Failure Analysis of a Network-on-Chip for Real-Time Mixed-Critical Systems

Eberle A. Rambo, Alexander Tschiene, Jonas Diemer, Leonie Ahrendts, and Rolf Ernst
Institute of Computer and Network Engineering
TU Braunschweig, Germany
{rambo|tschiene|diemer|ernst}@ida.ing.tu-bs.de l.ahrendts@tu-bs.de

***Abstract*—Multi- and many-core architectures using Networks-on-Chip (NoC) are being explored for use in real-time safety-critical applications for their performance and efficiency. Such systems must provide isolation between tasks that may present distinct criticality levels. The NoC is critical to maintain the isolation property as it is a heavily used shared resource. To meet safety-standard requirements, such architectures require a systematic evaluation of the effects of all possible failures such as in the form of a Failure Mode and Effects Analysis (FMEA). We present the results of a detailed system-level analysis of a typical real-time mixed-critical network-on-chip architecture. This comprises an FMEA and error effects classification regarding duration and isolation violation.**

I. Introduction

Networks-on-Chip (NoCs) have been proposed as a scalable interconnect for multi- and many-core processors. Such processors are widely used in consumer electronics and are now being evaluated for future real-time safety-critical systems. They offer increased performance and efficiency (with respect to power, area and cost) compared to their single-core counterparts, and also allow the integration of multiple applications with different criticalities, which formerly ran on distinct chips, in a single one [1].

Any device implementing a safety critical function (so called "safety-function") has to pass a certification or qualification process defined in standards such as IEC 61508 [2] and its domain-specific counter-parts ISO 26262 for automotive electric/electronic systems and DO-254 for airborne electronic hardware. Certification typically requires a *Failure Mode and Effects Analysis* (FMEA) [3] which captures all possible system failures that need to be addressed by fault-tolerance mechanisms to ensure that the safety-function performs reliably even in the presence of (uncorrelated) errors. The FMEA starts with a collection of possible faults (e.g. single-event upset in a flip-flop) from which possible errors are derived (e.g. change of the value of a register) and all resulting failures are evaluated (e.g. incorrect routing decision).

In a real-time safety system, it is mandatory to ensure the isolation between critical tasks even in the presence of errors. Therefore, we consider the propagation of a error from one task to another as a failure. The NoC, being used as the central interconnect in such a system, is a critical shared resource and must be analyzed accordingly. A more detailed analysis can help to better understand the errors, the extent of their effects in the NoC in the short and long term, which ones are more likely to occur, and how to prevent/mitigate them. Furthermore, one does not need to be excessively conservative and consider that all unmasked errors lead to failure, as it is conventionally done, since there are effects that are transient and can be safely tolerated.

This paper presents a detailed system-level analysis of a typical packet-switched NoC for real-time mixed-critical systems. It comprises an FMEA, and a classification of error effects regarding duration and ability to compromise task isolation. The analysis results give a comprehensive insight into the behavior of the NoC under the effect of faults. The results include the identification of a system failure mode in which packets are being blocked by unreleased resources inside the NoC switch, which has not been described in literature so far.

II. Related Work

The authors in [4] give an overview about outstanding research problems in NoC design listing fault-tolerance and reliability as key problems. However, the motivations given in [4] and many related works are the increased transient faults due to transistor scaling and not certification for safety-critical systems. [5] presents an FMEA method for SoC-level design compliant with IEC 61508 targeting RTL and gate level. The proposed methodology was applied to design memory subsystems for microcontrollers. [6] presents a SoC-level risk assessment using a SystemC TLM model, which seems sufficient for risk assessment, but does not yield sufficient insight into error propagation.

The effects of single bit errors on the function of a network interface (NI) are analyzed in [7]. For this, the paper introduces a network vulnerability factor, which takes into account that bit errors that do not contribute to the future program state are masked. From this, the reliability of a NI or NoC architecture can be computed. In contrast to this work, we do not focus on reliability but on fault detection coverage. As required for higher safety levels, our approach ensures that all potential errors are covered and detected. In this analysis, we reflect the fact of error masking by ignoring masked errors further in the analysis.

For wide-area networks, there is a long history of studies on reliability including FMEA, see e.g. [8]. However, these studies are not directly applicable to on-chip networks as they usually assume that packet drops are only a degradation and not a failure. In wide-area networks, FMEA studies usually aim at increasing the availability of the network. In contrast,

The authors wish to thank Moritz Neukirchner for the valuable discution and contribution to the paper.

