Methods for Fault Tolerance in Networks on Chip

Martin Radetzki, Chaochao Feng, Xueqian Zhao, Axel Jantsch

Networks on Chip constitute the interconnection architecture of future, massively parallel multiprocessors that assemble hundreds to thousands of processing cores on a single chip. Their integration is enabled by ongoing miniaturization of chip manufacturing technologies following Moore's Law. It comes with the downside of the circuit elements' increased susceptibility to failure. Research on fault tolerant Networks on Chip tries to mitigate partial failure and its effect on network performance and reliability by exploiting various forms of redundancy at the suitable network layers. The article at hand reviews the failure mechanisms, fault models, diagnosis techniques, and fault tolerance methods in on-chip networks, surveys and summarizes the research of the last ten years. It is structured along three communication layers: the data link, the network and the transport layers. The most important results are summarized and open research problems and challenges are highlighted to guide future research on this topic.

Categories and Subject Descriptors: A.1 [**INTRODUCTORY AND SURVEY**]: General; C.1.2 [**Multiple Data Stream Architectures (Multiprocessors)**]: Interconnection architectures; C.1.4 [**Parallel Architectures**]: Distributed architectures

General Terms: Design, Reliability

Additional Key Words and Phrases: Network-on-Chip, failure mechanisms, dependability, fault models, diagnosis, fault tolerance, reconfiguration

1. INTRODUCTION

Computer networks and interconnection networks for parallel processing have been subjects of research since the early days of computing. Since the year 2000, the field of on-chip interconnection networks or Networks on Chip (NoC) has emerged due to the advent of multi-core CPUs and the foreseeable trend towards massively integrated many-core architectures. A number of surveys [Ogras et al. 2005; Bjerregaard and Mahadevan 2006] have been published on NoC in general, and textbooks have become available on the topic, e.g. [De Micheli and Benini 2006]. Still, a variety of open research questions exists on the more advanced topics in NoC [Owens et al. 2007; Marculescu et al. 2009]. Among the latest research trends, fault tolerant NoC design has a prominent place in terms of relevance and attention, as can be judged by inevitable technology trends and published papers.

The survey at hand assumes textbook-level knowledge of NoC technology and terminology. It aims at giving a complete overview of all aspects of fault tolerance (including failure mechanisms, fault models, diagnosis, redundancy and reconfiguration techniques) pertinent to NoC. To assemble as much of the necessary context as possible, we first motivate the need for fault tolerance (Section 1.1), clarify terminology

Martin Radetzki, supported by the Deutsche Forschungsgemeinschaft (DFG) under grant RA 1889/4-1, address: Universität Stuttgart, Institut für Technische Informatik, Pfaffenwaldring 5b, 70569 Stuttgart, Germany; email: martin.radetzki@informatik.uni-stuttgart.de. Chaochao Feng, School of Computer, National University of Defense Technology, Changsha 410073, China; email: fengchaochao@nudt.edu.cn. Chaochao Feng, Xueqian Zhao, Axel Jantsch: Royal Institute of Technology, ES / ICT / KTH, Forum 120, 16440 Kista, Sweden; email: {cfeng,xueq,axel}@kth.se.

(Section 1.2) and give a compact review of the physical processes that lead to faults in Section 1.3. Further sections come to the core by investigating methods for fault tolerance along the layered organization of NoCs, focusing on the data link (Section 2), network (Section 3), and transport layers (Section 4) which are key in providing reliable data transmission.

1.1 Motivation and Scope

Networks on Chip (NoC) by far constitute the largest cohesive subsystem in manycore architectures. Compared to bus subsystems, they offer reduced hardware overhead, better scalability, and higher data throughput [Angiolini et al. 2006; Lee et al. 2007]. By adjusting design parameters, NoCs can be flexibly adaptated to application requirements. To date, NoC prototypes connecting 48 [Intel Labs 2010], 64 [Bell et al. 2008] and 80 [Vangal et al. 2007; Hoskote et al. 2007] cores have been built. Within the next 10 years, continued scaling [ITRS 2009] will enable thousands of cores to be integrated with NoCs as the central backbone.

However, size, complexity, and integration density will make future NoCs increasingly vulnerable. The handling of enormous inter-core data rates requires high circuit activity involving long, high-capacity wires, and results in significant power dissipation. Heat accelerates circuit aging processes that miniaturized circuits are prone to. Beyond that, communication in SoC will also suffer from increasingly significant crosstalk and radiation effects. Furthermore, large distributed structures such as NoCs are especially susceptible to variations caused by production or environmental influences. The forecast that on a future 100 billion transistor chip, 20 billion transistors will be malmanufactured and further 10 billion will fail during operation [Furber 2006] may be pessimistic. Substantial increase in failure rate, however, is evident in future CMOS technologies [Borkar 2005; 2007; Constantinescu 2003].

Low-level hardening alone will not be effective in tackling these issues while maintaining economically viable yield. We must learn to build reliable systems from unreliable components without introducing excessive overhead [Borkar 2005]. As NoCs provide inherently redundant communication pathways, they are potentially robust against partial failure. To exploit this property, adequate choices of fault models, error detection, diagnosis procedures, fault tolerance and reconfiguration have to be investigated on all layers of the network. The survey at hand provides an overview of the state-of-the-art in this emerging field of research. Whereas we assume that the reader is familiar with basic NoC terminology, an overview of fundamental fault tolerance terminology is given in the next subsection.

Obviously, fault tolerance is a much larger topic and has been studied for many years in various domains. Other surveys exist e.g. for dependable computing and fault tolerant control systems [Avižienis et al. 2004; Zhang and Jiang 2008]. We do not intend to cover fault tolerance in general but focus specifically on Networks on Chip. More precisely, the survey is directed at 2-D, electrical NoCs ignoring 3-D and TSV (Through Silicon Via) specific issues as well as optical networks. There recently emerged topics are currently subjects of intensive research and deserve a separate careful review and assessment.

Finally, it is worth emphasizing that this survey concentrates on the fault tolerance of the interconnection infrastructure alone. Hence, work on fault tolerance or redundancy of processor cores, the use of the NoC for testing cores or vice versa, or application specific aspects such as fault tolerant mapping and migration of application tasks and their communication are considered out of scope. A review on the related topic of error detection and recovery in multicore processors has recently been presented by Gizopoulos et al. [2011].

1.2 Terminology

In fault tolerance research, a cause-effect chain that extends over multiple layers has to be considered. At the lowest layer, physical *failure mechanisms* cause elementary devices such as transistors or wires to fail. At the next higher layer, such failure is modeled as a *fault*. The set of all modeled faults constitutes the *fault model*. It encompasses the effect of faults, the location of faults, and potentially further parameters such as

faults' duration and probability. The simplest and most wide-spread fault model of digital circuits represents outputs of logic gates (or the connected wires) as stuck at certain values (stuck-at-0, stuck-at-1).

1.2.1 *Hierarchical fault modeling.* A fault may, but need not always, activate an *error*. An error denotes the deviation of information from the fault-free case. For example, a stuck-at-0 fault activates an error only if the correct logic value were 1. Otherwise, the fault does not become manifest. An error may or may not propagate to a primary output of the module in which it occurs. In the first case, the error is said to cause a *failure* of the module. In the latter case, the error is said to be *masked*, not causing module failure but possibly remaining *latent*.

As we proceed further to higher layers, lower-level modules are instantiated as components and connected. Their failure is again modeled as faults; however, the fault model becomes more abstract, appropriate for the layer, but more remote from the original physical failure mechanism. Abstract fault models are necessary because it is intrinsically hard to work directly with models of the physical phenomena due to the enormous amount of details and the complexity of dependencies and interaction. Whereas low level fault models are often *structural* (related to failing components), at higher levels *functional* fault models that describe failure of subsystem functions become more prevalent. Again, faults may cause errors, and errors may lead to failure. This chain is repeated upwards the system's design hierarchy until the top layer is reached, where system failure is observed.

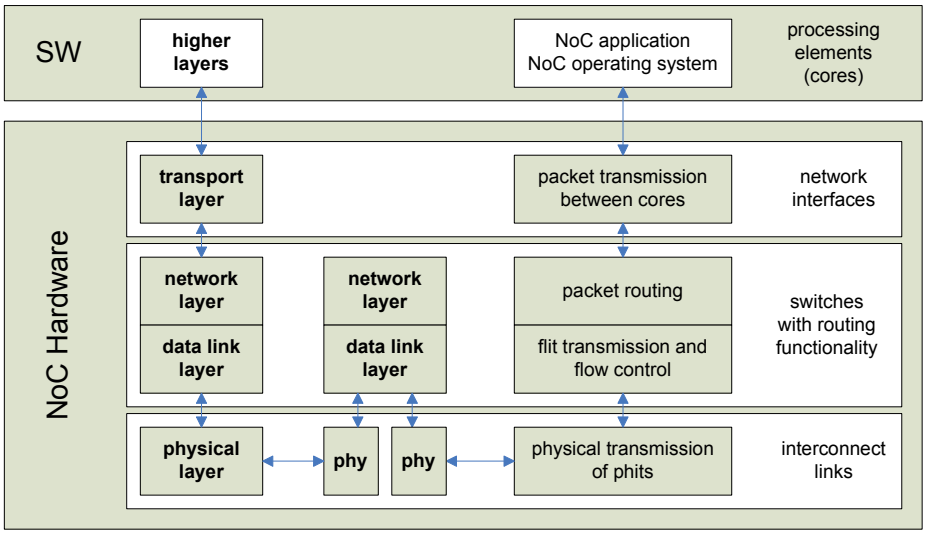

Fig. 1. Network on Chip layers and modules

1.2.2 *Hierarchy of NoC layers.* To assess fault tolerant Networks on Chip, we adopt the layer organization according to the OSI reference model of communication protocols (Figure 1). In particular, the *data link layer*, *network layer*, and *transport layer* are considered in this survey. Higher layers, relating to the application software and middleware, are not considered here as they assume the transport layer to provide reliable data transmission. The physical layer corresponds to the transmission of information units (*phits*), which is not explicitly considered in this survey because it comprises a rich amount of research work and is not specific to NoCs. At the data link layer, reliable data transmission and flow control of so-called *flits* is

implemented. The network layer is responsible for switching and routing. Finally, the NoC transport layer provides end-to-end transmission of data *packets*. A packet consists of one or multiple flits transmitted one after another, and a flit may be composed of multiple phits.

1.2.3 *Fault tolerance through redundancy.* On each layer, specific fault tolerance mechanisms are relevant. Therefore we organize the main chapters of this article according to the NoC layer structure. Within each layer, redundancy is the fundamental mechanism to achieve fault tolerance. Redundancy may be achieved by redundant components to cope with failing ones (*spatial redundancy* or *modular redundancy*), by re-execution of a failed computation or data transmission with the same component (*temporal redundancy*), and by adding information for error correction to the data being processed (*information redundancy*). The kind of redundancy employed will be used to further distinguish fault tolerance techniques.

1.2.4 *Fault classes.* Redundancy must be chosen appropriately to tolerate different *fault classes*. A widely used grouping is the one into transient, intermittent, and permanent faults [Constantinescu 2003].

Transient faults appear more or less randomly for one or several cycles. Transient logic faults due to neutron and alpha particles are truly random; their rate is constant during the lifetime of a chip.

Intermittent faults are easy to confuse with transient faults. Constantinescu [2003] proposes three criteria to distinguish intermittent from transient faults: (1) "Intermittent faults occur repeatedly at the same location"; (2) "Errors induced by intermittents tend to occur in bursts"; (3) "Replacement of the offending circuit removes the intermittent fault". Intermittent delay faults on wires are due to electromagnetic interference, either crosstalk or self-coupling, when they depend on the operating conditions such as the temperature. They also depend on manufacturing variations and imperfections. Thus, their probability is higher on specific links. All the aging processes discussed below can first lead to intermittent delay or logic faults and later on progress into permanent faults.

Permanent faults can either be logic faults, where transistors or wires are permanently open or shortcut, or delay faults, where transistors and wires are too slow causing setup and hold timing violations that eventually generate an incorrect logic value.

This distinction focuses on the condition of individual wires and transistors. If a fault leads to a functional error only for specific data patterns, it is still considered a permanent fault and we say it is *masked* under certain conditions. For instance cross talk may lead to a delay fault only under a worst case transition pattern of three or more neighboring wires, e.g. one wire switches from 0 to 1 while both its neighbors switch from 1 to 0. We call this a permanent delay fault which is masked for some input patterns. Similarly, a broken transistor in a NAND gate leads to a wrong output value only for some input patterns and the error is masked for others.

