compared to its fault-free value. Regarding the Transpose traffic pattern, the reduction of the S_{PIR} is 49%, due to proportion of random traffic included which bring more contention. Finally, the random traffic pattern is the more sensitive with a reduction of 62% on the S_{PIR}, which is reasonable regarding the amount of injected faults, i.e., 10 for an 8×8 NoC.

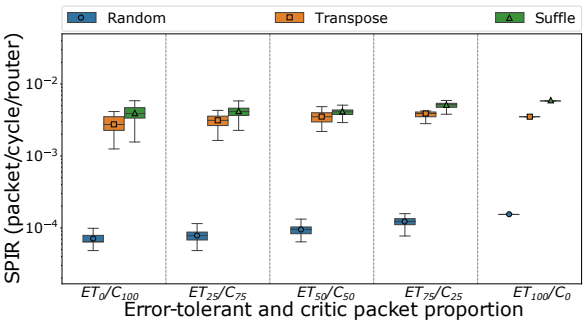
5.3 Packet-Type Impact

In this part, we explore the impact of the ration between error-tolerant and critical packets, which transit the NoC, on the latency overhead and the *SPIR*. Due to page limitations, for each packet ratio we represent one box plot considering a fault density from 0 to 0.16.

Fig. 6a presents the latency overhead of BiRoPro according to ET_x/C_y, with x and y the percentage of error-tolerant and critical packets, respectively. Results are normalized taking as basis a fault-free NoC with an XY routing algorithm. Overall, we observe wider box plots compared to the ones presented in Fig. 4b. Indeed, each box represents a fault density ranging from 0 to 0.16. For instance, ET_0/C_{100} in Fig. 6a integrates all the results from Fig. 4b. In Fig. 6a, we can observe that the latency overhead of BiRoPro decreases when the proportion of error-tolerant packets increases. This is



(a) Latency overhead.



(b) Saturation PIR.

Figure 6: Impact of the approximate-packet proportion ET_x/C_y with x and y respectively the percentage of error-tolerant and critical packets considering 8×8 NoC.

due to the fact that error-tolerant packets directly use faulty paths instead of circumventing faults, or by duplicating the flits to reach isolated IPs. Finally, with only error-tolerant packets, the proposed algorithm has only 2% of latency penalty compared to a fault-free NoC.

Fig. 6b presents the *SPIR* of BiRoPro for the Random, Transpose and perfect Shuffle traffic patterns according to the ratio of packet type. We observe that the SPIR increases (higher is better) with the increase of error-tolerant packets. Moreover, it almost reaches the fault-free *SPIR* for a traffic with only error-tolerant packets. Indeed, the *SPIR* for ET_{100}/C_0 are 1.539×10^{-4}, 3.512×10^{-3} and 5.821×10^{-3} $packet/cycle/router$, respectively for Random, Transpose, and perfect Shuffle traffic patterns. Error-tolerant packets can be directly go through faulty routers only with duplicated spread header, then maintaining packet latency low.

6 HARDWARE COST STUDY

In this section, we evaluate the hardware cost induced by the BiRoPro approach in terms of area and power overheads. The extra hardware blocks used in the BiRoPro approach are synthesized on 28 nm FDSOI technology through a High Level Synthesis (HLS) tools of Mentor Graphic by targeting a 1 GHz clock frequency.

Table 1 presents the hardware costs of the proposed technique which impacts both router and NI architectures. We show the result for two sizes of flits, 32- and 64-bits, respectively with S_{SF} equal to

Flit and subflit size	Routers				NIs				Total			
	Costs		Overheads		Costs		Overheads		Costs		Overheads	
	Area (µm²)	Power (mW)	Area (%)	Power (%)	Area (µm²)	Power (mW)	Area (%)	Power (%)	Area (µm²)	Power (mW)	Area (%)	Power (%)
$S_F = 32$ $S_{SF} = 4$	2, 289, 123	1, 981.2	7.0	6.4	770, 534	688.1	147.5	101.2	3, 059, 657	2, 669.3	24.8	21.1
$S_F = 64$ $S_{SF} = 8$	3, 946, 602	3, 442.4	7.6	7.3	1, 457, 844	1, 167.2	156.0	84.1	5, 404, 445	4, 609.6	27.6	19.9

Table 1: Hardware overhead for BiRoPro at the Network Interface (NI) and NoC level considering 8 subflits per flit.

4 and 8 bits. This ratio of flit and S_{SF} bring a good trade-off between error mitigation and area overhead [12]. We can see that, for 32-bit flits, the area overhead if only 7% for the routers. Regarding the NI the area overhead is 147% as it includes packets reorganization with the possibility to spread each flit in two. However, it has to be noticed that NIs are only small part of the while NoC. Indeed, the total area overhead for the NoC is 24.8% and 27.6%, for 32- and 64-bit flits, respectively. Regarding the power overhead, at the NoC scale, it requires 21.1% and 19.9% more power, or 32- and 64-bits flits, respectively.

7 CONCLUSION

In this paper, we present BiRoPro, an adaptive routing algorithm with bit-shuffling for NoC based Approximate Computing. Our proposal is able to manage communications in a faulty NoC with respect to permanent fault positions and data type by distinguishing fault-tolerant and critical packets. The results demonstrate that the proposed approach allows to maintain all the communication paths within a faulty NoC where classic routing algorithm such XY or adaptive algorithm cannot. Moreover, we performed a comprehensive analysis to illustrate the robustness of our algorithm with respect to the number of faults injected by exploring the packet latency and saturation packet injection rate (S_{PIR}). We show that our proposed algorithm performs better while the number of fault-tolerant packets increases, but also performs well with critical data. Finally, in terms of overheads, the proposed solution only requires 27.6% and 19.9% area and power overheads, respectively, for a 64-bit NoC.