This work has been partially funded by the FP7 project Certification of Real Time Applications Designed for Mixed Criticality (CERTAINTY), agreement no. 288175, and by German Research Foundation (DFG) as part of the priority program "Dependable Embedded Systems" (SPP 1500 – spp1500.itec.kit.edu).

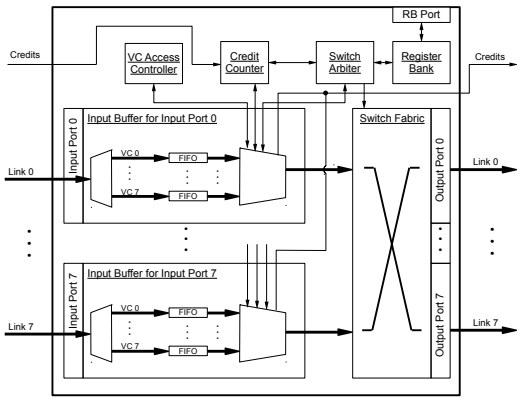

Fig. 1. Switch architecture

the FMEA presented in this paper aims to be a base for a fault-tolerant NoC with minimal overhead, capable of maintaining task isolation even during the occurence of single event upsets as required for the use in safety-critical systems.

Much research has been published regarding mechanisms to detect or correct soft errors in the NoC, using schemes such as probabilistic flooding, directed flooding, random walk and retransmission [9]–[12]. Similarly, mechanisms that correct hard errors have been proposed. In [13], the authors present a fault-tolerance scheme which is deadlock-free and uses adaptive replication to reduce the power consumption overhead. [14] presents a fault-adaptive cost-based deflection routing mechanism. It uses detailed fault status of NoC crossbar connections obtained through distributed online diagnosis. [15] presents a fault-tolerant NoC scheme using bidirectional channel, instead of detouring packets as in traditional schemes.

While faults in the NoC can be covered by integrating fault-tolerance mechanisms directly into the NoC itself, they have also been treated at a higher level. [16] presents an adaptive checkpointing scheme which periodically saves each applications state so that a fault-free state can be loaded in case of an error.

Most existing publications regarding NoC fault tolerance have in common that they present or evaluate certain fault-tolerance mechanisms but lack a systematic assessment of all potential NoC failure modes and hence can not prove that all single errors are covered as required by safety standards. Moreover, the reliability of the task isolation provided by the NoC has not been addressed yet, which is essential for mixed-critical real-time systems.

III. System Description

In order to perform a thorough analysis, it is necessary to provide a description of the system and its intended function. In this paper, we consider a typical packet switched NoC [1]. We focus on the common 2D mesh topology in which each switch is connected to up to four neighbor switches and is also connected to up to four IP blocks through network interfaces (NIs). The NoC's function is to:

- transfer any message to the intended recipient,
- maintain packet integrity,
- maintain isolation (avoid corruption of other parallel data transmissions), and
- ensure that the quality-of-service guarantees are met.

For the scope of this paper, we draw the system boundary between the NoC switches and the network interfaces (NIs) that are used to connect the IP blocks (e.g. processors, memory). We exclude the network interface from the FMEA for modularity, as there may be different network interfaces depending on the type of IP. Analyzing the NoC as a subsystem without the NIs (instead of including it in the analysis of the complete multi-core) allows reuse of certification for different NoC instantiations and hence saves certification costs. Regarding the NIs, we only assume that packets are injected fault-free.

Figure 1 shows a block diagram of the switch architecture. Data packets coming over the links are received in one of the input buffers, which represent distinct virtual channels (VCs). Inside the input buffers, the packets are stored in FIFOs whose space is managed on the granularity of Flow Control Units (flits) by a credit counter which prevents over- and underflows. Packets are forwarded as soon as the first flit is available, and multiple flits of a single packet may span multiple switches (wormhole switching).

The packets are composed by one Head Flit (HF), zero or more Body Flits (BF), and one Tail Flit (TF). Packets comprised of only a HF are also possible and are called a Single Flits (SF). A packet's HF includes routing information, which is determined at the source following e.g. an XY routing scheme. The route is encoded as a series of "runs" with each run containing a direction (north, east, west, south) and the number of hops that are traversed in this direction. At each hop, the route field is updated so that the current output port can always be found at the same bit location for fast routing decisions.