In general, spatial redundancy is best suited for handling permanent faults, temporal redundancy can well cope with transient and intermittent faults, and information redundancy helps with all fault classes. Since different kinds of faults may occur in an NoC, a combination of redundancy techniques may be required to address them all. This leads to the field of *fault diagnosis*. After the occurence of an error has been detected, fault diagnosis determines—to the degree possible—the location of the underlying fault and classifies the fault as transient, intermittent, or permanent. Discrimination of fault classes, e.g. [Bondavalli et al. 2000], is essential to select appropriate countermeasures.

1.2.5 *When to deal with faults.* Fault diagnosis and fault tolerance may be employed at different stages in the life of a system. After production, integrated circuits are tested *offline*, with external test equipment. Test responses may serve for fault diagnosis and subsequent circuit repair actions. Testing and diagnosis may also be performed with dedicated logic integrated into the chip itself. This leads to *built-in self-test* and *self-diagnosis* (BIST, BISD) that are performed at power-up, and that may trigger self-reconfiguration actions

to achieve fault tolerance in the field. Similar actions can be performed *online*, during circuit operation. In this case, we distinguish between methods that preempt regular operation and methods that are performed concurrently, with minimal disturbance of normal functionality. Online reconfiguration is a challenging topic relevant also for other areas like performance and functional adaptation. A general framework, that could also be utilized for fault tolerance provision, is presented by Strano et al. [2012]. Since error detection and fault diagnosis are essential in controlling fault tolerance, a separate subsection is devoted to respective techniques on the relevant network layers.

A categorization of in-the-field failures from a manufacturers point of view is given by Sanyo Semiconductors [2011]: (1) An *early failure* originates from manufacturing imperfections and rate decreases over time. Many early failures can be detected during burn-in tests before the chip is put in operation. (2) *Random failures* have a constant failure rate over time and are due to radiation and manufacturing defects. (3) *Wear-out failures* increase over time due to aging processes.

1.3 Physical Failure Mechanisms

MOSFET (metal oxide semiconductor field effect transistor) based circuits can experience malfunction due to a variety of reasons rooted in the physics of devices and materials. Broadly, they can be grouped into four categories: Radiation, electromagnetic interference, electrostatic discharge, and aging. All four categories are affected by process variations and by dynamic variations of temperature and power consumption.

1.3.1 *Radiation.* The main types of radiation causing soft errors are terrestrial cosmic neutrons and alpha particles that originate from radioactive impurities in the devices and packaging materials [Borkar 2005; Constantinescu 2003; Dodd and Massengill 2003]. These particles may cause the flipping of bits in SRAM and DRAM memory cells, which is called *Single Event Upset (SEU)*. When a particle causes a wire or a logic gate to assume an incorrect voltage level, it is usually called a *Single Event Transient (SET)*. In the literature the usage of these terms is not entirely consistent, but in the following we use SEU when a memory cell is directly affected, and SET when a wire or a logic gate is affected.

The probability of such an event to happen depends on the critical charge necessary to flip a bit. With scaling the critical charge of memory cells shrinks, hence the likelihood of an SEU grows. Little can be done to protect circuits from cosmic neutrons, but purer materials are expected to keep the alpha radiation in check. Thus, it is expected that the total *Soft Error Rate (SER)* due to radiation increases at a similar pace of around 8% per logic state bit for each technology generation as it has been observed in the past [Hazucha et al. 2003]. Since the number of state bits per chip grows with scaling, the effect of radiation on the failure rate per chip is significant as illustrated in Figure 2. In addition to memory cells, logic circuits are also susceptible to radiation caused transients. In fact the effect on logic grows faster with technology scaling and is of the same order of magnitude in recent technologies [Shivakumar et al. 2002; Baumann 2005; Zhang and Shanbhag 2006], although it results in errors less frequently due to masking effects.

1.3.2 *Electromagnetic interference.* A main source of electromagnetic interference is crosstalk between long parallel wires. To counteract the increasing delay and resistance of thinner wires, the height of the wires decreases slowly with technology scaling, if at all. This leads to thin but tall cross sections of wires resulting in growing coupling capacitance and inductance between parallel wires. As a consequence, the signal on one wire influences neighboring wires, resulting in increased signal delay, glitches or damped voltage oscillations [Cuviello et al. 1999]. In addition, self-interference due to the skin effect is of growing concern with higher frequency and smaller wires, because at high frequency an alternate current is pushed to the surface of the wire making the resistance, hence the signal delay, dependent on frequency [Walker 2000].

1.3.3 *Electrostatic discharge.* A strong electric current can cause a destructive breakdown of a device. The current enters either through the I/O pins or is induced by strong electric fields. Three types of electrostatic breakdowns are distinguished: (1) Dielectric oxide breakdown; (2) PN junction breakdown; (3) wiring breakdown. Due to scaling the underlying phenomena have become harder to analyze and control [Yang 2010; Sanyo Semiconductors 2011]. However, protection against these effects is placed at the pins of the ICs and internal circuitry such as the NoC is not affected.

1.3.4 *Aging.* Various physical effects lead to degradation of performance and function over the lifetime of a CMOS device.

Electromigration is the most important mechanism that first leads to increasing delay and then to a malfunction of wires [Keane and Kim 2011; Constantinescu 2003]. It is a current induced transport of metal atoms. Since it thins-out the wire at the point where the wire is already thinnest due to higher current density, initial variation of the wire thickness is aggravated over time. It first causes increased delay and eventually creates an open or short between previously unconnected nets.

NBTI: There are three main effects that wear out the CMOS transistors. *Negative Bias Temperature Instability (NBTI)* increases the threshold voltage of the PMOS transistor over time, because charge migrates into the insulating gate oxide. It is very sensitive to the temperature (Figure 3) but is partially reversed when the stress is removed. That means that a higher frequency leads to slower wear out [Zhang and Orshansky 2008; Keane and Kim 2011; Wittmann et al. 2005]. The corresponding effect on the NMOS transistor called *Positive Bias Temperature Instability (PBTI)* is less important in practice.

Hot Carrier Injection (HCI): When fast carriers, which are electrons or holes, penetrate the insulating silicon oxide layer below the gate, the transistor switching characteristics gradually change. In particular the threshold voltage increases which leads to slower reaction times and degrading performance [Takeda and Yang 1995; Keane and Kim 2011].

Time Dependent Dielectric Breakdown (TDDB): Given an operating voltage of 1V and a gate oxide thickness of 1.2nm, the electric field in a CMOS transistor across the gate is about $\frac{1V}{1.2nm} = 8.33\frac{MV}{cm}$. Strong electric fields ($\geq 10\frac{MV}{cm}$) induce dielectric breakdowns due to crystal defects and heavy metal contamination in the dielectric material. But even at lower electric fields, trapped charges accumulate in the oxide and eventually lead to a conducting path causing a failure of the transistor. Since the operating voltage has scaled less than ideal in recent technology generations, this effect has become more severe. It can be countered by reducing the contamination and crystal defects [Constantinescu 2003; Keane and Kim 2011; Miranda and Sune 2004; Wu et al. 2001].

Plenty of recent work has focused on the problems of wear-out and aging phenomena. For instance Agarwal et al. [2007], Keane and Kim [2011], and Keane et al. [2007] propose ways to measure wear-out related degradation to be able to adapt circuits to new operation conditions, such as lower frequency. Mintarno et al. [2011] propose a self-tuning scheme that adapts supply voltage, clock frequency, and external cooling. These parameters are modified during lifetime of a system based on models for aging.

1.3.5 *Process variability.* Variability of material impurities, doping concentrations, and the size and geometries of the structures caused by inaccuracies of the photo-lithographic and etching processes affect the occurrence and severity of all the above mentioned phenomena. For instance, whether a neutron flips a DRAM cell depends on how much charge is collected in that cell, which in turn varies across the chip due to process variability. Similarly, electromagnetic interference and electromigration vary with the size and detailed geometry of the wires. The same holds for electrostatic discharge and all aging related processes. Impurities in the gate oxide facilitates faster build-up of trapped charges, which in turn leads to more localized electric fields that accelerate the dielectric break down. Thus, minimizing process variability

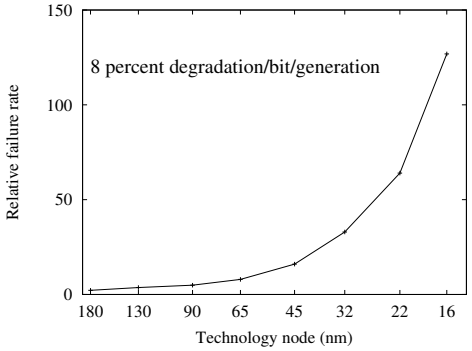

Fig. 2. Soft-error failure-in-time of a chip (logic and memory). [Borkar 2005].

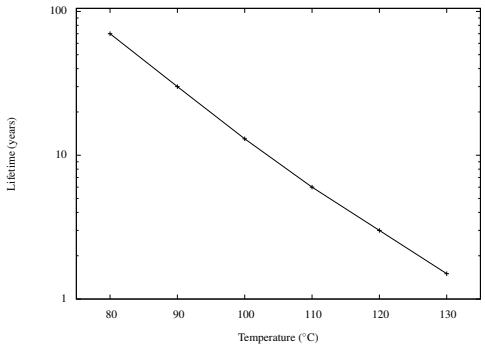

Fig. 3. Lifetime of an inverter chain decreases by a factor of 2.2 for every 10°C increase in operating temperature due to Negative Bias Temperature Instability (NBTI) [Zhang and Orshansky 2008].

plays a key role in establishing a well defined level of quality and reliability. Unfortunately, in the course of scaling the process variability in relative terms has increased to the point where the worst case of one technology node is hardly better than the worst case of the previous node, although the average case is improved significantly [Kuhn et al. 2008; Borkar 2005; Saha 2010].

1.3.6 *Dynamic temperature variation.* High temperatures and temperature variations during operation play an important role as well. As illustrated in Figure 3, NBTI is very temperature sensitive. But all other processes in the aging category are temperature dependent as well [Borkar 2005; Zhang and Orshansky 2008]. In addition, temperature has a direct impact on power consumption and performance. Sub-threshold leakage current rises exponentially with the temperature [Zhang et al. 2003]. Different parts of the chip (SRAM, embedded DRAM, interconnect, CPU core, etc.) exhibit different levels of activity and therefore give rise to different levels of power consumption and heat generation. This leads to strains in the power supply network and variations of performance and wear-out. Since these temperature variations also lead to different pace of wear-out in different parts of the chip, these faults are not distributed uniformly but appear concentrated in clusters.

Due to the combined effect of both, static process variations and dynamic temperature variations we see on-chip variations of performance of up to 30% and of total power consumption of 50% or more [Borkar 2005].

Table I shows an overview of physical causes and their effects. Survey articles by Constantinescu [2003], Borkar [2005], and Keane and Kim [2011] provide more detail. Sanyo Semiconductor Co. maintains an online website that explains the physical phenomena behind fault occurrences, gives basic formulas for mean time to failure (MTTF), and discusses counter acting measures [Sanyo Semiconductors 2011, chapter 4]. A review and assessment of the impact of scaling on the expected physical phenomena in process technologies below 65nm is given by McPherson [2006], while the consequences of increased variability are analyzed by Borkar [2005].

Next we move to the core of the survey by reviewing the techniques developed for detecting and tolerating faults in Networks on Chip.

Physical cause		Accelerated by process variation	Fault class	Burstiness
Radiation	Neutrons	No	Transient logic	Yes
	α-particles	No	Transient logic	Yes
Electromagnetic interference	cross coupling of parallel wires	No	Intermittent delay	No
	self coupling, Skin effect leading to higher resistance	No	Intermittent delay	No
Aging	Electromigration	Yes	intermittent $\rightarrow$ permanent delay and logic fault	No
	Bias Temperature Instability (BTI)	Yes	intermittent $\rightarrow$ permanent delay fault	No
	Hot carrier injection	Yes	intermittent $\rightarrow$ permanent delay faults	No
	Oxide breakdown/Time Dependent Dielectric Breakdown (TDDB)	Yes	intermittent $\rightarrow$ permanent logic fault	No
Power density variation and heat flux	leakage power variation due to temperature differences	Yes	Intermittent $\rightarrow$ permanent delay and logic faults	Yes
	performance variation due to temperature differences;	Yes	intermittent delay failures	Yes
	variations in wear-out effects due to temperature differences	Yes	intermittent and permanent, delays, opens and shorts	Maybe

Table I. Overview of fault causes and effects.

2. DATA LINK LAYER (FLITS OVER LINKS, FLOW CONTROL, ERROR PROTECTION)

According to the OSI model (Figure 1), the data link layer provides the functionality to transfer data between switches with error control. This section covers three aspects including the fault models, fault detection, and redundancy techniques. The data link layer of NoC works tightly with the circuit implementation, thus we cover some physical level techniques that are applied in NoC design. Nevertheless, further discussions on the physical layer models and hardening techniques are not the focus of the paper. We firstly explore the typical fault models (Section 2.1). Then error detection techniques are discussed (Section 2.2) followed by redundancy techniques (Sections 2.3 - 2.5).

