H_2A: A Hardened Asynchronous Network on Chip

Julian Pontes[1,2], Ney Calazans[1], Pascal Vivet[2]
Faculty of Informatics - FACIN, - PUCRS[1]
Porto Alegre, RS, Brazil
ney.calazans@pucrs.br

CEA-LETI[2]
Grenoble, France
{julian.hilgembergpontes, pascal.vivet}@cea.fr

***Abstract*— One of the next challenges for asynchronous communication architectures is reliability, in the form of robustness to single event effects, when under the impact of particles generated by ionizing radiation. This occurs because technology downscaling continuously increases the logic sensitivity of silicon devices to such effects. Contrary to what happens in synchronous circuits, delay variations induced by radiation usually have no impact on asynchronous quasi-delay insensitive combinational logic blocks, but in storage devices, bit flips may corrupt the circuit state with no recovery solution, even when using asynchronous circuits. This work proposes a new set of hardening techniques against single event effects applicable to asynchronous networks-on-chip. It presents practical case studies of use for these techniques and evaluates them in close to real life situations. Obtained results show that the achieved increase in asynchronous network-on-chip robustness has the potential to leverage this communication architecture solution as the main choice for the next generations of complex silicon devices on advanced nodes technologies such as 32 nm, 28 nm and below.**

Keywords—asynchronous circuits; networks on chip; soft error; single event effect

I. INTRODUCTION

An energetic particle crossing a semiconductor junction generates a track of electron/holes pairs. When the victim node is in reverse bias, the electrical field present in a PN junction can collect the ions generated by the particle strike. Depending on the energy of the particle and on the electrical field strength, a particle strike can cause different single event effects (SEEs). SEEs can be divided in two main categories: hard errors and soft errors [8]. Hard errors are permanent damage in a MOS device. A soft error however can be a pulse transient at some combinational logic device (a Single Event Transient or SET) or a bit-flip in some storage element (a Single Event Upset, SEU). Technology downscaling continuously increases logic sensitivity of devices to such effects [6].

Asynchronous circuits have shown better response under SEEs than their synchronous counterparts [12]. Contrary to what happens in synchronous circuits, delay variations induced by radiation usually have no impact on asynchronous quasi-delay insensitive (QDI) circuits. However, bit flips may corrupt data transmissions and stall the circuit with no recovery solution. Due to its delay immunity, asynchronous circuits are gathering attention, particularly for Networks on Chip (NoCs) implementation [4]. Asynchronous NoCs can provide a high performance communication architecture well suited for Globally Asynchronous Locally Synchronous (GALS) SoCs [4]. In these, synchronous IPs communicate asynchronously [3]. This removes the need of a global clock tree, reducing synchronization problems and the power associated to clock distribution trees. By providing both, delay variation immunity and robustness to SEEs, asynchronous NoCs reach the status of the most promising communication architecture for SoCs in deep submicron technologies.

This work proposes a new asynchronous network on chip architecture robust to SEEs. Hardening techniques are applied at the handshake and packet protocol levels, to guarantee the control robustness, thus reducing NoC stall probability.

The rest of this work is organized as follows. Section II presents some basic concepts about asynchronous circuits, to help understanding the work. Section III discusses previous work. Section IV discusses the asynchronous NoC used here as base architecture. Section V presents the hardened version of the base asynchronous NoC. Section VI presents some experimental results and their discussion. Finally, Section VII brings a set of conclusions and directions for future work.

II. ASYNCHRONOUS CIRCUITS

Asynchronous circuits are a class of circuits that do not use clocks to implement sequencing. The absence of a global or some local clock signals is in fact the only characteristic shared by all asynchronous circuits. This class of circuits is indeed a vast territory containing a large number of design choices [14]. Delay Insensitive (DI) circuits are the most robust asynchronous sub-class. In this sub-class, the delay in the wires and/or gates do not affect the circuit behavior. This subclass however, is very limited [14]. By assuming the use of isochronic forks, i.e. forks where all branches have delays within a given time interval, the DI circuits class can be expanded to the encompassing Quasi Delay Insensitive (QDI) class. Using QDI enables the implementation of complex data processing circuits. Data flow QDI circuits operation is based on the combined use of handshake protocols and DI data codes. Section II-A explains details these two mechanisms.

