This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LCA.2014.2360686, IEEE Computer Architecture Letters

A Soft Error Tolerant Network-on-Chip Router Pipeline for Multi-core Systems

Pavan Poluri and Ahmed Louri
Department of Electrical and Computer Engineering, University of Arizona
Email: pavanp@email.arizona.edu, louri@email.arizona.edu

Abstract—Network-on-Chip (NoC) paradigm is rapidly evolving into an efficient interconnection network to handle the strict communication requirements between the increasing number of cores on a single chip. Diminishing transistor size is making the NoC increasingly vulnerable to both hard faults and soft errors. This paper concentrates on soft errors in NoCs. A soft error in an NoC router results in significant consequences such as data corruption, packet retransmission and deadlock among others. To this end, we propose *Soft Error Tolerant NoC Router (STNR)* architecture, that is capable of detecting and recovering from soft errors occurring in different control stages of the routing pipeline. STNR exploits the use of idle cycles inherent in NoC packet routing pipeline to perform time redundant executions necessary for soft error tolerance. In doing so, STNR is able to detect and correct all single transient faults in the control stages of the pipeline. Simulation results using PARSEC and SPLASH-2 benchmarks show that STNR is able to accomplish such high level of soft error protection with a minimal impact on latency (an increase of 1.7 % and 1.6 % respectively). Additionally, STNR incurs an area overhead of 7% and power overhead of 13% as compared to the baseline unprotected router.

Index Terms—Network-on-Chip, Soft Error, Reliability, Performance

1 Introduction

Network-on-Chip (NoC) [3], [6] is a scalable interconnect architecture comprised of shorter wires and is designed to tackle the increasing wire delay and limited scalability issues of shared buses. An NoC is comprised of routers that perform routing and links that are used for data traversal. With the rapid decrease in the feature size, the components of an NoC are becoming increasingly susceptible to both hard faults and soft errors. Therefore, it is imperative to integrate fault tolerant techniques into the design of reliable NoCs.

In this paper, we focus specifically on soft errors that can occur in an NoC router pipeline as it is responsible for the steady flow of packets through the router. Hard faults are discussed in a different work [11]. We propose *Soft Error Tolerant NoC Router (STNR)* architecture, that is capable of tolerating utmost three independent soft errors in its pipeline. The primary characteristic of STNR is its ability to effectively utilize idle cycles in the pipeline stages for redundant execution and comparison and a rollback in the event of a soft error. Unique attributes of STNR include high level of protection from soft errors, fault containment and no transmission penalty encountered by a packet traversing through the pipeline in the absence of a soft error.

2 NoC Router

Figure 1(a) shows a standard 4x4 mesh topology based NoC. Figure 1(b) [7] illustrates the architecture of an NoC router with *P* input and output ports with each input port comprised of *V* virtual channels. Routing Computation (RC), Virtual Channel Allocation (VA) and Switch Allocation (SA) form the control logic of the router. A PxP crossbar (XB) connects the input ports of the router to its output ports. Data traverses in an NoC in the form of *flits*. A packet is segmented into a *head flit*, *body flit(s)* and a *tail flit*. Figure 1(c) [7] shows the 4-stage pipeline comprised of RC, VA, SA and XB stages.

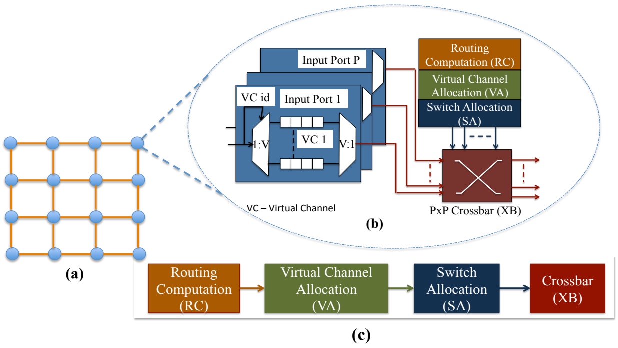


Fig. 1: **(a) 4x4 mesh based NoC (b) NoC Router Architecture (c) NoC Router Pipeline**

The *RC stage* processes head flits and is responsible for selecting the output port based on the destination information in a head flit. The *VA stage* also processes head flits and is responsible for allocating buffer space to the flit at the downstream router. Both RC and VA stages are idle for body and tail flits. The *SA stage* is active on all flits and is responsible for granting access to crossbar for the flits of different input virtual channels. In *XB stage*, flits that have been granted access in the SA stage traverse through the crossbar and leave the router.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LCA.2014.2360686, IEEE Computer Architecture Letters