2.1 Fault models

The fault models at the data link layer can be classified into two categories: models for the switch logic and models for the wires in the link. For each, transient faults and permanent faults are considered respectively. As for the intermittent fault, only the crosstalk caused delay fault and glitches are concerned, which are generated essentially from electromagnetic interference under specific transition patterns. The other forms of intermittent faults, e.g. induced by aging effects, are treated as permanent faults for simplicity because the countermeasures for permanent faults are also effective to them in fault tolerant NoC designs.

2.1.1 *Fault models for the switch logic.* Transient faults in the switch logic mainly come from the radiation effects. For the storage elements such as the flip-flops, the effects are modeled as state upset [Frantz et al. 2006b]. However, the transient effects on the combinational logic signals manifest as voltage distur-

bances in the form of glitches. This transient disturbance can be characterized by a pulse with the duration subject to some distribution. Yaghini et al. [2011] assume the pulse duration to be exponentially distributed and capture the delay by a delay module. The occurrence of the transient faults is thought to be uniformly distributed. For permanent faults in logic, the stuck-at fault model is widely used, especially in fault diagnosis. Stuck-at is a logic level abstraction that assumes that nodes are permanently stuck at either logic 0 or logic 1.

2.1.2 *Fault models for the link wires.* The transient disturbance on the link wire can be treated as voltage noise [Lehtonen et al. 2007b; Fu and Ampadu 2009] to deduce the error probability of the channel, which is first proposed by Hegde and Shanbhag [2000]. Because the disturbance may come from various sources, the value of the voltage noise is considered to be subject to a normal distribution with the variance of σ_N^2. Then the error probability ε in the link is modeled in terms of the link voltage swing V_{swing} and the variance of the noise voltage. The probability of the link error is characterized by a Gaussian distribution as

$$\varepsilon = \int_{\frac{V_{swing}}{2\sigma_N}}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy. \tag{1}$$

Crosstalk is considered as one of the main fault sources for link wires, which may lead to glitches and intermittent delay faults. By analyzing the transition patterns, Zimmer and Jantsch [2003] bring forward mathematical models to predict the error probability, considering the fault duration and disturbance on multiple wires. Considering more general transition patterns, Patooghy et al. [2010] propose an analytical model to predict the delay caused by crosstalk in NoC links. It considers the occurrence distributions of various transition patterns in a communication link (a bundle of wires), and predicts the delay based on calculating the numbers of theses transition patterns.

2.2 Error detection

2.2.1 *In-operational detection.* To detect transient faults, the time redundancy technique is an effective solution. The basic principle is to sample the target signal several times (usually twice) with some interval and then compare these samples to identify the faults, as shown in Figure 4. The delayed sample can be obtained by either using delayed clock or utilizing the propagation delay of the logic. Various implementations and cost reduction of the delay sampling circuit have been studied in [Anghel and Nicolaidis 2000].

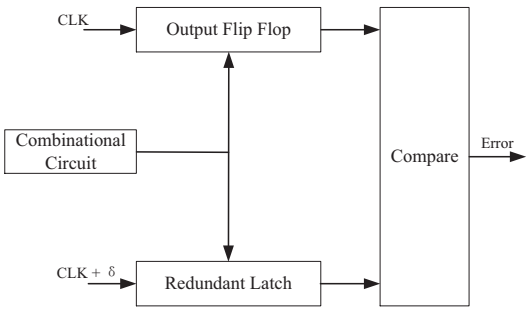

Fig. 4. Error detection scheme based on multi-sampling [Ravindran 2009].

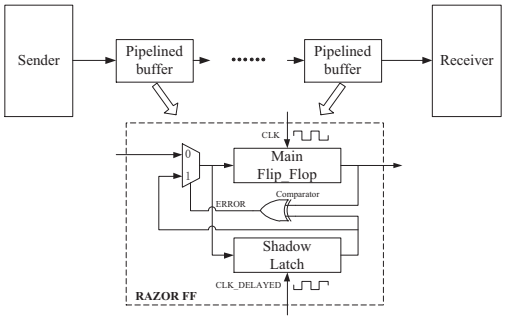

Fig. 5. Pipelined link with error control by Razor flip-flop.

This principle is also applied in Razor flip-flops [Ernst et al. 2003] that not only detect the error but also restore the value when the error is detected. A 2:1 multiplexer is controlled by the result of the comparator

to select from the original sample and the delayed sample, as shown in Figure 5 [Tamhankar et al. 2005]. When the link between switches is very long, it is usually implemented as a pipeline to improve the average throughput. For each stage the buffer can be implemented by Razor flip-flops to detect the transient fault and the timing violation induced fault on the link. It should be noted that the delay is carefully selected to guarantee the fault to be properly latched. Typical durations of SETs are characterized by Gadlage et al. [2010] who show that pulse durations are expected to be below 100ps for 65nm technology.

Coding provides another in-operational detection, known as information redundancy technique. The original data is transferred into some code structures with extra information to check the correctness. Generally the switch contains encoder and decoder at the input and the output ports. Coding schemes such as error correction codes (ECC) and parity bits have been widely used to catch faults in NoC links and routing components. This topic will be elaborated in Section 2.4.

2.2.2 *Built-in self test (BIST).* BIST is extensively used to detect the permanent faults. Cota et al. [2008] give an review of BIST techniques for NoC diagnosis. BIST based error detection generally comprises the test data generator (TDG) and the test response analyzer (TRA) in the switch. In order to detect the faults on the control wires as well as the data path, TRA and TDG can also be placed in the network interface [Cota et al. 2008]. If TDG and TRA are placed in both of the switch and the network interface, BIST can identify faults either in switches or in links, as developed by Kakoee et al. [2011a]. Moreover, they implement a fault diagnosis module (FDM) in each switch to locate the fault. The faulty part can be disabled to isolate the detected fault, and use the healthy part to continue working [Lin et al. 2009]. Another in-line test method is proposed by Lehtonen et al. [2010] to test the integrity of all wires on the link. A test pattern generator (TPG) implemented in the switch can be invoked regularly without interrupting the normal data transmission.

In the following subsections, we focus on the redundancy techniques. Some of them are essentially coupled with fault detection, e.g. coding and time delayed sampling. Practical fault tolerant designs usually use hybrid redundancy, which combines several types of redundancy comprehensively. For example, error detection code (information redundancy) detects the error and notify the source node to retransmit (time redundancy), where the signal for notification is protected by triple modular redundancy (TMR, spatial redundancy) to ensure correctness.

2.3 Time redundancy techniques

Time redundancy is achieved by repeating the computation, sampling or retransmission. At the data link layer, three types of time redundancy are widely used, namely multi-sampling and correction, hop-to-hop (H2H) retransmission and split-link transmission.

2.3.1 *Multi-sampling.* Multi-sampling is a typical time redundancy technique to detect and correct the faults, especially effective for transient faults. Thus it is widely used to detect the faults in combinational logic and wires [Nicolaidis 1999; Anghel and Nicolaidis 2000; Elakkumanan et al. 2006; Mitra et al. 2006]. If the data is sampled twice with some time interval, then the fault can be detected because transient faults disappear after some time and do not occur in different samples (see Section 2.2 and Figure 5). Three or more times of sampling can also correct the fault by majority voting, where a transient fault is masked directly. For instance, Frantz et al. [2007] combine multi-sampling with TMR to sample the signal in the link three times. Then the majority value is selected as the correct output. In [Eghbal et al. 2010], the multiplexer in the switch is tripled combined with time redundancy. The output of the main multiplexer is read for three times thus the potential fault can be detected and corrected.

2.3.2 *Hop-to-hop retransmission.* Retransmission, also known as Automatic Repeat Request (ARQ), has been proposed and used for decades to provide error control in communication networks. The ARQ

technique at the data link layer in NoCs is implemented as the hop-to-hop (H2H) retransmission. It is often coupled with error detection and correction technique to implement so called Hybrid ARQ (HARQ). Once an error is detected, then a negative acknowledgement (NACK) or a request is sent to the source node for retransmission. For on-chip interconnects, Fu and Ampadu [2009] study various HARQ schemes in terms of energy efficiency, delay and area cost. Kim et al. [2005] propose five different hybrid retransmission schemes to cope with NoC link errors, and the buffer requirement and latency cost are evaluated. Different from end-to-end (E2E) schemes which check the data only at the destinations, H2H schemes check the data at each switch on the path. More detailed difference of these two retransmission types can be found in articles by Murali et al. [2005] and Park et al. [2006]. We focus on the H2H retransmission here, which explores fault tolerance based on flits.

The basic principle of H2H retransmission is illustrated in Figure 6 [Murali et al. 2005]. The switch contains a decoder to implement error detection and correction coding. A retransmission buffer stores the transmitted flits. If an error is detected but cannot be corrected in the next hop, the control signals will be set to require retransmission and the flit stored in the retransmission buffer will be issued subsequently. To guarantee reliable control of the retransmission, the control signals, e.g. the Negative Acknowledgement (NACK), are protected by triple modular redundancy. Retransmission is able to tackle multi-bit faults, because the clean data is always maintained as long as the faults are detected. Figure 6 just shows the general diagram of H2H retransmission, the practical implementation and cost may vary in different designs.

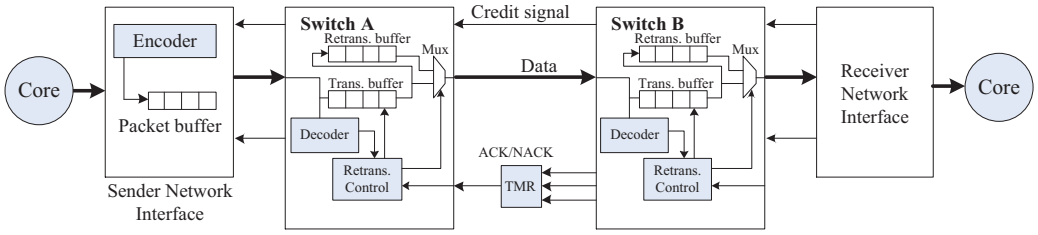

Fig. 6. The basic principle of H2H retransmission. The flits are buffered in a retransmission buffer. Once an error is detected, the NACK signal is sent back from the next hop and is used by the retransmission control logic to start the retransmission process and configure the multiplexer at the output port.

Park et al. [2006] propose a HARQ scheme which combines H2H retransmission with the forward error correction (FEC) at the receiving node to handle link faults. By a novel design of the retransmission buffer, they implement a minimal-overhead flit-based H2H retransmission scheme. The depth of the retransmission buffer is designed to equal the time (the number of clock periods) in which the NACK signal arrives from the next node. Note that the retransmission buffer should be protected as well as the control signals in the link. The error in the retransmission buffer may cause endless retransmission because the original data is erroneous. For a single soft error, it can usually be corrected by the correcting unit at the next switch. Kim et al. [2005] propose to simply duplicate the buffer if the cost is bearable. As for the multi-bit upset in the retransmission buffer, Park et al. [2006] suggest ignoring it to reduce cost because it is highly unlikely to occur.

2.3.3 *Split-link transmission.* The third type of time redundancy technique is the split-link transmission, which utilizes part of the link to continue communication when a permanent fault occurs in the link. It is a sort of graceful degradation based fault tolerance. The flits are split into several parts and transmitted through the healthy wires in a faulty link. At the receiver, the segments of the flits are reassembled to

restore the original data. Lehtonen et al. [2007b] propose the split-link transmission to tackle the intermittent and permanent faults. The flits are split into two parts and the transmission switches to split mode when a permanent fault is detected by coding. A similar work is also developed by Yu and Ampadu [2010]. However, the scheme only splits the wires into two fixed parts, and fails to work if both parts contain faulty wires. One solution is to split the link into more parts in cost of time and hardware, as discussed by Palesi et al. [2010].

A more general solution is developed by Vitkovskiy et al. [2010] to cope with arbitrary number of permanent faults at arbitrary positions. It is a technique that combines retransmission and split-link with the capability of self diagnosis. When the permanent faults are detected, the source node is notified to sent the test vectors to diagnose the positions of faults and the maximum number of adjacent faults. Then rotated flits are retransmitted and reassembled to restore the correct data. This scheme can tolerate an error rate up to 50%. It should be noted that split-link transmission dramatically increases the hardware cost to cope with the permanent faults in the link by up to 35% [Palesi et al. 2010], 33.6% [Vitkovskiy et al. 2010] and even 75% [Lehtonen et al. 2007b]. Complex control logic and additional transmission cycles are required. However, split-link transmission guarantees the continuous working of the chip in presence of permanent faults, thus improves the yield. Moreover, it is also effective in dealing with the process variation induced link degradation [Hernandez et al. 2009].