A. Handshake Protocol

Local handshake is the widest adopted protocol in asynchronous circuits. In fact, this protocol is not exclusively employed in asynchronous circuits. Synchronous and GALS circuits employ such communication protocol. A basic handshake protocol employs two wires, Request and Acknowledge, and can be implemented in several different ways [14]. A commonly used version in QDI circuits is the four-phase handshake protocol, also called Return to Zero. Here, a Request initiates communication. Acknowledge informs that the Request signal was received and the required operation was,

978-1-4799-1132-5/13/$31.00 © IEEE

or will be, performed. Request is then removed and in response to it, Acknowledge is also removed. Data signals, when present, are enclosed by the control signals changes, and are free to switch before activating the control signals of the protocol. To guarantee correct operation, events in control signals must be signaled just when the data interface is in a stable state.

B. Delay Insensitive Codes

To remove the timing constraint between Request and Data signals in a handshake protocol, data can be encoded using a DI code. In this class of codes, validity of data is embedded within the data itself. In this way, the Request signal is no longer necessary, since data validity can be extracted by inspecting the Data signals. A Completion Detector is a circuit that detects data validity in QDI circuits.

The delay insensitivity property of a code is guaranteed when the code is *unordered* [1]. A binary code C is unordered when none of its codewords is contained in another valid codeword of the same code [15]. This work assumes the use of *m-of-n DI codes*. This is a class of unordered codes which uses *n* bits. Any valid m-of-n codeword has exactly *m* bits equal to '1'. Table 1 shows an example, the one-of-four DI code. This code uses four bits to carry up to two data bits of information.

Table 1 – The one-of-four code definition, and its spacer.

Value	A3	A2	A1	A0
Spacer	0	0	0	0
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0

When a DI Code is associated to the four-phase handshake protocol, it is necessary the definition of a special codeword to represent the absence of data, referred here as a *spacer* (see Table 1). In four-phase protocols, a spacer is necessary between every two data to complete the protocol. Figure 1 shows a DI data transmission using the one-of-four code and a four-phase handshake protocol. Note that spacers are not valid codewords.

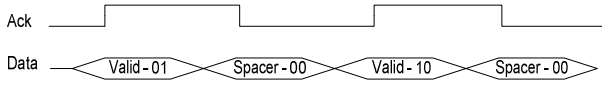

Figure 1 – Four-phase handshake protocol.

C. The C-element

The C-element is one important logic component for most, if not all asynchronous design styles. It is used in practically all such design styles as well as in some hardened synchronous designs [7]. This component enables to implement event synchronization, by producing a change at its output only after a given set of input events occur at its inputs. In practice, a basic C-element copies the value at its inputs to the output when all inputs are either 0 or 1, otherwise it keeps its prior output value. Figure 2 shows a state graph for a 2-input C-element. In Figure 2 states are labeled according to the sequence Input-Input-Output. Solid lines represent the normal C-element behavior. Dashed lines represent its behavior when an SEE occurs in each particular state. Depending on the value of its inputs, a C-element can act as a combinational cell (in states 000 and 111) or as a memory element (in the remaining states). In a combinational state, the C-element can be victim of a SET while when in memory states it can be an SEU victim, as Figure 2 shows.

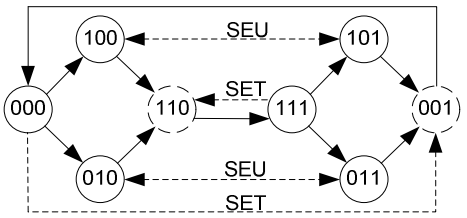

Figure 2 –C-element state transition graph showing possible SEEs.