3 Soft Error Effects on the Pipeline

A soft error in RC stage would result in the calculation of an incorrect output port leading to incorrect executions of VA and SA stages resulting in packet drop or deadlock or increase in the latency. A soft error in VA stage would result in an incorrect virtual channel being allocated to the packet that could result in data corruption leading to retransmission of the packet and an increase in the latency. A soft error in SA stage could result in flits of the same packet being forwarded to different downstream routers. If a body or a tail flit is forwarded to a different downstream router than the head flit, it will be dropped and the entire packet needs to be transmitted. Soft error in the crossbar does not forward the packet to an incorrect downstream router and hence is not as vital as the remaining stages. Thus, it is important to tolerate soft errors in the first three stages of the pipeline, which is precisely the focus of STNR.

4 Related Work and Motivation

In this section, we provide a brief overview of the approaches used to tackle the issue of soft errors in router pipeline.

In [9] the authors discuss the use of cyclic redundancy check codes to perform end-to-end and switch-to-switch error detection. In [8] the authors propose to use look-ahead routing, hamming code, retransmission buffers and compact header to tackle transient faults in the router. In [10] the authors propose to use state information to perform comparisons in one clock cycle to detect transient faults in RC, VA and SA stages. In [12] the authors propose the use of inherent information redundancy present in the router pipeline to detect transient faults in RC stage and arbitration units. In [5] the authors propose to borrow RC units from neighboring input ports for soft error tolerance. They use self-correcting round robin arbitration mechanism proposed in [12] to protect arbiters in SA stage from soft errors.

Our primary motivation for proposing STNR is to develop a low cost technique that has a very high level of soft error detection and that also achieves fault containment for RC, VA and SA stages. The proposed approach is uniquely different from all existing techniques mentioned above because, it can detect and correct all single transient faults in RC, VA and SA stages with minimal impact on area, power and latency. It should be noted that the proposed architecture does not provide protection for the buffers and the crossbar. Buffers are usually well protected by Error Correcting Codes based techniques. The impact of soft errors on the crossbar are not as severe as on the other critical components of the router pipeline [8].

5 Soft Error Tolerant NoC Router

5.1 Proposed Architecture

Careful observation of the router pipeline operation reveals that RC and VA stages are only active while processing head flits and idle for the remaining flits of a packet and STNR exploits these idle cycles to provide soft error protection.

5.1.1 Temporal Redundancy for RC Stage

We propose to modify the control logic of the router state machine to enable the RC unit to repeat its computation in the cycle following the original route computation. Consequently, the redundant execution of the RC stage takes place in parallel with that of the VA stage, which is executed assuming that the original routing computation is error-free. The results of the two executions of the RC unit are compared in the same cycle as that of re-execution and 1) if the results are identical, the pipeline proceeds in the normal manner, 2) if an error is detected, the state machine asserts the necessary control signals that reset the result of VA stage and trigger a rollback that will initiate the re-execution of RC stage in the next cycle. We assume that the rollback computation is error-free as the probability of back to back soft errors in the same computation stage is extremely low (as observed from the soft error rate calculation results in section 8). The temporal redundancy technique for the RC stage costs two cycle delay per soft error.

5.1.2 Temporal Redundancy for VA Stage

The control logic of the router state machine is modified to enable the VA unit to repeat its computation in the cycle following the original virtual channel allocation. The redundant computation is done in parallel with the SA stage that performs its task assuming that the result of VA stage is error-free. The results of the two executions are compared and 1) if the results are identical, the pipeline proceeds in the normal manner, 2) if an error is detected, the state machine asserts the necessary control signals that reset the result of the SA stage and trigger a rollback that will initiate the re-execution of VA stage in the next cycle. The temporal redundancy technique for the VA stage costs two cycle delay per soft error.

5.1.3 Spatial Redundancy for SA Stage

Temporal redundancy cannot be used for the SA stage because this pipeline stage does not have any idle cycles. Additionally, this stage requires much less hardware and area compared to VA stage. Based on these observations, we chose to duplicate the SA unit for providing soft error protection. The switch allocation is performed twice in the original and duplicate units and the results are compared for error detection. The state machine is modified such that in the event of an error, the flit is not propagated to the crossbar and SA stage is repeated in the next cycle. The spatial redundancy technique for the SA stage costs one cycle delay per soft error.