2.4 Information redundancy techniques

The information redundancy at the data link layer is embodied as using coding schemes to detect and correct the error. We first introduce the coding schemes for the communication between switches, that aim to protect the data from the faults in the link and switch circuits. Then the topics on power efficiency and hardware cost reduction are discussed.

2.4.1 *Code schemes for NoCs* . We consider three types of targets to be protected by information redundancy at the data link layer, namely the link wires, the storage elements and critical control signals in the switch. The input buffers at the switch ports are the most important storage components because they hold packets. To ensure the correctness of the packet before going to the routing and arbitration logic, the input buffer at each input port can be protected by codes with different protection granularity. Generally the whole packet is encoded. However, the header information of the packet usually needs more careful protection than the payload data. Unequal coding meets this requirement and limits the cost to the same level as single-error correcting (SEC) code [Dutta and Touba 2007]. Control signals can be protected against transient faults by exploiting the inherent information redundancy [Yu et al. 2011; Yu et al. 2012]. The valid patterns are recognized through the analysis of the combination of control, request and response signals. Then the invalid combinations are flagged as errors.

The coding schemes to protect the data on the link are implemented by adding an encoder at the output of the port and the received data is decoded at the input to detect and correct the errors. The most popular codes are SEC codes such as the Hamming code. A set of SEC codes can be combined with interleaving techniques to correct adjacent multi-wire errors [Yu and Ampadu 2008; 2011]. Most of the time, both the link and the switch buffers are protected to achieve better fault tolerance [Frantz et al. 2006a; Dutta and Touba 2007; Yu et al. 2010], which can be easily implemented because the same coding technique is shared.

Forward error correction (FEC) can be coupled with the crosstalk avoidance code (CAC) to achieve so called joint fault tolerance. CAC encodes the wires of NoC to forbid some transition patterns to reduce crosstalk and energy dissipation. The combination of FEC and CAC can reduce the transient faults as well as correct the data errors [Pande et al. 2006]. The simplest choice is adding parity bits to the CAC code to provide error detection and correction, such as duplicated-add-parity (DAP) code, modified dual rail (MDR) code and boundary shift code (BSC). Ganguly et al. [2007] combine the Hamming code with DAP or BSC

to achieve crosstalk avoidance and double error correction. Moreover, a triple-error-correction scheme with crosstalk avoiding is developed by Ganguly et al. [2009]. Different from simply combining FEC and CAC, Zhang et al. [2009] propose a selected crosstalk avoidance code (SCAC) which combines CAC and error correction code together through codeword selection from an original CAC codeword set. SCAC is proved to be single error tolerant and crosstalk avoiding and is cost and power efficient.

2.4.2 *Considerations on power efficiency.* Power efficiency is always one of the main concerns when selecting and designing the coding schemes, because information redundancy requires extra circuitry. Different coding schemes require different hardware cost and implementation complexity. This results in various tradeoffs between fault tolerance capability and power efficiency. The major part of power dissipation is due to the interconnection wires [Jantsch et al. 2005; Vitkovski et al. 2008] while logic in the switch consumes a minor part. Various coding schemes result in different circuit implementation complexity, thus add different capacitive loads and switching activities to the link wires which lead to differentiated power dissipation. The results from literature indicate that error detection codes (e.g. ED, PAR, CRC) with retransmission achieve better power efficiency than correcting codes (e.g. SEC, SECDED) [Bertozzi et al. 2002; 2005]. Retransmission applies simpler decoding with less complexity, and it can provide stronger correction capability, e.g. burst error tolerance.

Power efficiency may be improved by two classes of coding techniques: joint coding schemes and strengthening the capability of simple codes. A natural thought is to combine the correcting code with low power code (LPC). However, low power coding schemes seem not to be effective due to the significant delay overhead, which eventually increases the power consumption when normalized to the same performance [Vitkovski et al. 2008]. In contrast, error protection codes have the potential to decrease power consumption by operating the network at a lower voltage, which decreases the signal swing. Sridhara and Shanbhag [2005] propose and evaluate various joint schemes in terms of power, area and reliability. They propose to use a nonlinear source coder (CAC and LPC) to reduce the transition activity before applying linear codes to protect the data. The power can be reduced by up to 33% compared with the scheme with only Hamming code. The other direction is to use a simple code while exploiting techniques to strengthen the error tolerant capability to cope with more errors, as developed in [Yu and Ampadu 2008]. The data is split into blocks to be interleaved, while a SEC code is applied for each block. Then an adaptive error control method is used to select the ECC scheme dynamically. This technique is reported to reduce the power consumption by up to 19%.

Reduction of the power dissipation may also be achieved by comprehensively exploring information redundancy at various layers, as the E2E protection is revealed to be more power efficient than the protection at the data link layer [Jantsch et al. 2005]. Murali et al. [2005] also derive similar result when investigate retransmission based protection. E2E retransmission has smaller power consumption than H2H retransmission under various setups except when the link lengths are very small and large packet-buffering is required by E2E scheme. Yu and Ampadu [2012] improve the power efficiency significantly by applying a dual-layer ECC coding scheme. The end-to-end ECC and hop-to-hop ECC are dynamically adapted by an switching protocol according to various noise conditions.

2.4.3 *Hardware cost improvement.* The hardware cost is another critical issue of information redundancy especially for on-chip design. Several classes of techniques can help reduce the coding induced overhead. The first class is to choose the codes which can make use of the code space as much as possible to reduce the codeword length, and those with high coding rate to reduce the hardware cost. The codewords with less 1s are preferable for linear codes to simplify the coding circuits.

The second is to use simple codes as much as possible to reduce the cost. Codes that detect/correct not more than two random errors (SEC/DED) are usually selected to reduce the cost of coding circuits. In the

work of Zimmer and Jantsch [2003], the data is split into blocks and SEC is applied for each block. The scheme is proved to be effective in handling adjacent multi-wire errors. SEC is competent in correcting more than one error when it is combined with the interleaving technique [Yu and Ampadu 2008]. The cost is improved compared with applying other codes to fulfill the same error correcting requirement. Another thought is developed by Dutta and Touba [2007], who use an unequal coding scheme to protect different parts of the packet. It has similar costs as SEC codes while providing better error detecting and correcting capability, thus the cost is relatively reduced.

The third class is to reuse the coding circuits as much as possible. For linear code and parity check code, the coding circuits usually involve several levels of XOR gates to implement the linear transformation and parity check. By detailed analysis on the characteristics of the generating matrix, sharing and reuse of the XOR-tree is possible when generating the check bits. By circuit sharing, the area cost of SEC is reduced by up to 30% [Yu and Ampadu 2008]. Based on the analysis of the switch circuit and signal response patterns, exploiting the inherent information is another promising technique to dramatically reduce the cost, because least redundancy is introduced [Yu et al. 2011; Yu et al. 2012].

2.5 Spatial redundancy techniques

Spatial redundancy, also known as structural redundancy, covers a wide range of techniques to detect, correct and mask errors. It is based on the replication of the protected target and provides direct and effective fault tolerance in cost of hardware. Spatial redundancy can be classified into three groups: *static redundancy*, *dynamic redundancy* and *hybrid redundancy* [Dubrova 2008]. Static redundancy masks the fault passively while dynamic redundancy firstly detects the faults before acting to correct it. Hybrid redundancy combines static and dynamic methods. The widely used triple modular redundancy is a static method and can mask a single error by more than 200% area cost counting the voter. Concerning the imperfection of the TMR voter, triple duplicated voters can be adopted. More details of modular redundancy techniques are provided by Shooman [2002] and Dubrova [2008].

On the data link layer, spatial redundancy is generally used to secure the function of link wires and to guarantee the correctness of control signals. TMR and spare wires are widely adopted techniques. Important control signals can be protected by TMR, such as the NACK signal in H2H retransmission [Murali et al. 2005; Lehtonen et al. 2007b]. Besides, TMR is also used to harden the logic, e.g. the multiplexer of the switch crossbar [Eghbal et al. 2010], to guarantee correct switching in case of faults. Another thought to enhance reliability is to add hardware redundancy from the very beginning. Kakoee et al. [2011b] double all the physical links between switches to enhance the NoC connectivity. Their work reports up to 10 times higher probability that the NoC is still fully connected in the presence of up to 100 faulty links in an 8×8 NoC.

Spare wires can be prepared to replace the faulty wires once the permanent faults are detected. It is a dynamic redundancy method, and the number of spare wires decides the capability to tolerate faults. To realize the dynamic configuration, the fault detection unit and the link configurability are required. The fault detection can be fulfilled in various ways as introduced in Section 2.2. However, the faulty wires should be specified before reconfiguration. Lehtonen et al. [2007b] and Lehtonen et al. [2010] utilize Hamming code to locate the faulty wire and use a configuration vector to set up the connectivity of the wires by tristate gates. The faulty wires can also be diagnosed with the method proposed in [Vitkovskiy et al. 2010] as introduced in Section 2.3.

Though spatial redundancy brings encouraging capacity for fault tolerance, the overhead is usually large. Tradeoffs can be considered in two directions. On one hand, carefully defining the redundancy granularity helps. Taking TMR as an example, the cost of TMR is heavily affected by the partition granularity and the protection level as analyzed by Constantinides et al. [2006]. They propose a decomposition scheme to

achieve tradeoff between cost and fault tolerance. On the other hand, spatial redundancy can be coupled with the other redundancy techniques to reduce the cost. This strategy is widely adopted, such as incorporating with retransmission, coding and so on.

2.6 Summary

Fault tolerant design at the data link layer is to cope with the faults in the link wires and the logic such as the input/output buffers and the flow control logic. Each of the three types of redundancy (time/information/spatial) can be used to either detect or correct the errors induced by these faults, however at different costs. In practice combinations of techniques have to be used such as [Lehtonen et al. 2007b; Yu and Ampadu 2011] which apply coding, retransmission, spare wires and split-link transmission together to provide a complete protection against transient, intermittent and permanent faults.

Generally, transient and intermittent faults can be countered with error coding and multi-sampling; and permanent faults can be countered with error correcting code, fault detection and rearranging of spare wires and spare resources. For time redundancy techniques the overhead mainly comprises extra latency while for spatial redundancy the hardware costs rises dramatically due to the replicated modules and the reconfiguration logic. As for coding techniques, the power and area overhead varies and heavily depends on the coding scheme and the implementation. The retransmission technique coupled with simple error detection codes is revealed to achieve better power efficiency than error correcting codes under the same reliability requirement of mean time to failure (MTTF).

3. NETWORK LAYER (PACKET SWITCHING AND ROUTING)

The network layer is concerned with packet routing and switching (cf. Figure 1). Corresponding fault models and fault diagnosis are discussed in Section 3.1. Since any packet retransmission would be handled as part of a flow control protocol either end-to-end (transport layer, Section 4) or switch-to-switch (data link layer, Section 2), time redundancy techniques are not pertinent on network layer. The remainder of this section therefore concentrates on information redundancy (Section 3.2) and spatial redundancy (Section 3.3) techniques.

3.1 Faults and their diagnosis

Switches and links are the primary NoC components on network layer. Therefore, the most prominent fault models applied to interconnection networks and also in NoC research relate to these components. A link fault models the failure of a link; in case of bidirectional links a distinction between unidirectional and bidirectional faults can be made. A switch fault captures the failure of a switch, and can alternatively be modeled as a failure of all incident links. However, complete failure of just a few switches can have severe impact on network performance, whereas a physical defect inside a switch is likely to affect only a part of its functionality [Dalirsani et al. 2011]. To make use of the remaining functionality, more fine-grained fault models have been introduced, which will be discussed below. A system-level fault model that captures the functional effect of faults in higher-level categories such as misrouting, flit loss, flit duplication, etc., has been proposed by Aisopos et al. [2011].

Diagnosis of *link and switch faults* has been widely researched already in the field of general interconnection networks. We discuss here NoC-specific novel work only. A method for power-up diagnosis of NoC switches and links in a mesh topology has been devised by Raik et al. [2007]. Directed test packets are injected at the network boundaries to systematically diagnose each individual element. The probe send algorithm [Mediratta and Draper 2007] floods the network with test packets to explore its fault-free subset; any component not contained therein is assumed to be faulty. The drawback of these methods is that any regular network operation is stalled until diagnosis is finished, thus reducing availability and performance.

Port stuck-at faults model the failure of individual switch ports, thereby enabling switch degradation by deactivating the defective port only, instead of the complete switch. Based on this fault model, concurrent online diagnosis of packet misrouting due to defects of the switch control logic has been developed by Alaghi et al. [2007]. Another method [Grecu et al. 2006] equips regular data packets with a parity bit to diagnose switch and link faults concurrent with regular operation.