III. Related Work

Most works about hardening techniques against soft error in NoCs limit application to error correction or detection schemes in the data links. The work of Murali et al. [9] presents retransmission schemes that can be applied between routers (switch-to-switch), or at Sender and Receiver (end-to-end). Another way to prevent data errors in NoC links is using Forward Error Correction (FEC) as proposed in [17]. The work of Rossi et al. [13] presents different levels of data error protection, enabling correction and/or detection. Depending on the requirements, errors can be detected, corrected by single error correction (SEC), single error correction double-error detection (SECDED) or even *symbol* error correction. The work of Yu and Ampadu [17] also proposes data correction at switch-to-switch and end-to-end levels. None of these implementations proposes hardening techniques for the control circuits in routers. In this way, errors can easily cause the NoC to stall, even when data correction is applied.

The work of Frantz et al. [5] presents a synchronous NoC evaluation under the presence of faults generated by SEU and/or crosstalk. Authors evaluate hardware and software mitigation techniques for a NoC with 8-bit data flits. They applied hardening techniques to data and control logic. At the data level, the proposed techniques use Hamming codes or Cyclic Redundancy Check. At the control level, triple modular redundancy (TMR) combines with a triple sampling scheme, where the TMR employs a combinational voter inside the circuit. Since the authors overlooked SETs, they consider the voter to be fault free, which leads to optimistic soft error rates.

All above mentioned works assume synchronous NoC implementations. Just a few works propose hardening techniques for asynchronous NoCs implementations. Agyekum and Nowick [1] propose an unordered DI code that enables two-bit error detection and single bit error correction capabilities. However, this code is difficult to implement in a fully QDI way and becomes complex for large words, due to the size of the completion detection circuit. Bainbridge and Salisbury [2] propose a set of techniques applicable to QDI NoC links, particularly for links based on m-of-n codes, to reduce glitch sensitivity due to crosstalk. The techniques they propose are mostly sample filtering techniques, to reduce glitches in data signals and in some cases in acknowledge signals. However, none of these is adequate for SEEs. The work of Yebin et al. [16] presents hardening techniques to the inter-chip communication using the SpiNNaker NoC. Results show a reduction in

NoC deadlocks due to glitches in the inter-chip link. However, no technique is proposed in this work to address problems likes SEEs and glitches arriving inside the chip. The authors propose new techniques to deal with errors in the NoC control and reduce the probability of stalls and packet errors.

IV. THE ASYNCHRONOUS NOC

The HermesA NoC [11] is used as the base communication architecture in this work. HermesA routers comprise up to five input and output ports. Ports are named North, South, East, West and Local, as Figure 3 shows. The Local port enables communication between the router and its IP Core. Remaining ports provide communication between neighbor routers.

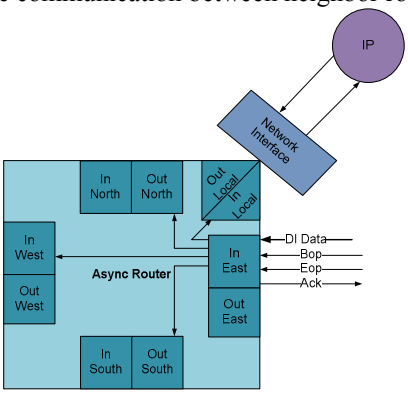

Figure 3 – The HermesA router basic architecture.

Packets in HermesA are delimited by two special sideband signals, Begin of Packet (BOP) and End of Packet (EOP), as Figure 3 details. In the HermesA one-of-four NoC implementation, these two signals are encoded in a single one-of-four codeword, called ***Flit Type***. They are responsible for indicating the first and last flits of a packet. Source routing is employed in this work, differently from the implementation described in [11], which uses distributed routing. The first flit contains routing information and is signaled by the BOP signal. The last flit is delivered together with the EOP signal, which acts to release NoC allocated resources. Using these sideband signals, it is possible to have packet sizes with any positive size (>1). The Input Port in HermesA consists in two modules: Path Definition and Packet Dispatcher/Section Closing. The Input Port in this work has one data input and four possible data outputs. A North Input Port for example, can send incoming packets to the South, East, West or Local ports. The routing of a packet to the opposite direction usually does not make sense. The Input Port is responsible for determining the appropriate Output Port for the incoming packet. The Path Definition Module does this when the Input Port receives the first flit. In the source routing implementation, each two bits (a single 1-of-4 codeword) carry a routing decision. The HermesA Input Port extracts the most significant 1-of-4 codeword from the flit as the routing decision and rotates the flit. Only the Flit Type, at the least significant 1-of-4 codeword, is left untouched by the flit rotation in the Path Definition module. In this way, the next router will find in the most significant 1-of-4 codeword the next routing decision.