5.1.4 STNR Pipeline

In this sub-section, we describe the working of the STNR pipeline. To simplify the explanation, we provide cycle-by-cycle traversal of a flit in the proposed pipeline. We refer to Figure 2 for this explanation.

Cycle 1 - The original RC is performed and the result is stored in the virtual channel state.

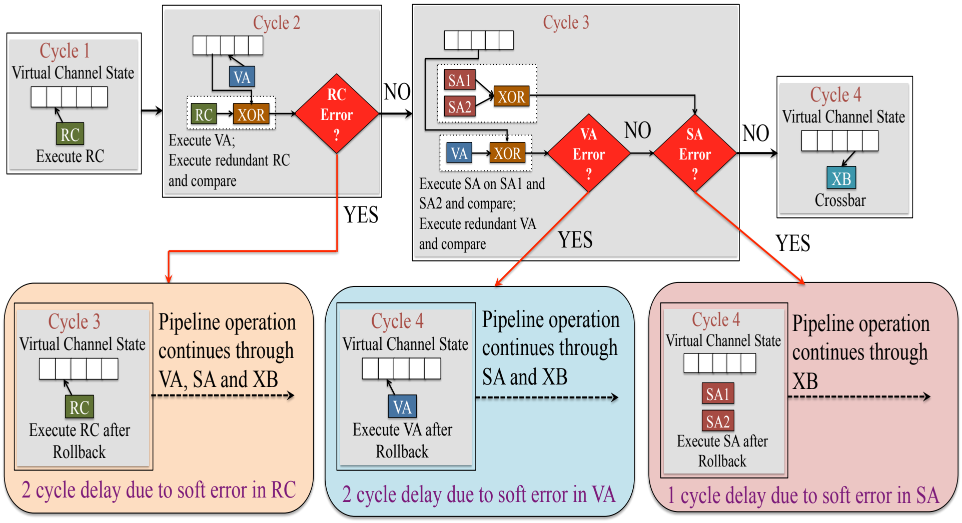


Fig. 2: **Working of the STNR Pipeline**

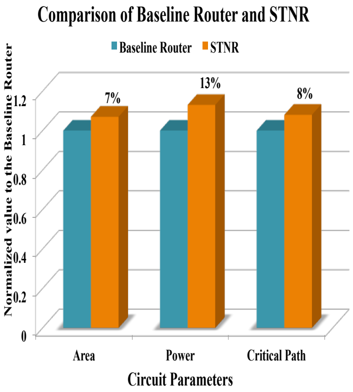


Fig. 3: **Performance Results**

Cycle 2 - The redundant RC is executed in parallel with VA. The result of VA is stored in the VC state. The results of the original and redundant RCs are compared to detect an error in the RC stage using an XOR gate.

Cycle 3 - The state of the router pipeline in this cycle depends on the manifestation of an error in the RC stage. (i) If RC is error free, then the SA stage is active where the two SA units perform arbitration while the redundant VA is done in parallel. (ii) If an error is detected in the RC stage, a rollback is triggered, and the pipeline performs RC in this cycle.

Cycle 4 - Assuming no error is detected in the RC stage, the state of the router pipeline in this cycle depends on the manifestation of an error in either VA or SA stage. Because VA is prior to SA in the generic pipeline, error detection is first performed on VA then followed by SA. (i) If VA and SA are error-free, then the crossbar stage is active and the flit leaves the router. (ii) If VA is error-free, but an error is detected in SA stage, a rollback is triggered and the two switch allocation units repeat allocation in this cycle. (iii) If an error is detected in VA, a rollback is triggered and the pipeline performs VA in this cycle.

5.2 Salient Features of STNR Pipeline

- **High Level of Error Detection and Fault Containment -** Dual execution in conjunction with comparison provides the highest level of error detection, as there is no assumption on the fault. As a result, the protected stage will always detect a single soft error. There are two possible scenarios: *(i) If a soft error has affected a pipeline stage, the comparator will detect this error and trigger a rollback and (ii) If a soft error affects the comparator, it falsely detects the presence of an error in the computation and triggers a rollback*. Hence, the effect of a fault is *limited to the affected router*. Error might only propagate when two soft errors: one in the pipeline stage and one in the comparator affect their execution in the same cycle.
- **Minimum Latency Penalty -** With the protection enabled, a flit incurs no additional latency to traverse the pipeline in the absence of a soft error. The extra latency due to a soft error is 1 cycle for spatial redundant protection and 2 cycles for time redundant protection.