Crossbar faults [Kohler and Radetzki 2009] model the failure of individual connections from an input port to an output port, thus covering buffers and control logic as well as the crossbar switch. They facilitate switch degradation on a finer granularity, compared to the port fault model. A technique for diagnosing switch-internal crossbar faults with CRC-protected data packets has been presented by Kohler et al. [2010]. Error counters serve to discriminate transient faults from permanent faults because fault-adaptive routing is effective only against permanent (or long-lasting) faults.

3.2 Information redundancy techniques

The use of information redundancy on network layer must be differentiated from the application of error detecting or correcting codes to flits and packets, which belong to the link and transport layers, respectively. The technique specific to the network layer consists in replicating packets and routing them via multiple paths to increase the likelihood that at least one copy arrives correctly at the destination node. This approach is also known under the terms of *probabilistic routing* or *stochastic communication* [Bogdan et al. 2007].

Flooding is the most data intensive of the stochastic communication approaches. In its pure form, each switch copies each received packet to all of its output ports. This means that in d routing steps with k-port switches, k^d copies are produced. This exponential growth may lead to a high probability of delivery for a given packet, but limits the capability of the network to accept other packets. To make stochastic communication applicable in real scenarios, where fault tolerance must be traded off against effective data throughput, refined techniques have been investigated.

Gossip flooding [Dumitras and Marculescu 2003] performs packet replication with a given probability, p. For $p = 1$, it is identical with flooding. Figure 7 shows the number of network cycles after which a packet sent by switch S first appears at a given link. By choosing a smaller p, packet replication and its induced overhead can be limited. Error detection on switch-to-switch basis allows to drop corrupted packets so as to avoid their further replication. By comparing packets or unique packet IDs, switches may identify identical packets and limit their further replication. Packets that make no or little progress towards their destination can be identified with a time-to-live (TTL) field that is counted down, triggering packet deletion when reaching zero. *Directed flooding* [Pirretti et al. 2004] reduces the overhead furter by favoring productive outputs, i.e., outputs that bring the packet closer to its destination, and assigning a lower packet replication probability to non-productive ones. The extreme case is $p^+ = 1$ for productive, and $p^- = 0$ for non-productive outputs. In a mesh, as shown in Figure 8, this leads to a wavefront of packets (dashed lines) traveling in the rectangular region between the sender and destination nodes, S and D. In combination with identification of identical packets, the upper bound on the number of packets is linear with the distance d. However, the approach is not effective if packet source and destination share the same x or y coordinate.

Random walk [Pirretti et al. 2004] limits the overhead further to a constant value, c. c packet copies are created and sent out in a seed phase only, either by their original sender or through a limited initial flooding phase. Other switches perform no further replication but route packets with some degree of randomness, as shown in Figure 9. To achieve effective fault tolerance, routing has to ensure that these copies take disjoint paths, because otherwise a fault in the common part of paths could eliminate all copies. This property can be guaranteed by employing path-disjoint routing algorithms to the original and its copy [Patooghy and Miremadi 2008], where two separate virtual networks are employed to avoid deadlock that could result from interference of the two routing methods. Song et al. [2009] suggest to learn from random walks by filling

routing tables based on acknowledgements of beneficial, randomly round routes.

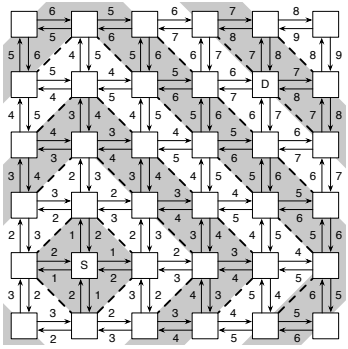

Fig. 7. Flooding / gossip flooding, $p = 1$.

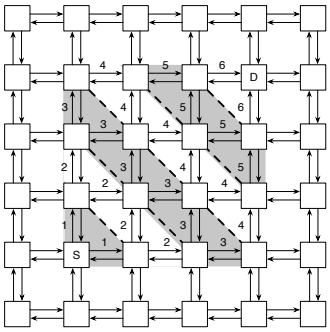

Fig. 8. Directed flooding, $p^+ = 1$, $p^- = 0$.

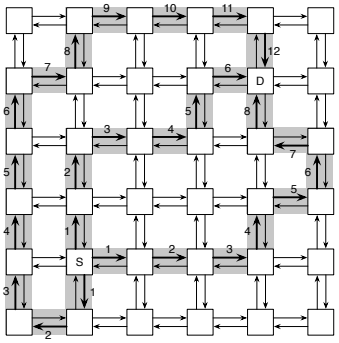

Fig. 9. Random walk, $c = 3$.

The least data overhead achievable by any of these techniques amounts to 100% as one copy of a packet is the least redundancy that can be generated on the granularity of packets. Especially when just a small number of bits are affected by a fault, the use of error correcting codes or retransmission mechanisms, as discussed in Sections 2 and 4, may be more efficient. However, the latter mechanisms fail if permanent faults occur that exceed the correcting capability. For example, the complete failure of a link cannot be tolerated. In such cases, informationally redundant routing on network layer provides better resilience as it exploits the path redundancy of NoCs, i.e., the presence of multiple, at least partly disjoint paths between a packet's source and its sink. This resilience comes at the cost of overdesigning the network so that it can sustain significant redundant traffic without becoming saturated, which would cause unacceptable latency penalties.

3.3 Spatial redundancy techniques

Path redundancy is a feature that comes for free in many NoC topologies, and it can be employed to provide spatial redundancy without replicating hardware modules. For example, the most prominent topology, the k-ary 2-mesh, offers $\frac{(\Delta x+\Delta y)!}{\Delta x!\Delta y!}$ minimal paths between a sender and a receiver Δx and Δy hops apart in x and y dimension, respectively. This redundancy can be utilized to switch to an alternative path dynamically if the standard path fails. Thereby, path redundancy is exploited without the overhead of statically replicating all packets (cf. Section 3.2). A different kind of overhead is incurred, though, since diagnosis of faults and routing reconfiguration to enable alternative paths require some extra logic compared to a plain NoC implementation without consideration of faults.

However, the NoC routing mechanism must support the use of and transition to an alternative path. This implies that a deterministic (non-adaptive) routing mechanism such as popular dimension order routing cannot support fault tolerance through spatial redundancy. Moreover, fault tolerant routing must in general support non-minimal routes. For example, consider $\Delta x = 0$ or $\Delta y = 0$, which leave just a single minimal path in a mesh. Topological change and the use of alternative paths also run the risk of breaking the invariants of any deadlock-free routing designed for the fault-free case. Deadlock avoidance or recovery in presence of faults therefore has to be addressed. The new routing scheme originating from adaptation to a fault should preferably be guaranteed deadlock free. Beyond that, it is also necessary to avoid deadlocks that can result from the interplay between the old and the new routings in a transition phase, while the network is being reconfigured. A theory and a method for the latter problem have been presented by Duato et al. [2005] and Lysne et al. [2005], respectively.

Fault tolerant routing is among the most active areas in recent NoC research. A multitude of routing algorithms have been proposed so that further structuring is required to gain access to this wide field. In the following subsections, we investigate fault tolerant routing algorithms according to the degree of globality in the employed fault information. We start with offline techniques (3.3.1) that can gather and process information with arbitrarily complex techniques. Any online fault adaptation must take into account the resource limitations imposed by the need to perform all processing in the NoC itself, in a distributed way. This can be achieved by delimiting the region of an NoC in which faults are allowed to influence the routing at a given NoC switch. For example, a remote fault may not have to be considered in a given part of the NoC. Subsequently, we investigate routing algorithms that consider purely local fault information of the switch itself and its immediate neighborhood (3.3.2). Some more knowledge about faults is implied by the assumption of properly shaped regions in which faults are fully contained (3.3.3). The introduction of a fault lookahead (3.3.4) allows making better informed choices. Finally, we investigate the global distribution of fault information (3.3.5).

3.3.1 *Offline routing adaptation.* Offline techniques for diagnosis and routing adaptation can be applied in the process of testing fabricated chips, in order to reconfigure partially faulty chips to become usable. Another use case is in the start-up phase of a NoC based system, during which a built-in self-test with subsequent reconfiguration may be performed. Thus, offline techniques are effective against permanent faults; however, they cannot react fast enough to deal with transient or intermittent ones. After computing routing changes offline, they must be stored in the NoC. This typically requires the use of some sort of routing table. Keeping storage requirements low and offline adaptation efficient are the main challenges in this field.

Techniques to reduce the size of routing tables for irregular mesh-based NoCs have been proposed by Bolotin et al. [2007]. A *turns table* keeps information only for those destination addresses that require a packet to take a turn at a given switch. The approach makes use of the fact that packets are more likely to pass straight through, and defines this as the default in absence of a table entry. The authors propose a second scheme, XY deviation table, which assumes XY routing as the default, implements it with combinational logic, and stores only irregularity-induced deviations with routing table entries. An alternative representation of routing deviations from a basic scheme is proposed by Rodrigo et al. [2010]. A small number of *derouting bits* are sufficient to store routing deviations that are independent of the destination address. This principle is used by a series of routing implementations classified as *Logic Based Distributed Routing* (LBDR).

Another technique for storage-efficient routing tables is *region based routing* as proposed by Flich et al. [2007] and Mejia et al. [2009]. From known routing restrictions imposed on a basic routing scheme, available routing paths are computed and regions that can be reached with the same routing are identified. The routing information for each region is then merged and stored efficiently in a region-based table. Region based routing is often combined with segment based routing [Mejia et al. 2006] as the underlying routing algorithm. This offline technique has been applied to meshes and tori. It partitions the network into segments in which restrictions can be placed independent of other segments. These restrictions are employed to avoid faulty network elements and to break cycles that could otherwise cause deadlock.

An approach that updates regular routing table entries based on *rerouting rules* is described by Fick et al. [2009]. The occurrence of a fault effectively disables certain routing turns, which means that other routing restrictions originally introduced to avoid deadlock become superfluous. The identification and deadlock-free choice of routing turns to be re-enabled are based on a heuristics.

3.3.2 *Strictly local adaptation.* Local adaptation techniques make changes local to a single switch in order to recover it fully or partly from from faults. This approach assumes a fault model on a finer granularity than switches. Most work on fault tolerant routing in off-chip interconnection networks, on the other

hand, takes a faulty switch fully out of service. This distinction is reasonable as faulty switches can be replaced with little effort if they are discrete components, whereas an on-chip switch that is taken down will permanently limit communication performance until the end of the chip's lifetime.

The dynamic adaptive (DyAD, [Hu and Marculescu 2004]) routing method employs routing adaptation to avoid congested switches. When there is no congestion, minimal routing is performed based on a deterministic subset of odd-even routing. Only in case of congestion, the odd-even scheme's adaptivity is employed to choose the less congested of multiple routing alternatives at the cost of potentially non-minimal paths. The same principle could be employed to avoid faults, but fault tolerance would be limited by the degree of adaptivity available.

A large range of NoC switch designs, from unprotected ones to switches equipped with online repair and recovery capabilities, have been investigated by Constantinides et al. [2006] with respect to cost and reliability. Fick et al. [2009] equip a switch with the capability to replace a defective submodule, e.g. a FIFO buffer for incoming packets, by sharing a corresponding, intact submodule with another port. Faulty crossbar connections can be avoided by using a special escape bus. This way, many faults can be contained locally in the switch at the cost of some performance degradation. Only if the number or locations of faults exceed the built-in fault tolerance, offline routing modifications described above (cf. Section 3.3.1) are triggered. Similar to the escape bus concept, switch internal default backup paths have been proposed [Koibuchi et al. 2008]. These are employed to form a deadlock-free, ring topology backup network with two virtual channels. Concatto et al. [2009] describe another approach featuring redundancy of multiplexers and sharing of FIFOs to circumvent faults on the switching path. Temporal and spatial redundancy of multiplexers as means against SEU have been investigated by Eghbal et al. [2010]. Crosspoint redundancy in NoC crossbar switches is proposed by Grecu et al. [2006] in order to increase yield. Yu et al. [2011] propose a self-correcting arbitration mechanism capable of tolerating transient errors. The approach described by Kim et al. [2006] is similar in using modular redundancy without creating overhead through extra module instances. Functionality is borrowed from adjacent switches to recover from the defectiveness of a switch's own submodules.