The first flit creates a Communication Section, i.e. allocates some NoC resources (Input/Output ports). The use of the wormhole switching mode requires resource allocation that creates a reserved path, similar to what happens when using circuit switching. The main difference is that here small pieces of information (the so-called *flits*) are assumed. Thus, the virtual circuit created by the first flit usually do not hold the resources for long periods. The last flit releases the components for other packets inside the NoC. The Packet Dispatcher/Section Closing module performs the task of resource releasing.

The HermesA Output Port is responsible for controlling the access to shared resources. In this case, the resource is either the next router Input Port or the external NI at the router Local Port. To perform this control action, it is necessary to implement arbitration mechanisms that decide which incoming traffic to serve at each specific moment. The first flit generates a request to the arbiter. The arbiter chooses one among the incoming requests to grant. The last flit releases the arbiter and ends packet transmission.

V. THE HARDENED HERMESA (H_2A) NOC

The construction of a circuit robust to SEEs is a complex task, due to the number of considerations needed to ensure that the module behavior can resist to the many possible distinct effects. A coarse classification of such effects splits them into effects on data and control parts of the module, because each of these are amenable to treatment by distinct robustness enhancement techniques. Both type of errors are important and must be treated. This work focuses on the control part of asynchronous NoCs since this issue is less explored than data error correction and the traditional synchronous hardening techniques cannot be explored in asynchronous circuits. An error in the control can make an Input or Output port to stall. When using wormhole switching this local error can easily propagate and cause a full NoC stall. Then, the only recovering method is to reset the full NoC and possibly all network interfaces. In HermesA, control is split in two distinct protocols levels: the four-phase handshake protocol and the packet protocol levels:

- The handshake protocol level – Most asynchronous handshake protocols employ sequential components that may corrupt the protocol operation when subject to SEEs. Specific techniques may be designed to guarantee reliability to the protocol under the effect of SEEs;

- The packet protocol level – Specifically in NoCs, the reliability of the first flit containing the routing information and the sideband signals BOP/EOP is crucial. Devising techniques to enhance link reliability is thus quite important.

A. Handshake Protocol Level Hardening

This work proposes a modification of the asynchronous weak conditioned half buffer (WCHB) [14] to change timing windows, thus increasing four-phase protocol robustness. The name of the proposed technique is Normal Open Asynchronous Pipeline (NOAP). The main idea is to synchronize data and acknowledge at each buffer stage.

Figure 4 shows a 3-stage pipeline implementation using the NOAP technique.

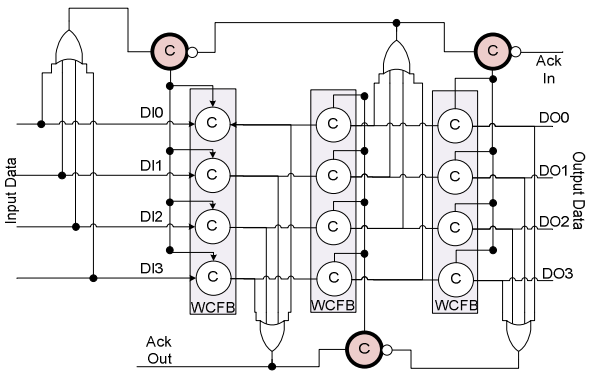

Figure 4 – Normal open asynchronous pipeline implementation.

The only difference of this implementation with regard to the classical asynchronous WCHB is the addition of a C-element to each stage (emphasized in the Figure) to synchronize data and acknowledge signals. In the initial state, Input Data=0000; all WCHB C-elements have their inputs equal to 00 and Output Data=0; in all stages the additional C-element (referred here as NOAP C-element) has inputs at 01 (0 from the pre-buffer detection and 1 from the acknowledge signal). Since the NOAP C-element was initialized (the existing global reset signal is omitted in the Figure, for clarity), the output is equal to 0. In this state, the WCHB C-elements are transparent to the spacer. No data can cross the buffer before the data/acknowledge synchronization. During the initial state, just a SET↑ can happen in the buffer C-elements but this SET is filtered by the next stages of the pipeline that are in the same state.