6 Hardware Synthesis Results

We developed in Verilog both the baseline router and STNR with each router comprised of 5-input and output ports with each input port consisting 4 VCs and synthesized using Cadence Encounter Compiler at 45nm technology. Synthesis results reveal that STNR incurs an area, power and critical path overhead of 7%, 13% and 8% with respect to the baseline router (Figure 3).

7 Latency Analysis

7.1 Fault Model

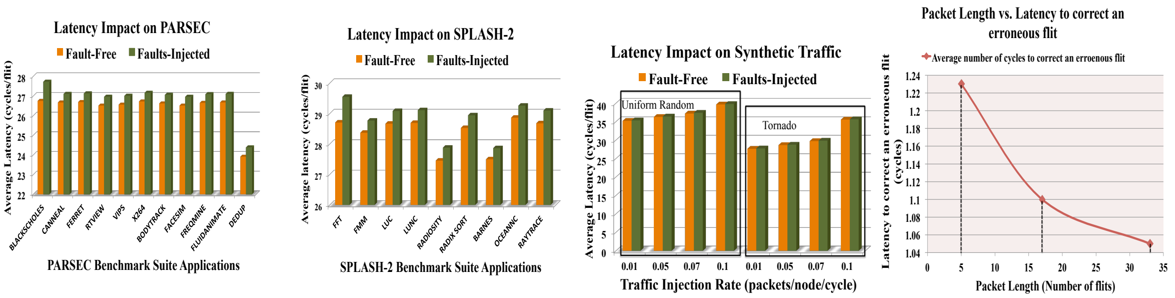
We inject transient faults into different stages of the pipeline based on the probabilities provided by the fault modeling tool of Aisopos et.al [2]. The fault probabilities of the pipeline stages vary based on the router configuration and its operating temperature.

7.2 Results

We use GEM5 [4] and GARNET [1] to simulate an NoC and the 4-stage pipeline. Transient faults are injected during the simulations based on the fault model [2]. The router configuration used for simulating faults is a 5-input, 5-output port router with 4 VCs per input port. The packet length is set to 5 flits.

For benchmark traffic, we simulated a 4x4 mesh based NoC. The routers' operating temperature is selected to be 100°C. Figures 4a (PARSEC) and 4b (SPLASH-2) show that, in the presence of faults, the average latency has increased by 1.7% and 1.6% respectively, compared to the fault-free scenario.

For synthetic traffic, we simulated a 8x8 mesh based NoC. We simulated both *uniform random* and *tornado* traffic patterns with injection rates of *0.01, 0.05, 0.07* and



(a) **4x4 NoC** (b) **4x4 NoC** (c) **8x8 NoC** (d) **8x8 NoC**

Fig. 4: **Latency Results with STNR**

0.1. The routers' operating temperature is selected to be 85°C. Figure 4c shows that, in the presence of faults, the average latency has increased approximately by 0.5% compared to the fault-free scenario.

As the number of flits in a packet increases, the packet consumes significantly more time in SA as compared to RC and VA stages and therefore, has a higher probability of soft error in SA stage. Since, it takes 1 cycle to correct a soft error in the SA stage, as the number of soft errors in the SA stage dominates the total number of soft errors, the average latency to correct an erroneous flit tends to approach 1 cycle (Figure 4d). Thus, longer packets encounter less latency to correct an erroneous flit resulting in increased throughput.

8 Reliability Analysis

In STNR, we consider the transmission of an erroneous flit as failure. This happens only in the following three cases. (i) A soft error in the RC stage and in the XOR gate responsible for error detection in RC. (ii) A soft error in the VA stage and in the XOR gate responsible for error detection in VA. (iii) A soft error in the SA stage and in the XOR gate responsible for error detection in SA. We estimate the probability of these three cases by calculating the soft error rate (SER) of the circuits using the model presented in [13]. In the interest of space, we directly provide the SER values of these stages.

The SER of RC, VA and SA stages is calculated to be $0.72 * 10^{-7}$, $6.33 * 10^{-6}$ and $2 * 10^{-7}$ respectively. The SER of an XOR gate is calculated to be the order of 10^{-8}. Thus, the probability of a soft error in the RC stage as well as in the XOR gate responsible for error detection in RC is calculated as $0.72 * 10^{-7} * 10^{-8} \approx 10^{-15}$. Similarly, the probability of a soft error in the VA stage as well as in the XOR gate responsible for error detection in VA and the probability of a soft error in the SA stage as well as in the XOR gate responsible for error detection in SA are of the order of 10^{-15}. Based on this value, it can be deduced that the probability of two soft errors occurring in the same pipeline stage in the same cycle is very less and since, STNR will always detect a single soft error, it has a very high level of soft error tolerance. This has been observed during the latency simulations, where all the injected soft errors were detected by STNR.