Based on the model of crossbar faults (cf. Section 3.1), routing can be adapted to avoid faulty parts of the data path inside of its own [Kohler and Radetzki 2009]. This comes at the cost of sending a packet on a detour, which is enabled in a deadlock-free way by employing deflection routing as the underlying routing scheme. Potential livelock due to packets sent on an infinite series of detours is avoided by increasing packet priority and thus reducing the risk of detour with increasing packet age. Local rerouting to circumvent link faults has also been proposed [Alaghi et al. 2008], in some cases combined with packet replication. This work does not elaborate, however, on the matter of preventing deadlock or livelock.

An approach to consider reliability through fault tolerance jointly with quality of service is described by Kakoee et al. [2011b]. A dual physical channel NoC switch is designed to separate QoS traffic from best effort traffic without the control overhead introduced by virtual channels. The redundancy of physical channels is also exploited to uphold communication when one of the channels fails.

Kohler and Radetzki [2009] describe the hardening of critical fault tolerance infrastructure (storage and communication of fault information) against faults of its own. Error-detecting codes (parity block codes) are employed when information can be recovered; otherwise error-correcting codes (Hamming) are used. A two-dimensional scheme with row and column parity bits is proposed by Eghbal et al. [2010] to protect routing tables against SEU of their storage elements.

3.3.3 *Fault regions.* By providing each switch with information of the fault status of its immediate neighbours (to which it is adjacent), a more sophisticated routing around faults can be implemented. The basic idea is to identify faulty regions and to modify the routing of intact switches on the border of these regions so that packets are routed along the borderline. By proper combination with (and modification of)

Approach	Method	link/port/buffer	crossbar	arbiter	routing logic	routing table	fault state
BulletProof [Constantinides et al. 2006]	application of generic FT techniques (TMR, spares) on switch and component level	X	X	X	X	-	-
Vicis [Fick et al. 2009]	internal resource sharing	X	-	-	-	-	-
	escape bus	-	X	-	-	-	-
RoCo [Kim et al. 2006]	external resource sharing	-	-	X	X	-	-
	bypass path	X	X	-	-	-	-
DBP [Koibuchi et al. 2008]	default backup path to from ring topology	X	X	X	X	n/a	-
Concatto et al. [2009]	redundant MUXes, sharing of FIFOs	X	X	-	-	-	-
Eghbal et al. [2010]	temporal and spatial MUX redundancy	-	X	-	-	-	-
	row and column (2D) parity scheme	-	-	-	-	X	-
Grecu et al. [2006]	crosspoint redundancy	-	X	-	-	-	-
Yu et al. [2011]	self-correcting arbitration	-	-	X	-	-	-
Alaghi et al. [2008]	routing that avoids use of faulty links	X	-	-	-	-	-
Kohler and Radetzki [2009]	routing restricted to using intact components	X	X	n/a	-	n/a	-
	error detecting / correcting code	-	-	n/a	-	n/a	X
	sanity check (permutation property of routing)	-	-	n/a	X	n/a	-
Kakoee et al. [2011b]	dual (redundant) physical channels in switch	X	X	X	X	X	-

Table II. Overview of fault tolerance techniques local to individual switches.

deadlock-free distributed routing regimes, the presence of resource dependency cycles and thus deadlocks can be prevented at design time. Most of the work in this category assumes a 2D or higher-dimensional mesh as the underlying topology. The efficiency of the different approaches varies widely depending on the definition of fault regions because fault regions usually contain intact switches which have to be deactivated in order to avoid deadlock.

The smallest possible fault region consists of a *single switch*. A classic theorem by Glass and Ni [1993] states that $n-1$ faults can be tolerated in an n-dimensional mesh with a variant of negative first routing. This does not preclude the possibility of tolerating a larger number of faults with other methods. Zhang et al. [2008] propose the formation of cycle-free contours around individual faulty nodes in a mesh. X first routing is employed as the basic routing regime. Packets are routed around the fault location along the contour in which two turns are prohibited to avoid deadlocking cycles. The work is based on a single fault assumption, at least in the experimental investigations.

A notable number of works investigate routing around *rectangular regions* in a mesh. Bobda et al. [2005] employ XY routing as the basic routing technique. Adaptivity is achieved by surrounding modes that allow a packet to switch from Y back to X direction upon reaching or leaving the region boundary. A dedicated stamp bit in the packet header is set to ensure that a packet does not turn back and oscillate when on a detour. Schäfer et al. [2005] use the adaptivity of the odd-even turn model to route around rectangular blocks and suggest a block placement method. The motivation for both works lies in the need to integrate larger IP blocks that disturb a regular 2d mesh topology in a heterogeneous NoC. While the same technique is applicable to fault regions in principle, the need to enclose faults in rectangles may lead to the deactivation of many intact switches if more than a single switch fails.

Virtual channels have been used to ensure absence of deadlock while routing around *multiple rectangular*

fault regions. The relevant body of work in this field includes many pre-NoC era papers on fault tolerant routing in interconnection networks. Boppana and Chalasani [1995] modify e-cube (dimension order, DOR) routing for routing along fault rings and fault chains that enclose rectangular fault regions, and employ four virtual channels. VCs increase adaptivity by allowing multiple deadlock-free DOR rounds, one per VC. Research on reduction of VC requirements has culminated in the work by Fukushima et al. [2009] which requires no VCs at all. With three virtual channels, Chen and Chiu [2001] and Duan et al. [2009] allow fault regions to have certain non-convex shapes such as L or T shape, using variants of dimension order routing. Xinming and Xuemei [2010] suggest the use of four virtual subnetworks for routing around rectangular fault regions in a dual-torus topology termed PRDT(2,1).

Wu and Wang [2002] allow multiple fault regions of even more *complex shape*, namely rectlinear-monotone polygonal fault blocks. These are the union of overlapping rectangular fault regions and allow to enclose multiple faults in a shape smaller than a single large rectangle. Disjoint fault regions are also supported, but they must be more than two hops apart, otherwise they are joined into a single, larger region. The odd-even turn model is used as the basis for deadlock-free routing. Switches not bordering to fault regions use the degree of freedom available in the odd-even regime to target minimal (Manhattan) routing. If the packet reaches an obstacle, the packet is routed along its boundary either clockwise or counter-clockwise depending on which of these two options involves no forbidden turns. More recently, another turn model based routing has been proposed [Jovanovic et al. 2009], claiming to reach all nodes that are not completely blocked by faulty ones while ensuring deadlock-freeness without virtual channels.

3.3.4 *Routing with fault lookahead.* Fault regions tend to contain nodes that are healthy and connected. Their deactivation can result in an unnecessarily severe performance degradation. This downside comes from the favorable property that only the nodes at the boundary of a fault region need to have knowledge of the fault. A lookahead to faults farther away, as proposed by Feng et al. [2010a], enables more complex, even concave fault regions. These fault regions include less healthy nodes. The exact shape of the regions has been studied for a lookahead of two hops. An extension to longer-ranging lookahead and the systematic construction of the corresponding fault shapes would be of interest, at least from a theoretical perspective. Practical issues, e.g. the additional wires needed to communicate fault information over more than two hops, are arguments against an extension of the lookahead. In any case, a set of permitted fault region shapes is defined in which faults have to be included. The process of covering a given set of faults with minimal fault shapes is not yet defined and can likely not be performed online.

Table III gives an overview of properties of the locally informed fault tolerant approaches from Sections 3.3.3 and 3.3.4. It uses the acronyms DOR (dimension order routing), WH (wormhole switching), VCT (virtual cut-through), and SAF (store-and-forward). Question marks denote information that is not explicitly stated in the paper.

3.3.5 *Global topology exploration.* Conceptually, a fault lookahead as wide as the network diameter can be implemented, which would give each node complete information of the status of all other nodes. Techniques for global topology exploration aim at the same goal with less overhead for status communication. *Distance vector routing* [Malkin and Steenstrup 1995], invented for macroscopic networks, exchanges information on the delivery time of packets between adjacent switches. Each switch manages a routing table whose information is filled based on the information in its immediate neighbors' routing tables and the latency of the hop to each neighbor. Thereby, delivery time is computed transitively in a distributed fashion. The concept has been employed in the NoC context [Schönwald et al. 2007] where switches exchange distance information (hop count as a replacement of time) in-band with a special type of flits. Latency is minimized by a force directed wormhole routing algorithm for load balancing and avoiding faulty links and switches. Apparently, deadlocks are not accounted for; evaluation relies on a pattern of three faulty switches

Approach	Topology	Fault model	Fault tolerance	Virtual channels	Routing	Switching
Glass and Ni [1993]	n-mesh	node	$n-1$ nodes	none	negative first	WH
Zhang et al. [2008]	2-mesh	node	1 node	none	X first	WH?
Boppana and Chalasani [1995]	2-mesh	node, link	rectangles	4	e-cube	WH
Xinming and Xuemei [2010]	dual 2-cube	node	rectangles	4	vector routing	?
Chen and Chiu [2001]	2-mesh	node	non-convex	3	e-cube	WH
Duan et al. [2009]	2-mesh	node, link	non-convex	3	DOR "FTR4WM"	WH
Fukushima et al. [2009]	2-mesh	node	rectangles	none	DOR	?
Wu and Wang [2002]	2-mesh	node	polygones	none	odd-even	WH?
Jovanovic et al. [2009]	2-mesh	node	connected	none	odd-even	WH
Feng et al. [2010a]	2-mesh	link	non-convex	none	deflection	SAF

Table III. Overview of locally informed fault tolerant routing (? = not explicitly stated in paper).

that may not cause deadlock. In NoC context, Ali et al. [2005] state that distance vector routing can cause a "count-to-infinity" problem when delays are added up in a cycle. A refined algorithm, *link state routing* [Moy 1995], avoids this problem by constructing a network graph in each node to gain a complete view of the network. This requires processing capability that would typically not be implemented in hardware. Therefore, link state routing is more suitable to be implemented as software in CPU cores, which makes it most suitable for source routing (cf. Section 4).

Q routing [Boyan and Littman 1994], inspired from machine learning techniques, follows a similar approach in distributing delivery time or hop count information over the network. However, routing tables are updated only gradually. A learning factor α limits the gradient at which values in the routing tables can change. This slows adaptation but may make the routing more robust against short-term fluctuations, similar to an attenuated control loop. Q routing has first been employed in NoCs by Majer et al. [2005] to route in dynamically changing networks on reconfigurable fabric, but not considering faults explicitly. Feng et al. [2010b] take faults into account by a Q routing algorithm with learning factor $\alpha = 1$, which is equivalent to distance vector routing. Switches exchange distance information off-band, with dedicated signals. A hierarchical routing table is proposed to avoid the overhead of fully implemented global routing tables. The table of a given switch holds information of switches in the same quadrant of a mesh network only, and has one entry for each of the other three quadrants. The algorithm is able to tolerate all faults as long as the network is still connected. Deadlocks are avoided through the deflection principle.

Most work in fault tolerant routing, even when able to explore an unknown toplogy, assumes a regular mesh as the fault-free basis, which may be distorted by faults. Recent work [Radetzki 2011] supports an *arbitrary connected topology* and takes the design-time knowledge of a given topology into account in order to transmit and store the fault-induced deviations only. This reduces the storage requirements of routing tables significantly by subtracting the redundant design-time information on the base topology. Moreover, the fact that relative distance information is transmitted allows hierarchical routing tables to be optimized individually for each switch instead of assuming a global segmentation into, e.g., quadrants.

Another class of *topology-agnostic methods* (see [Flich et al. 2012] for a survey of this class of routing algorithms) explores an unknown or altered topology by flooding the network with exploration packets / flits to build a spanning tree, as first proposed by Schroeder et al. [1991]. Deadlock-free routing can then be

ensured by routing packets in this tree, upwards hops strictly before downwards hops (up*/down* routing). In a fault tolerant variant, Immunet [Puente et al. 2008], a switch that detects a fault starts the exploration process, thereby becoming the root of the spanning tree. By means of a depth-first tree traversal, a deadlock-free, ring topology virtual network is constructed. This safe network serves as an escape channel open to packets that cannot be routed in the main virtual channel due to fault effects. Ariadne [Aisopos et al. 2011] features a refined distributed route-finding process during which each node broadcasts a route-finding packet along the spanning tree following the up*/down* scheme to program routing tables in all other nodes. In this process, shortcuts to bypass higher-layer nodes may be established that transform the spanning tree into a directed acyclic graph. Route finding pre-empts regular operation, however, and despite shortcuts significant traffic is aggregated towards the root node.

3.4 Summary

With global fault-adaptive routing, be it determined online or offline, it is typically possible to utilize any remaining connectivity of an NoC affected by faults. This superior error correcting capability comes at the cost of limited scalability with increasing network size. Distribution of routing determination as enabled by stronger locality (fault regions or lookaheads) enhances scalability but suffers higher potential performance degradation, e.g. due to suboptimal routes or unreachability of healthy nodes. Strictly local adaptation of individual switches is the least costly of all alternatives, but can hardly achieve guarantees with respect to properties such as performance or deadlock-freeness.