Once new data arrives, these must be detected by the completion detector (the OR gate) and the result of the detection will enable the NOAP C-element. The NOAP C-element then changes the buffer state. The later becomes transparent to data like a normal buffer implementation. The main difference is that when the acknowledge arrives, the NOAP C-element do not close the asynchronous buffer. This buffer stays open until the synchronization between data and acknowledge takes place. Thus, under normal operation the C-element is open. The main advantage of the NOAP implementation is its filtering property, mainly in the empty state. This protocol modification brings two main advantages to pipeline robustness:

1. Circuit electrical filtering increase: In the NOAP handshake protocol, C-elements that are part of the WCHB stay in states 111 and 000. In a conventional asynchronous pipeline implementation, C-elements inputs remain mostly in 01X and 10X states, where X´s (the output value) can be either 0 or 1. The states where the C-element has inputs at the same logic level have higher critical charge. In this way, the NOAP technique increases the critical charge needed to cause SEEs as well.

2. Circuit logic filtering increase: The main characteristic of NOAP is the change in timing windows of the four-phase handshake protocol. NOAP timing windows reduce the handshake stall probability. It also reduces the probability that an SEU occur in the asynchronous buffer, since the most probable C-element state is 000 (a state susceptible to SETs). This removes dormant errors inside the input and output ports, mainly in the idle state, consequently protecting the first flit, which contains the routing information. Since C-elements of the asynchronous buffer are enabled just when both data and acknowledge arrive, NOAP has the ability to logically filter SET transients in the data signals, as well as in the acknowledge wire.

The main drawback of NOAP is the additional propagation delay in the forward and backward paths. The consequences are an increase in pipeline cycle time and a consequent reduction in throughput, as well as an increase in latency.

B. Packet Level Hardening

Just as the handshake protocol, the packet protocol sequence can also suffer with SEEs. This protocol is controlled by two sideband signals (BOP and EOP) that designate the flit type in the HermesA NoC. In addition, the packet protocol depends on the routing information contained in the first flit. To increase packet protocol robustness, this work suggests applying the double check technique [7] to BOP/EOP, as well as to the routing decision.

1) Double Check

Double check is a spatial redundancy technique proposed by Jang and Martin [7] to filter SEEs mostly in combinational logic. This technique consists in duplicating the logic and performing a logic check using the event synchronization property of C-elements. Figure 5 shows an example of the double check technique applied to a 3-input logic. In the Figure, Logic A and Logic B are two instances of a circuit. Inputs of both circuits are also the same, i.e. (XA=XB, YA=YB and ZA=ZB). The C-elements with outputs QA and QB synchronize the results of the logic. Updates in outputs QA and QB only occur when the C-element inputs are equal. In this way, the C-elements filter any SET in Logic A or in Logic B. Since during normal operation each C-element input pair has always the same value, C-elements may only present SETs. The problem of using double check is the area overhead. Since this technique can filter SETs, it is well suited to work combined with NOAP, where the handshake behavior increases the SET and shrinks the SEU window.

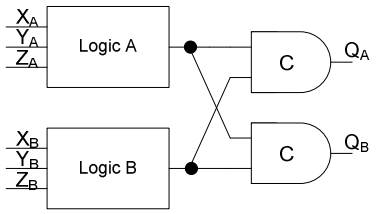

Figure 5 - Example of the double check technique applied to a 3-input single output logic.

Inside the Input and Output ports, BOP and EOP signals are responsible for packet control and by the flit flow, as discussed earlier. An SEE in these signals may alter the correct data path inside the NoC. An SEE in the BOP signal may, for example, propagate a regular flit to the Path Definition module and thus forward part of a packet to the wrong target. A wrong indication on the EOP signal may release the arbitration mechanism. Therefore, a packet can be interrupted and reach the correct target incomplete.