9 Conclusion

Aggressive technology scaling is increasing the vulnerability of transistors to soft errors. Hence, soft error tolerance needs to be considered in designing reliable systems. In this work, we have proposed Soft Error Tolerant NoC Router (STNR), a router capable of tolerating soft errors in the control stages of the pipeline. STNR accomplishes soft error tolerance by performing temporal and spatial redundant executions. Significant characteristics of STNR include fault containment, high level of error detection and minimum latency in transmitting an error-free message. Experimental results show that STNR detects every single soft error and incurs minimum overhead.

Acknowledgment

This research was supported by NSF awards CNS-1318997, ECCS-0725765, ECCS-1342702 and CCF-1420681.

References

[1] N. Agarwal, et.al, "Garnet: A detailed on-chip network model inside a full system simulator," in *ISPASS*, pp. 33-42, 2009.
[2] K. Aisopos et.al, "Enabling system-level modeling of variation-induced faults in networks-on-chips," in *DAC*, pp. 930-935, 2011.
[3] L. Benini and G. Micheli, "Networks on chips: a new SoC paradigm," in *IEEE Computer*, vol. 35, no. 1, pp. 70-78, 2002.
[4] N. Binkert et.al, "The gem5 simulator," in *SIGARCH, Computer Architecture News*, vol. 39, no. 2, pp. 1-7, 2011.
[5] C. Chen, et.al, "A Low Cost Method to Tolerate Soft Errors in the NoC Router Control Plane", in *SOCC*, pp. 374-379, 2013.
[6] W. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks", in *DAC*, pp. 684-689, 2001.
[7] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*, Morgan Kaufmann, 2003.
[8] J. Kim, et.al, "Design and analysis of an NoC architecture from performance, reliability and energy perspective," in *ANCS*, pp. 173-182, 2005.
[9] S. Murali, et.al, "Analysis of error recovery schemes for networks on chips," in *IEEE Design & Test of Computers*, vol. 22, no. 5, pp. 434-442, 2005.
[10] D. Park, et.al, "Exploring Fault-Tolerant Network-on-Chip Architectures," in *DSN*, pp. 93-104, 2006.
[11] P. Poluri and A. Louri, "An Improved Router Design for Reliable On chip Networks," in *IPDPS*, pp. 283-292, 2014.
[12] Q. Yu, et.al, "Exploiting Inherent Information Redundancy to Manage Transient Errors in NoC Routing Arbitration," in *NOCS*, pp. 105-112, 2011.
[13] M. Zhang, et.al, "Soft-Error-Rate-Analysis (SERA) Methodology," in *IEEE TCAD*, vol. 25, no. 10, pp. 2140-2155, October, 2006.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LCA.2014.2360686, IEEE Computer Architecture Letters

Pavan Poluri received the Bachelor of Engineering degree from Osmania University, India in 2007 and Masters of Science degree in Computer Science from University of Minnesota, Duluth, USA in 2009. He is currently pursuing his PhD degree from the High Performance Computing Architectures and Technologies Lab in the Department of Electrical and Computer Engineering at The University of Arizona, Tucson, USA. His research interests include Computer Architecture, Network-on-Chip (NoC), Reliability and Modeling and Simulation. He is currently a student member of IEEE.

Ahmed Louri received the PhD degree in computer engineering in 1988 from the University of Southern California (USC), Los Angeles. He is currently a full professor of electrical and computer engineering at the University of Arizona, Tucson, and the director of the High Performance Computing Architectures and Technologies (HPCAT) Laboratory (www.ece.arizona.edu/˜hpcat). His research interests include computer architecture, network-on-chips (NoCs), parallel processing, power-aware parallel architectures, and optical interconnection networks. He has served as the general chair of the 2007 IEEE International Symposium on High-Performance Computer Architecture (HPCA), Phoenix, Arizona. He has also served as a member of the technical program committee of several conferences including the ANCS, HPCA, MICRO, NoCs, among others. He is a fellow of the IEEE and a member of the OSA.