Compared to these structural redundancy techniques, various forms of informationally redundant routing impose a significantly higher overhead by replicating each packet at least once. In the case of point-to-point communication, random walk is the only realistic option. Flooding can be an alternative in broadcast scenarios or when unknown topology shall be explored. However, with the exception of some LBDR variants [Rodrigo et al. 2008], broadcast or multicast routing in conjunction with fault tolerance has received very little attention yet in the NoC context. In view of the importance of efficient one-to-many communications, e.g. for cache coherency or collective MPI (message passing interface) operations, their fault tolerant design still offers open research opportunities.

A general weakness of most works on network layer lies in missing (or at least undescribed) diagnosis capabilities. The design of fault tolerant routing for assumed network level faults is of little value without a diagnosis of these faults. Of course, established structural diagnosis techniques exist for digital circuits on gate level. However, deduction of network faults from structural faults is a challenging open problem. Functional diagnosis techniques for network faults may be an alternative worth investigation.

4. TRANSPORT LAYER (END-TO-END COMMUNICATION)

The transport layer, as shown in Figure 1, provides end-to-end communication services between network terminals. Fault tolerant techniques are needed to support reliable end-to-end communication for NoC. Fault tolerant schemes on the transport layer can be classified as time redundancy techniques (automatic repeat request (ARQ) schemes), information redundancy techniques (forward error correction (FEC) schemes), hybrid ARQ/FEC schemes [Ejlali et al. 2007] and spatial redundancy techniques. In this section, we briefly introduce the fault model for this layer first, then describe how to detect and locate errors, followed by four fault tolerant techniques.

4.1 Fault models

For the transport layer, three types of faults (transient, intermittent and permanent faults) affect either the correctness of the delivered data, the correctness of the destination address or completely loss of packet(s). Most literature about end-to-end error detection and protection use implicit fault models that basically as-

sume that some of the bits in a packet are corrupted by transient or permanent faults. A *functional fault model* refers to the structural faults in the multiplexer of the router datapath [Raik et al. 2007]. For the crossbar multiplexer, tests for each address value are performed to fully cover the structural faults. Transient faults which lead to *packet corruption* or *loss* are handled by Ali et al. [2007]. These fault modes can be summarized as *coarse granularity fault models* that care about whether the end-to-end path contains faults or not. They abstract from specific fault locations or mechanisms.

4.2 Error detection and location

In the end-to-end communication, error control codings for error detection and correction can be used to encode the packet. An encoder is added to the sender network interface (NI) and a decoder is added to the receiver NI. For error detecting codes (e.g. parity codes or Cyclic Redundancy Check (CRC) codes), the sender NI has one or more packet buffers in which it stores the packets that are transmitted. The receiver NI sends an NACK or an ACK packet back to the sender, depending on whether or not it has detected an error. Disadvantages of this scheme are that it can only handle transient faults and it cannot locate the position of the fault. If the fault is permanent or intermittent, it is likely that the retransmitted data is affected by the same fault. Because the checking takes place only at the receiver NI, it is impossible to locate the position of the fault precisely. For error correction codes (e.g. a single-error-correcting double-error-detecting code, SECDED), the receiver NI can correct both transient and permanent faults. However, only limited number of faults can be handled and it gets overwhelmed when permanent faults accumulate over time.

To locate faulty links in the network, Raik et al. [2007] extend the use of test configurations for diagnostic purposes. The diagnosis algorithm applies three test configurations: (a) straight paths, (b) turning paths, and (c) resource connections to cover the entire network, as shown in Figure 10. Configuration (a) lets the packets pass straight through the network and covers the faults in all straight connections in the switches. Configuration (b) takes advantage of the deterministic XY routing to test the turning paths. Configuration (c) is needed to cover the links to resources. These functional test configurations can locate faults in individual links between switches and the connectivity inside switches. These three test configurations can achieve high fault coverage for the crossbar switch and the I/O registers. However, during the test the network cannot run applications and the authors do not provide a strategy to determine when to run this test.

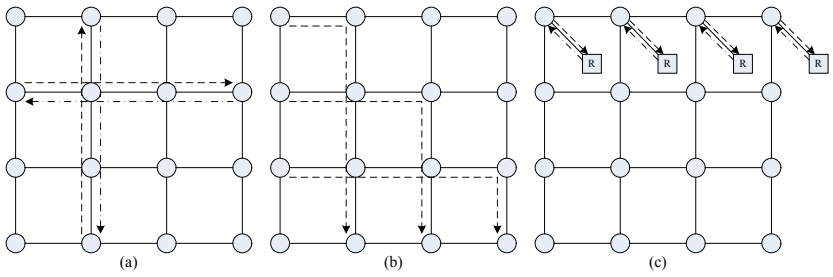

Fig. 10. Test configurations in [Raik et al. 2007]: (a) straight paths; (b) turning paths; (c) local resource connections.

A probabilistic method to locate the defective wire has been proposed by Shamshiri et al. [2011]. The network interface of the destination node observes the error pattern on each packet using an end-to-end Error Correcting Code (ECC), code that is a four interleaved error-locality-aware code for 64-bit wide links. The network interface of the destination node observes the error pattern on each packet. Whenever a destination node observes a potential permanent error on a bit position, it sends a control message to a host processor indicating the bit position of the error(s) and the source and destination of the erroneous packets. The host

processor is one of the cores of the mesh that acts as a master and dispatches tasks to other cores. The host processor uses a scoreboard to collect information about the error patterns. When the number of erroneous bits under the same column of the scoreboard is greater than a threshold, a permanent error can be deduced on a wire at the corresponding position along the path. To illustrate this probabilistic diagnosis method, Figure

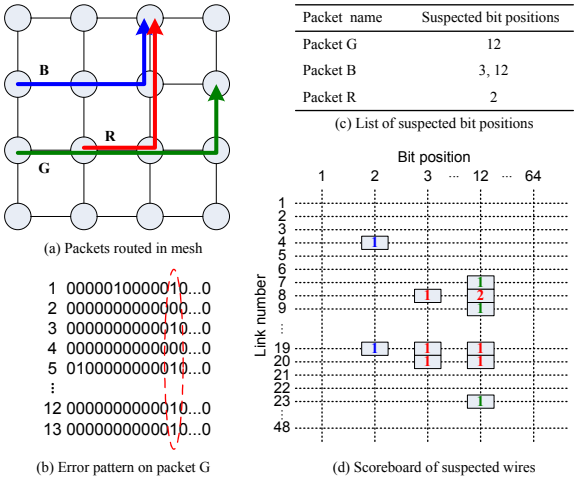

Fig. 11. An example of the probabilistic end-to-end diagnosis method[Shamshiri et al. 2011].

11(a) shows a 4×4 mesh with three packets routed with XY routing: B, G, and R. Figure 11(b) shows an exemplary error pattern observed on packet G. Figure 11(c) shows the suspected bit positions observed at the destination of each packet in this example. Figure 11(d) shows the contents of the scoreboard after these three erroneous packet transfers. Using the scoreboard, the host can observe that wire 12 of link 8 is more likely to be defective because it was a suspected wire twice.

4.3 Time redundancy techniques

ARQ is a time redundancy error-control method for data transmission which uses acknowledgements and retransmissions to achieve reliable data transmission. In end-to-end ARQ schemes, an error detection code is used to encode the packet at the sender NI, and the receiver NI decodes the packet and sends an acknowledge (or not-acknowledge) signal/packet back to the sender. A packet buffer is needed to buffer packets which must be retransmitted due to errors. When the sender receives the not-acknowledge signal/packet, it retransmits the packet again. To account for errors on the ACK or NACK signal/packet, a time-out mechanism for retransmission is typically added in the sender NI. If the sender does not receive an acknowledgment after a predefined time, it retransmits the packet until receiving an acknowledgment or exceeding a predefined number of retransmissions. Figure 12 illustrates the end-to-end ARQ architecture. If the ACK or NACK is transmitted via dedicated wires, for handling errors in the dedicated control signals, hardware redundancy such as triple modular redundancy (TMR) can be used.

A simple and cost-effective end-to-end packet-based communication protocol to address transient faults for NoC has been proposed by Ali et al. [2007]. In the Go-back-n retransmission policy the receiver generates a single ACK for a pre-defined set of packets that it receives, hence reducing the amount of traffic as compared to acknowledging every packet. The buffers are only maintained at the sender's side where packets are kept until an ACK arrives. If a packet is corrupt or missing, the receiver requests for a retransmission

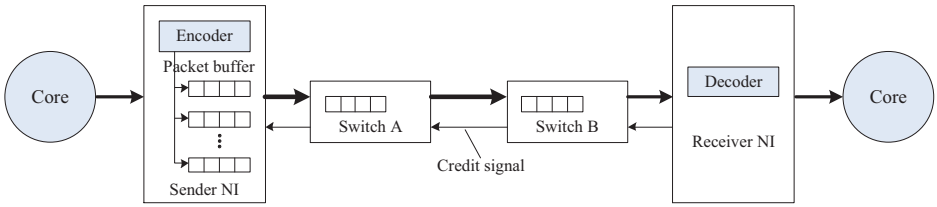

Fig. 12. End-to-end ARQ architecture [Murali et al. 2005].

and discards all the incoming packets until it receives the required packet. The advantage of this scheme is that the receiver receives packets in sequence, so no reordering and buffering of packets is required at the receiver side. A timeout mechanism is implemented both at the sender and receiver. The timeout depends on the maximum time it takes for the n packets to receive the receiver plus the time the ACK packet needs on the way back. Figure 13 illustrates three cases of a packet affected by a transient fault.

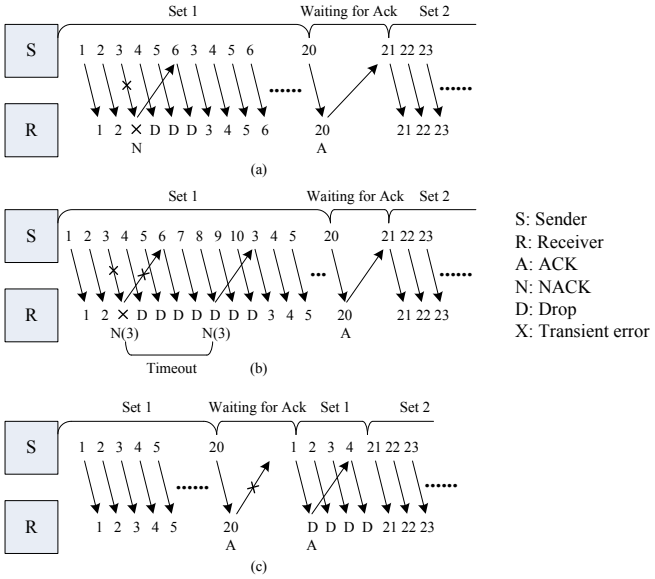

Fig. 13. Three cases of the End-to-end ACK protocol [Ali et al. 2007].

4.4 Information redundancy techniques

For an FEC (Forward Error Correction) scheme, the sender uses an error correction code to encode packets and the receiver decodes packets and corrects errors without any retransmission request. FEC techniques utilize information redundancy to achieve fault tolerance in end-to-end communication. The two main categories of FEC codes are block codes and convolutional codes. Block codes [Huffman and Pless 2003] with polynomial decoding complexity work on fixed-size packets. Practical block codes can generally be decoded in polynomial time to their block length. Convolutional codes [Viterbi 1971] with linear decoding complexity work on bitstreams of arbitrary length. They are most often decoded with the Viterbi algorithm [Forney 1973], though other algorithms are sometimes used. Hamming codes are one of classical block

codes which are mostly used in on-chip end-to-end communication. Lehtonen et al. [2007a] analyzed different error correcting coding methods (Hamming, BCH and RS codes) for NoC in terms of error correction capability, circuit area and power consumption.

Jantsch et al. [2005] have concluded that transport layer error protection is more power efficient than both no protection and link layer protection. In this work, end-to-end error protection is added for the payload and the header is still protected on the link layer. For the payload stronger codes can be used to compensate for the accumulation of errors over multiple hops. The header should be protected at each link since if the address is corrupted, the packet will not be delivered correctly.

Three end-to-end error recovery schemes are compared in [Murali et al. 2005]. Parity code, CRC code and single-error-correcting multiple-error-detecting code (SECDED) are used to encode the packet in end-to-end communication. To account for errors on the ACK or NACK packets, a time-out mechanism for retransmission is also used at the sender. To detect reception of duplicated packets, packet sequence identifiers are used. In the coding scheme, the receiver corrects any single bit error on a packet, but for double and some multiple bit errors, it requests end-to-end retransmission of data from the sender NI. From the energy perspective, compared with link level error control schemes, this paper has concluded that for networks with long links or hop counts, end-to-end error control schemes are more power efficient.