In this work, the flit type (BOP/EOP) information is duplicated to increase the robustness of the packet control. Figure 6 shows the double check scheme adopted for the flit type and routing information. In the Figure, there are two incoming one-of-four codewords. Both codewords represent the same data since these were duplicated (A0 = B0; A1 = B1; A2 = B2; A3 = B3). The demux decision is extracted after double check, where the two codewords related to the flit decision are then restored. Thus, if an SEE occurs, double check filters it.

The routing decision is part of the first flit in each packet. It controls the path that the packet crosses inside the NoC. In this way, an SEE that changes the routing decision may forward the packet to a wrong target. Hardening of the routing decision also employs double check. This is applied to the source routing mechanism, as Figure 6 details. Thus in the H_2A NoC each Input Port extracts the ***two*** most significant one-of-four codewords and performs double check on them. The result controls the output multiplexer of the Input Port and is kept in the routing register until packet transmission ends. The routing register is also duplicated to increase packet robustness. The main drawback is that duplication of each routing decision reduces the number routers that a packet can cross inside the NoC, due to routing information duplication. A HermesA 32-bit flit has 16 one-of-four codewords carrying up to 16 routing decisions. It consequently can cross up to 16 routers before reaching the final target. In H_2A, a 32-bit flit is composed also by 16 1-of-4 codewords, but because routing info is duplicated, a packet can cross only 8 routers.

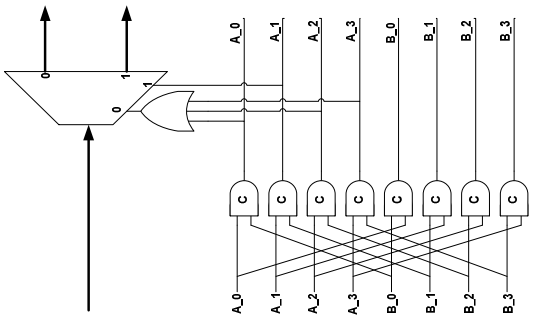

Figure 6 – Double check scheme adopted for the BOP/EOP and routing information.

VI. RESULTS

This Section presents the results of SEE evaluation comparing HermesA to three different H_2A NoC versions: one using just flit type double check (Flt Type DC), one using just NOAP (NOAP) and one combining both techniques (Flit Type+NOAP). SEE evaluation follows the method proposed by the authors in [10], where an SEE victim cell is randomly chosen during timing annotated simulation. Simulation occurs after place and route of one router using timing annotation. The entire standard cell library was characterized to extract electrical responses to SEEs as [10] suggests. This work uses a 65nm bulk technology. The SEE evaluation assumes conditions of 1 Volt supply, 25°C, and typical process parameters. These operating conditions apply to both the timing cell description, and the SEE standard cell characterization process.

Traffic was injected at one of the input ports. The injected packet rate is $10x10^6$ packets/second, with 16-flit packets. One SEE injection was performed at each packet. For this evaluation, $100x10^3$ packets and $100x10^3$ SEEs were injected. Figure 7 shows the relation between the Number of Data Errors and the Injected Charge for the four distinct implementations.

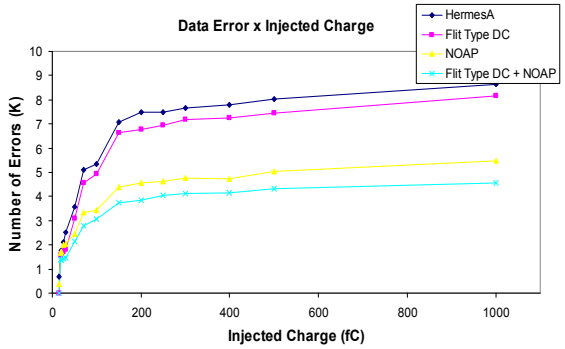

Figure 7 - Relationship between injected charge the Soft Error Rate for different implementations of the H_2A NoC.

In the graph, it is possible to note that the use of the NOAP combined with the flit type double check (Flit Type DC in the graph) technique displays the best results for all levels of injected charge. All implementations present similar curves, where it is possible to distinguish a point where the increase in the injected charge does not increase the error rate. This point is close to the C-element critical charge level. This means that the injected charge level has less impact in the soft error after exceeding the critical charge of the C-elements. The benefits of the proposed NoC hardening solutions are clearer when looking to the number of errors during packet transmission. Figure 8 shows these packet errors. Packet errors are errors in the packet reception, like incomplete packets, wrongly routed packets, wrong flit indication and unexpected packets. Again, the combined use of Double Check in control and the NOAP technique reduced about 45 times the error rate.