ACKNOWLEDGMENTS

This work is supported by the ANR SHNoC project, grant ANR-18-CE25-0006 of the French Agence Nationale de la Recherche.

REFERENCES

[1] 2016. *Space Product Assurance: Techniques for Radiation Effects Mitigation in AASIC and FPGAs Handbook.* Technical Report. ESA Requirements and Standards Division.

[2] M. T. Bohr. 2018. Logic Technology Scaling to Continue Moore's Law. In *Electron Devices Technol. and Manuf. Conf. (EDTM).* IEEE, 1–3. https://doi.org/10.1109/EDTM.2018.8421433

[3] W.J. Dally and B.P. Towles. 2004. *Principles and Practices of Interconnection Networks.* Elsevier.

[4] A. DeOrio, D. Fick, V. Bertacco, D. Sylvester, D. Blaauw, J. Hu, and G. Chen. 2012. A Reliable Routing Architecture and Algorithm for NoCs. *IEEE Trans. on Comput.-Aided Des. of Integr. Circuits and Syst. (TCAD)* 31, 5 (May 2012), 726–739. https://doi.org/10.1109/TCAD.2011.2181509

[5] E. Dubrova. 2013. *Fault-Tolerant Design.* Springer.

[6] M. Ebrahimi, M. Daneshtalab, J. Plosila, and H. Tenhunen. 2013. Minimal-Path Fault-Tolerant Approach Using Connection-Retaining Structure in Networks-on-Chip. In *Int. Symp. on Networks-on-Chip (NOCS).* IEEE/ACM, 1–8. https://doi.org/10.1109/NoCS.2013.6558401

[7] R. Fernandes, C. Marcon, R. Cataldo, and J. Sepúlveda. 2020. Using Smart Routing for Secure and Dependable NoC-Based MPSoCs. *IEEE/ACM Trans. on Netw. (TON)* 28, 3 (June 2020), 1158–1171. https://doi.org/10.1109/TNET.2020.2979372

[8] B. Fu, Y. Han, H. Li, and X. Li. 2014. ZoneDefense: A Fault-Tolerant Routing for 2-D Meshes Without Virtual Channels. *IEEE Trans. on Very Large Scale Integration (VLSI) Syst.* 22, 1 (Jan. 2014), 113–126. https://doi.org/10.1109/TVLSI.2012.2235188

[9] K. Khalil, O. Eldash, A. Kumar, and M. Bayoumi. 2019. Self-Healing Hardware Systems: A Review. *Microelectron. J.* 93 (Nov. 2019), 104620. https://doi.org/10.1016/j.mejo.2019.104620

[10] S. Kundu and S. Chattopadhyay. 2014. *Network-on-Chip: The Next Generation of System-on-Chip Integration.* Taylor & Francis.

[11] G. Liva, L. Gaudio, T. Ninacs, and T. Jerkovits. 2016. Code Design for Short Blocks: A Survey. *Comput. Res. Repository (CoRR) - arXiv* (Oct. 2016). arXiv:1610.00873

[12] R. Mercier, C. Killian, A. Kritikakou, Y. Helen, and D. Chillet. 2021. BiSuT: A NoC-Based Bit-Shuffling Technique for Multiple Permanent Faults Mitigation. *IEEE Trans. on Comput.-Aided Des. of Integr. Circuits and Syst. (TCAD)* (2021), 1–1. https://doi.org/10.1109/TCAD.2021.3101406

[13] H. J. Mohammed, W. N. Flayyih, and F. Z. Rokhani. 2019. Tolerating Permanent Faults in the Input Port of the Network on Chip Router. *J. of Low Power Electron. and Appl.* 9, 1 (Feb. 2019), 1–11. https://doi.org/10.3390/jlpea9010011

[14] L.H. Mutuel. 2016. Single Event Effects Mitigation Techniques Report. *Federal Aviation Admin.* 15, 62 (Feb. 2016), 470.

[15] Umit Y. Ogras, Paul Bogdan, and Radu Marculescu. 2010. An Analytical Approach for Network-on-Chip Performance Analysis. *IEEE Trans. on Comput.-Aided Des. of Integr. Circuits and Syst. (TCAD)* 29, 12 (Dec 2010), 2001–2013. https://doi.org/10.1109/TCAD.2010.2061613

[16] A. Sánchez-Macián, P. Reviriego, and J. A. Maestro. 2014. Hamming SEC-DAED and Extended Hamming SEC-DED-TAED Codes Through Selective Shortening and Bit Placement. *IEEE Trans. on Device and Mater. Rel.* 14, 1 (Mar. 2014), 574–576. https://doi.org/10.1109/TDMR.2012.2204753

[17] H.S. Stone. 1971. Parallel Processing with the Perfect Shuffle. *IEEE Trans. Comput.* C-20, 2 (1971), 153–161. https://doi.org/10.1109/T-C.1971.223205

[18] D. Xiang, K. Chakrabarty, and H. Fujiwara. 2016. A Unified Test and Fault-Tolerant Multicast Solution For Network-on-Chip Designs. In *IEEE Int. Test Conf. (ITC).* 1–9. https://doi.org/10.1109/TEST.2016.7805827

[19] J. Xu, W. Wolf, J. Henkel, and S. Chakradhar. 2005. A Methodology for Design, Modeling, and Analysis of Networks-on-chip. In *Int. Symp. on Circuits and Syst. (ISCAS)*, Vol. 2. IEEE, 1778–1781.