Upon the arrival of a new packet in a switch, the routing information in the HF is read and the routing port is identified. In the sequence, a Virtual Channel Access Controller reserves access to the VC in the next downstream switch. The request may be refused if a packet from another input has already reserved the VC. When granted, the reservation is maintained until the last flit of the packet (TF) has been transferred. The switch arbiter grants access to the individual VCs requesting access to an output port via the switch fabric. The switch can forward two different traffic classes: Best-Effort and Guaranteed-Service which is bandwidth. The switch arbiter manages Quality-of-Service (QoS) guarantees [17]. It can be configured via a register bank, which stores parameters such as priority for each VC, and is accessed by the network interface via a dedicated port. To avoid input buffer overflow, neighboring switches exchange credit points to indicate free buffer space. They are registered in each switch's Credit Counter (credit based flow control).

Table I lists all signals that interconnect the components in Figure 1. The signals between switches are listed in Table II.

IV. Methodology

The FMEA is performed hierarchically [3]. For each component type, each component instance is analyzed. Each possible failure mode of the component instance is then examined for every possible system state, evaluating local and global effects of the failure mode. Local effects concern how the functionalities of individual sub-components or the local switch are

TABLE I
LIST OF SWITCH SIGNALS CONSIDERED IN THE FMEA

Module	Signal Name	Description
Input Buffer	port_request	Request access to output port
	port_request_BE	Request type (best-effort or guaranteed-service)
	port_request_GT	
	port_accept	Accept grant from switch arbiter
	flit_type_request	Request or release a VC to VC Access Controller
	vc_status_update	
	data_out	Forward data to switch fabric
	data_out_valid	
VC Acc. Ctrl	vc_access	Grant access to VC
Credit Counter	credit_available	Indicate availability of credit
	back_suction	Indicate low buffer occupancy
Switch Arbiter	input_port_valid	Control switch fabric
	input_port_select	
	port_grant	Grant port request to input buffer
Switch Fabric	output_data	Send data to downstream switch
	output_data_valid	
Register Bank	slv_data_out	Read data to register bank port
	mst_data_out	
	priority	Priority of each VC
	ds_thresh	Configuration of QoS
	int_thresh	

TABLE II
LIST OF LINK SIGNALS CONSIDERED IN THE FMEA

Signal Name		Description
output_credit		Indicates new available credit (from Input Buffer)
output_valid		Indicates new flit available (from Switch Fabric)
output_data	:VC	Indicates the virtual channel (all flits)
	:FT	Indicates the flit type (all flits)
	:Route	Route of the packet (HF/SF)
	:Supervisor	Indicates whether sender is the supervisor (HF/SF)
	:Payload	Payload of the flit (all flits)

affected. Global effects concern how the functionality of the NoC as a system is affected, i.e. the error propagation.

We benefit from the fact that every input and output of a switch behaves identically to reduce the analysis effort by constructing a minimal network configuration that shows all effects of possible path segments in larger networks. Figure 2 shows the network and the test packet route considered.

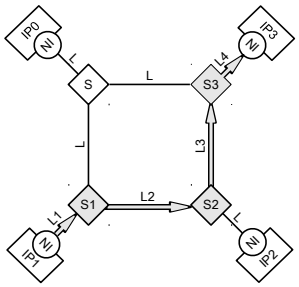

Fig. 2. Minimal network configuration: links (L), four switches (S) and the corresponding network interfaces (NI) connecting to the IP blocks (IP).

The analysis is performed on the block level, therefore errors are assumed on all connections between blocks and between adjacent switches. Errors are evaluated individually for each logical signal, such as the actual flit data or the synchronization between modules (e.g. valid signals).

An FMEA is a manual process with an extensive textual description of possible errors and failures as outcome. Due to space limitations, the full FMEA can be found in [18].

V. RESULTS

The FMEA has revealed 107 unique errors within 21 signal groups in the switch and 54 errors in 7 signal groups between switches.

A. Local effects

For each error, one or more of the following local consequences may occur:

- **The error is masked:** subsequent block ignores it. It happens when the faulty signal is only evaluated when other signals have a specific value.
- **A flit is corrupted**, i.e. the flit content changes. Depending on where this error occurs inside the flit, there may be different global consequences.
- **A flit is lost.** It may happen if there is an error in the communication between blocks, e.g. the input module sends a flit to the switch fabric but it doesn't get through, or the corruption of a flit's VC ID and the flit is hence stored in an incorrect VC buffer.
- **A flit is sent to the wrong output port.** It may be caused by an error in the control information, e.g. input port select, or an error in the communication between blocks, e.g. switch arbiter and switch fabric.
- **A flit transmission is delayed.** It may be caused by an error in the control information, e.g. VC priority, or an error in the communication between blocks, e.g. register bank and switch arbiter. Arbitration errors usually do not lead to a complete loss of functionality as the switch re-arbitrates every cycle.
- **A VC buffer is blocked.** It may happen if the switch makes an incorrect decision regarding VC reservation. An allocation error is usually permanent, as a VC buffer (e.g. one that had already been allocated) is only released if a TF is processed, which will never happen as the reservation error prevents progress.