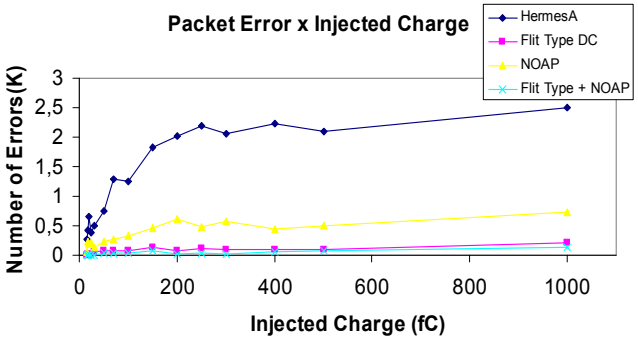

Figure 8 - Relationship between the number of packet errors and the injected charge for different implementations of the H_2A NoC.

Figure 9 shows the classification of the errors as a function of the victim flit. In this Figure, it is possible to note the capacity of the NOAP technique to filter errors when Ports are in the idle state. This observation relies on the number of errors in the first flit. The first flit carries faults that are dormant in the circuit during the idle state of the NoC. The NOAP technique filters this type of errors by enabling SETs instead of SEUs. Since the first flit carries the routing information, the use of NOAP increases the robustness of this control information, reducing the number of packet errors as well.

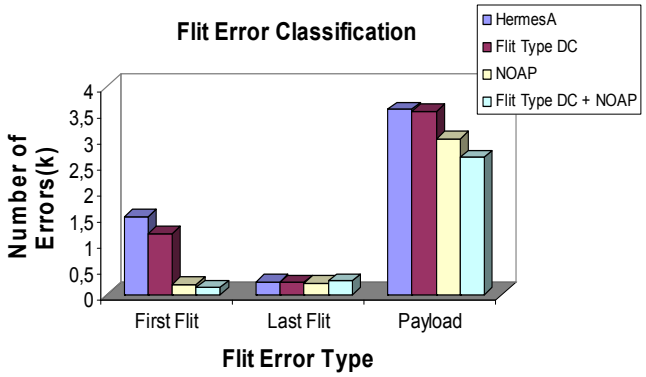

Figure 9 - Flit corruption evaluation.

Figure 10 shows the stall occurrences. The best technique to reduce stall, as predicted, is double check. However, it is interesting to note that even without redundancy, the NOAP technique displays good results. Stall reduction results from the filtering property during the idle state. This in turn reduces the number of faults in the flit type signals and therefore reduces the number of errors in the packet control flow. NOAP also protects the four-phase handshake protocol and from SETs in the acknowledge signal.

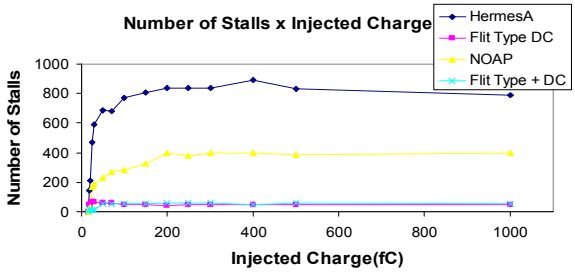

Figure 10 - Evaluation of the stall errors as a function of the injected charge for different implementations of the H_2A NoC.

Table 2 shows an overview of each technique and the area overhead for a router when compared to the HermesA router. The area overhead of both techniques is negligible when compared to traditional redundancy techniques like full double check.

Table 2 – Area comparison and hardening overview.

Design Level	Technique	Hardening	Target Error	Area Overhead
Handshake Level	Protocol Adaptation	Timing Filtering	Data and Packet	4.9%
Packet Level	Double Check	Logic Filtering	Packet and Stall	6.7%

Performance, latency and throughput are affected by both techniques. For the NOAP, an additional C-element is part of the forward (data) and backward (acknowledge) path. Double check also adds one C-element in the forward path of the demux control signal.

VII. Conclusions and Future Works

The timing robustness of QDI circuits gives asynchronous NoCs the robustness needed to help SoC designers to tackle problems that arise in modern deep submicron technologies. By studying the radiation effects over asynchronous NoCs, this work helps increasing the robustness of such communication architectures against soft errors as well. Results show that the H_2A reduces about 10 times the probability of stall in the NoC when compared to HermesA. Results also show a small area and consequently static power overhead associated with the proposed techniques. Future work are performance evaluation and full characterization of entire H_2A NoC instances.

VIII. References

[1] Agyekum, M. Y.; Nowick, S. M. “An error-correcting unordered code and hardware support for robust asynchronous global communication.” In: DATE’11, pp. 765-770, 2011.

[2] Bainbridge, W. J.; Salisbury, S. J. “Glitch sensitivity and defense of quasi delay insensitive network-on-chip links.” In: ASYNC’09, pp. 35-44, 2009.

[3] Chapiro, D. “Globally-Asynchronous Locally Synchronous Systems.” PhD Thesis, Stanford University, Oct. 1984, 134 p.

[4] Clermidy, F.; Cassiau, N.; Coste, N.; Dutoit, D.; Fantini, M.; Ktenas, D.; Lemaire, R.; Stefanizzi, L. “Reconfiguration of a 3GPP-LTE telecommunication application on a 22-core NoC-based system-on-chip.” In: NoCS’11, pp. 261-262, 2011.

[5] Frantz, A. P.; Cassel, M.; Kastensmidt, F. L.; Cota, E.; Carro, L. “Crosstalk and SEU-Aware Networks on Chips.” IEEE Design & Test of Computers, 24(4), pp. 340-350, Jul.-Aug. 2007.

[6] International Technology Roadmap for Semiconductors, http://www.itrs.net/, 2011.

[7] Jang, W.; Martin, A. “SEU-tolerant QDI circuits”. In: ASYNC’05, pp. 156-165, 2005.

[8] Mukherjee, S. “Architecture Design for Software Errors.” Morgan Kaufmann Publishers, Burlington, 2008. 337p.

[9] Murali, S.; Theocharides, T.; Vijaykrishnan, N.; Irwin, M.J.; Benini, L.; De Micheli, G. “Analysis of error recovery schemes for networks on chips.” IEEE Design & Test of Computers, 22(5), pp. 434- 442, Sept.-Oct. 2005.

[10] Pontes, J. J. H.; Vivet, P.; Calazans, N. L. V. “An Accurate Single Event Upset Digital Design Flow for Reliable System Level Design”. In: DATE’12, pp. 224-229, 2012.

[11] Pontes, J. J. H.; Moreira, M. T.; Moraes, F. G.; Calazans, N. L. V.: “Hermes-AA: A 65nm Asynchronous NoC Router with Adaptive Routing”. In: SOCC’10, pp. 493-498, 2010.

[12] Rahbaran, B.; Steininger, A. “Is asynchronous logic more robust than synchronous logic?” IEEE Transactions on Dependable and Secure Computing, 6(4), pp. 282-294, Dec. 2009.

[13] Rossi, D.; Angelini, P.; Metra, C. “Configurable Error Control Scheme for NoC Signal Integrity.” In: IOLTS’07, pp. 43-48, Jul. 2007.

[14] Sparsø, J.; Furber, S. “Principles of Asynchronous Circuit Design – A Systems Perspective.” Kluwer Academic Publishers, Boston, 2001. 354p.

[15] Bose, B. “On Unordered Codes.” IEEE Transaction on Computers, 40(2), pp. 125-131, Feb. 1991.

[16] Yebin Shi; Furber, S.B.; Garside, J.; Plana, L.A. “Fault Tolerant Delay Insensitive Inter-chip Communication”. In: ASYNC’09, pp.77-84, 2009.

[17] Yu, Q.; Ampadu, P. “Dual-Layer Adaptive Error Control for Network-on-Chip Links.” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 20(7), pp. 1304-1317, Jul. 2012.