B. Global effects

The global effects (i.e. system failures) depend on the type and location of the local errors. One or more of the following global effects may be the consequence of an error:

1) **Quality-of-Service violation.** The VC QoS guarantee is violated. This may happen temporarily, e.g. due to an incorrect switch decision or a signal glitch, or permanently, e.g. due to corruption of the VC priority, or due to VC buffer blockage.
2) **Packet loss.** The packet is lost in the network. This may be caused by the loss of HF or TF, route or VC field corruption, an incorrect decision in a switch, or VC buffer blockage. Also, the packet may be delivered to a wrong recipient.
3) **Packet corruption.** The packet arrives at the correct destination but is corrupt. This may happen e.g. due to a bit flip in the payload of a flit or the loss of a BF.
4) **Return route corruption.** The route field gets corrupted without immediate effect, e.g. because it affected only the part of the route that was already traversed, causing the NI to be unable to reconstruct the return route e.g. for an acknowledgment.
5) **VC buffer blockage.** This may be caused by the loss of a HF or TF, or incorrect switch decision caused by corrupt control data, e.g. credit counter and VC access control. This failure often happens together with

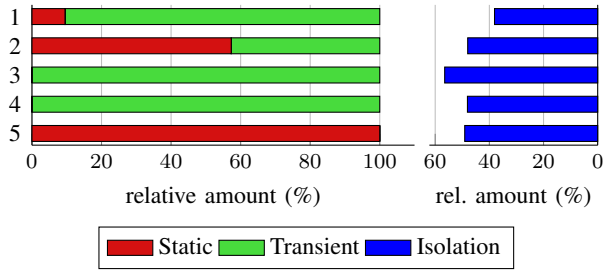

Fig. 3. Global effect: relative amount of errors.

packet loss, although they may happen independently. Due to wormhole switching, this error may propagate to downstream switches. For instance, if a packet's TF is lost, all switches downstream will not receive a TF and hence never release the corresponding VC reservation.

The effects of the corruption of the flit type and virtual channel fields can also be considered as a flit loss. The corruption alters the flit semantics and, in the perspective of its aggregating packet, the flit is lost. However, the flit still exists and it will now affect the transmission of other packets.

C. Effects characterization

Task isolation must be preserved in mixed-critical systems even in the presence of faults. Studying the effects that a faulty transmission belonging to one task has on other tasks in the NoC is crucial in order to prevent them properly. For that reason, the effects were analyzed in terms of isolation violation and also regarding how long the error remains in the NoC.

Figure 3 shows the percentage of cases where a global effect (1–5) presents a given characteristic. We focus initially on the conventional ones, Transient and Static. An effect is Transient when the effect goes with the affected packet or it is Static when the effect remains, affecting other transmissions.

Global effect 1 usually presents transient effect duration, while effects 3 and 4 are always transient. The most adverse cases are related to global effects 2 and 5, where effects remain in the NoC more than 57% and all cases, respectively. This is due to the fact that global effects 2 and 5 are caused by the local effects flit corruption (VC data), flit loss, flit sent to wrong output port, and VC buffer blockage, effectively breaking the state of the switch (which is defined by the packets being transferred and the state of these transfers), since now the state of one or more transfers don't reflect the reality anymore.

Figure 3 also shows the relative amount of cases where a failure mode also violates the isolation property of the system. Notice that this value is orthogonal to the characterization of Transient and Static, i.e. isolation can be violated. An isolation violation happens when a flit deviates from its route and affects other transmissions in the NoC, by either migrating to another VC or changing its route. Such cases may happen due to the local effects flit corruption (VC and route data) and flit sent to wrong output port.

VI. Conclusion

We have performed a detailed failure analysis on a typical NoC architecture for mixed criticality applications. We also describe a methodology used to reduce the analysis effort exploiting various abstractions and symmetries while ensuring that all potential errors were still captured. The analysis comprises an FMEA and error effects classification. It yields deep insight into the failure modes and their impact on the task isolation; and provides a full coverage of potential single faults, as required by safety regulations standards.

The outcome of this work forms a solid starting point for the selection, design and implementation of effective lightweight fault-tolerance solutions with lower performance overhead. The approach can be used to improve NoC resilience even in non-critical applications.

References

[1] B. Motruk, J. Diemer, R. Buchty, R. Ernst, and M. Berekovic, "IDAMC: A Many-Core Platform with Run-Time Monitoring for Mixed-Criticality," in *High-Assurance Systems Engineering (HASE), 2012 IEEE 14th International Symposium on*, 2012, pp. 24 –31.

[2] "IEC 61508: Functional safety of electrical/electronic/programmable electronic safety-related systems, ed.2.0," International Electrotechnical Commission, Tech. Rep., 2010.

[3] "IEC 60812: Analysis techniques for system reliability Procedure for failure mode and effects analysis (FMEA)," International Electrotechnical Commission, Tech. Rep., 2006.

[4] R. Marculescu, U. Ogras, L. Peh, N. Jerger, and Y. Hoskote, "Outstanding Research Problems in NoC Design: System, Microarchitecture, and Circuit Perspectives," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 1, p. 3, 2009.

[5] R. Mariani, G. Boschi, and F. Colucci, "Using an innovative SoC-level FMEA methodology to design in compliance with IEC61508," in *Design, Automation Test in Europe Conference Exhibition, 2007. DATE '07*, april 2007, pp. 1 –6.

[6] Y.-Y. Chen, C.-H. Hsu, and K.-L. Leu, "SoC-level risk assessment using FMEA approach in system design with SystemC," in *Industrial Embedded Systems, 2009. SIES '09. IEEE International Symposium on*, july 2009, pp. 82 –89.

[7] Y. Zou, Y. Xiang, and S. Pasricha, "Characterizing vulnerability of network interfaces in embedded chip multiprocessors," *Embedded Systems Letters, IEEE*, vol. 4, no. 2, pp. 41 –44, june 2012.

[8] J. Shi, S. Wang, and K. Wang, "Challenges and evaluation method in network performability analysis," in *Robotics, Automation and Mechatronics (RAM), 2011 IEEE Conference on*, sept. 2011, pp. 96 –101.

[9] T. Dumitraş, S. Kerner, and R. Mărculescu, "Towards on-chip fault-tolerant communication," in *Proceedings of the 2003 Asia and South Pacific Design Automation Conference*, ser. ASP-DAC '03. New York, NY, USA: ACM, 2003, pp. 225–232.

[10] P. Bogdan, T. Dumitraş, and R. Marculescu, "Stochastic communication: A new paradigm for fault-tolerant networks-on-chip," *VLSI design*, vol. 2007, 2007.

[11] M. Pirretti, G. Link, R. Brooks, N. Vijaykrishnan, M. Kandemir, and M. Irwin, "Fault tolerant algorithms for network-on-chip interconnect," in *VLSI, 2004. Proceedings. IEEE Computer society Annual Symposium on*, feb. 2004, pp. 46 – 51.

[12] J. Kim, D. Park, C. Nicopoulos, N. Vijaykrishnan, and C. Das, "Design and analysis of an NoC architecture from performance, reliability and energy perspective," in *Architecture for Networking and Communications Systems, 2005. ANCS 2005. Symposium on*, 2005, pp. 173–182.

[13] Y. Zou and S. Pasricha, "Narco: Neighbor aware turn model-based fault tolerant routing for nocs," *Embedded Systems Letters, IEEE*, vol. 2, no. 3, pp. 85 –89, sept. 2010.

[14] A. Kohler, G. Schley, and M. Radetzki, "Fault tolerant network on chip switching with graceful performance degradation," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 29, no. 6, pp. 883–896, 2010.

[15] W.-C. Tsai, D.-Y. Zheng, S.-J. Chen, and Y.-H. Hu, "A fault-tolerant noc scheme using bidirectional channel," in *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*, 2011, pp. 918–923.

[16] Y. Zhang and K. Chakrabarty, "Dynamic adaptation for fault tolerance and power management in embedded real-time systems," *ACM Trans. Embed. Comput. Syst.*, vol. 3, no. 2, pp. 336–360, May 2004.

[17] J. Diemer and R. Ernst, "Back Suction: Service Guarantees for Latency-Sensitive On-Chip Networks," in *The 4th ACM/IEEE International Symposium on Networks-on-Chip*, 2010.

[18] E. A. Rambo, L. Ahrendts, and J. Diemer, "FMEA of the IDAMC NoC," Institute of Computer and Network Engineering – TU Braunschweig, Tech. Rep., 2013.