Both ARQ and FEC schemes can suffer from errors in the header flit. If the destination address of a packet is corrupted during the transfer, the packet might be routed to a wrong destination. Moreover, if the source node address is corrupted, the ARQ technique cannot send the retransmission request to the correct source. Thus, it is very important to avoid corruption of the header information. Both in [Jantsch et al. 2005] and [Murali et al. 2005], the header is encoded individually and checked at each hop traversal.

4.5 Hybrid time/information redundancy

In hybrid ARQ/FEC (HARQ) schemes, the sender encodes packets using an error correction code. When the receiver can correct a detected error, it does not request a retransmission. However, when the receiver detects an error that it cannot correct, it requests the sender to resend the packet. This process is repeated until the packet is error free.

Ejlali et al. [2007] compared the ARQ, FEC and hybrid ARQ/FEC schemes (HARQ) in terms of performance, fault tolerance and energy efficiency. The work considers only the energy consumption of on-chip wires and error control circuit, and does not take the energy consumption of the buffers in the NI into account. For the environments with relatively low noise power, ARQ is the most preferable choice, however, as the noise power increases, HARQ proves more advantageous than the other schemes. For relaxed time constraints, ARQ is better than FEC. However, when tight time constraints are required, ARQ is not effective at all and FEC is the most preferable choice. If FEC and HARQ provide the same performability (the metric to measure the performance and reliability of communication schemes) and consume the same energy, FEC is more preferable because of its simpler implementation.

Another HARQ scheme has been proposed by Rossi et al. [2007], depending on the particular application, the error control scheme can be configured by the user in three different operating modes: correction mode, detection mode, and mixed mode. For each configuration mode, packet header and payload are encoded with different error control policies (varying from the end-to-end to the router-to-router control policy), considering SEC Hamming codes, SEC/DED Hsiao codes and Symbol Error Correcting codes. When the proposed scheme is configured in the correction mode, an error correcting code can guarantee that corrected packets are always forwarded, thus incurring minimum latency. In the detection mode, the decoding part for error correction is disabled. The retransmission scheme guarantees data integrity by assuring that the data used by the receiver are correct. In the mixed mode, different parts of the transmitted packets are protected by means of different error control approaches. For instance, errors affecting the header flit can be corrected,

while errors in the payload flits can be detected and retransmitted.

4.6 Spatial redundancy

Source routing can be used to achieve spatial redundancy on the transport layer, by selecting another route when links or routers on a given route are identified as erroneous. Kim and Kim [2007] have proposed a fault tolerant source routing for Network-on-Chip (called SRN). SRN is composed of two mechanisms of *Route Discovery* and *Route Maintenance*, which work together to allow nodes to discover and maintain source routes to arbitrary destinations in NoC. When a node S finds no route to a destination D in its *Route Cache*, it will initiate the *Route Discovery* protocol to dynamically find a new route to D. For *Route Maintenance*, each node transmitting the data is responsible for confirming that the packet has been received by the next hop along the source route. The data is retransmitted until this confirmation of receipt is received. Another spatial redundancy technique sends packets across multiple non-intersecting paths to protect against permanent link failures [Murali et al. 2006]. The non-intersecting nature of the paths makes sure that a link failure on one path does not affect the packets that are transmitted on the other paths.

The Network Interface (NI) itself can also be protected by providing spare resources and allowing for reconfiguration. Fiorin et al. [2011] propose a fault tolerant NI architecture to handle the faulty behavior in NI itself. The LUT, FIFOs and FSM in NI are protected with error correcting and detecting codes and the remaining components are implemented with TMR techniques.

If a router or NI fails, the attached processing element cannot send or receive packets. The multi network interface (multi-NI) architecture connects at least two NIs to each processing element [Rantala et al. 2009]. Koupaei et al. [2011] propose a reconfigurable NI with two local ports, a main local port and a backup local port. The NI can be reconfigured to tolerate certain NI internal faults and a broken primary local port.

Besides multiple NIs, an alternative end-to-end fault tolerant solution is to use a secondary verified network to transfer data checksums for error detection and recovery. Parikh and Bertacco [2011] propose a solution (ForEVeR) which combines formal methods and runtime verification to ensure functional correctness in NoCs. ForEVeR uses a secondary network which can be referred as a redundancy resource to transfer notifications and for recovery data packets to achieve fault tolerance.

4.7 Summary

In this section, we have reviewed the fault tolerant techniques which refer to the time redundancy, information redundancy, hybrid time/information redundancy and spatial redundancy for transport layer of NoCs. Table IV summarizes the fault types that the fault tolerant techniques for transport layer can handle. Time redundancy technique utilizes retransmission between sender and receiver NIs to handle transient faults. Error correcting codes are used for encoding packets in information redundancy technique. Due to the limited error correcting capability, information redundancy technique can only handle transient faults and partial permanent faults. The hybrid scheme overcomes the limit of the information redundancy and provides more reliable end-to-end transmissions. Redundant NIs and fault tolerant source routing belong to the spatial redundancy technique to process permanent faulty links and routers. Future research is needed to pay more attention to the realistic fault models which reflect real fault conditions in CMOS technologies of 20nm and below. In addition, locating errors accurately in end-to-end communication is a difficult and challenge work for future research.

5. CONCLUSION AND OUTLOOK

Networks on Chip are a special class of interconnection networks. Research on fault tolerant NoCs can therefore benefit from findings in this more general field. However, the constraints imposed by on-chip implementation pose a number of special challenges that justify NoC fault tolerance as a genuine area of

Table IV. Summary of fault tolerant techniques for transport layer

Fault tolerant techniques	Fault type	Related work
Time redundancy	transient faults	[Ali et al. 2007]
Information redundancy	transient faults and limited permanent faults	[Murali et al. 2005] and [Jantsch et al. 2005]
Hybrid ARQ/FEC scheme	transient faults and limited permanent faults	[Ejlali et al. 2007] and [Rossi et al. 2007]
Spatial redundancy	permanent faults	[Fiorin et al. 2011], [Koupaei et al. 2011] [Murali et al. 2006], [Parikh and Bertacco 2011] [Rantala et al. 2009] and [Kim and Kim 2007]

research. In particular, NoC solutions are typically implementented largely in hardware, and replacement of faulty parts is not an option. A considerable body of new research addressing these special concerns has emerged recently. In particular, NoC time and information redundancy techniques appear to be well researched on data link layer, and a large number of fault tolerant NoC routing approaches have already been published.

Despite these remarkable achievements, a fault tolerant NoC system solution has not emerged yet. Most current research considers fault tolerance and fault diagnosis in isolation from each other, and delimits the solution space to a single layer. All proposals assume, explicitly or implicitly, a fault model. Still, the state of the art regarding fault models is not satisfactory. Different abstractions of faults are needed at the different levels but the link between these abstraction levels and the physical processes is not well established or understood. Fault models are often motivated by the ease of use in analyzing a specific technique but lack a clear and quantitative connection to the physical failure mechanisms. An obvious difficulty is the secrecy of manufacturing data that would be required to establish accurate, quantitative fault models. Foundries do not make this data available, which means that academic research can only guess and make assumptions. The semiconductor industry could help the research by releasing useful fault models that, without revealing secrets, could serve as benchmarks against which proposed techniques are measured.

Yu and Lemieux [2005] and Breuer et al. [2004] offer discussions about the granularity at which redundancy and fault tolerance techniques are most efficient. Corresponding research would be a valuable contribution to the field of NoCs. Cross layer approaches are likely to provide better trade-offs of fault tolerance versus other design criteria such as area cost, performance, and power. For example transient faults could be captured at the link layer, permanent faults at the network layer, and rare faults that still escape are dealt with at the transport or even application layer. In view of the well established body of individual network fault tolerance methods, future work should increasingly address the investigation of such trade-offs rather than proposing more isolated partial solutions. Moreover, every proposed method for tolerating or correcting errors should be integrated with a corresponding fault detection and diagnosis technique, because they cannot be assessed in isolation and have value in combination only.

Hence, fault tolerant NoC design has to become more goal-driven, with clearly specified objective functions and constraints. Currently, differing goals such as maximum reliability or graceful performance degradation exist, but few works specify their goal explicitly and in measurable terms. This complicates significantly any comparison of competing approaches or the selection of a most favorable one for a given goal. Moreover, whereas initiative has been taken to define standardized benchmarks for fault-free NoCs [Grecu et al. 2007], the evaluation of fault tolerance methods diverges widely. The inclusion of standardized fault scenarios in benchmark suites would further advance comparative research. Evaluation may also have to go beyond simulation in order to make firm quality of service guarantees in presence of fault tolerance, which has hardly been investigated yet.

Another critique relates to the size and topology of networks evaluated by current research. Most work investigates 8x8 meshes at best; the assumption of even smaller NoCs is not uncommon. However, NoCs

of this size, including processing cores, can still be produced fault-free with viable yield with current technology, making fault tolerance art for art's sake at this scale. Researchers should think ahead multiple technology generations, predict realistic failure and soft error rates, and evaluate networks of relevant sizes, which will be in the order of 16x16 two technology generations (four years) from now, and 32x32 = 1024 nodes by 2020. Obviously, scalability and possibly the need to introduce hierarchy will become much more important concerns. Also, the mesh is not the most popular topology in industry today; ring structures and irregular networks are more common. The implications of topology for a fault tolerance technique should receive more attention with focus on industrially relevant structures.

Due to advances in 3-D stacking, TSV (Through Silicon Via), and packaging technology, researchers have studied 3-D NoC structures. Also, optical and mixed electro-optical NoCs have received significant attention and will require specific fault tolerance solutions. Researchers should include these novel structures in their considerations when conceiving holistic solutions for fault tolerance.

In summary, even though a number of open issues and challenges still exist, there is an impressive body of insight and techniques available. Thus, when the research on fault tolerance in NoCs becomes more holistic, goal directed, and better linked to the physical failure causes, we can be optimistic that practical solutions will emerge, that produce fault tolerance at reasonable cost delivering NoCs with a desired level of resilience even if device behavior becomes more and more stochastic.

REFERENCES

AGARWAL, M., PAUL, B., ZHANG, M., AND MITRA, S. 2007. Circuit failure prediction and its application to transistor aging. In *VLSI Test Symposium, 2007. 25th IEEE*. 277 –286.

AISOPOS, K., CHEN, C.-H., AND PEH, L.-S. 2011. Enabling system-level modeling of variation-induced faults in networks-on-chips. In *Proc. 48th ACM/EDAC/IEEE Design Automation Conference (DAC)*. 930–935.

AISOPOS, K., DEORIO, A., PEH, L.-S., AND BERTACCO, V. 2011. Ariadne: Agnostic reconfiguration in a disconnected network environment. In *Proc. Parallel Architectures and Compilation Techniques (PACT)*.

ALAGHI, A., KARIMI, N., SEDGHI, M., AND NAVABI, Z. 2007. Online noc switch fault detection and diagnosis using a high level fault model. In *Proceedings of the 22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems (DFT)*. 21–29.

ALAGHI, A., SEDGHI, M., KARIMI, N., FATHY, M., AND NAVABI, Z. 2008. Reliable noc architecture utilizing a robust rerouting algorithms. In *9th IEEE East-West Design & Test Symposium (EWDTS08)*.

ALI, M., WELZL, M., AND HELLEBRAND, S. 2005. A dynamic routing mechanism for network on chip. In *Proceedings of the 23rd NORCHIP Conference*. 70–73.

ALI, M., WELZL, M., AND HESSLER, S. 2007. And end 2 end reliability protocol to address transient faults in network on chips. In *Digest of the Workshop on Diagnostic Services in Network-on-Chips*.

ANGHEL, L. AND NICOLAIDIS, M. 2000. Cost reduction and evaluation of a temporary faults detecting technique. In *Design, Automation and Test in Europe Conference and Exhibition 2000. Proceedings*. 591 –598.

ANGIOLINI, F., MELONI, P., CARTA, S., BENINI, L., AND RAFFO, L. 2006. Contrasting a noc and a traditional interconnect fabric with layout awareness. In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '06)*. Vol. 1. 1–6.

AVIŽIENIS, A., LAPRIE, J.-C., RANDELL, B., AND LANDWEHR, C. 2004. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing 1,* 1 (January-March), 11–33.

BAUMANN, R. 2005. Soft errors in advanced computer systems. *Design Test of Computers, IEEE 22,* 3 (may-june), 258 – 266.

BELL, S., EDWARDS, B., AMANN, J., CONLIN, R., JOYCE, K., LEUNG, V., MACKAY, J., REIF, M., BAO, L., BROWN, J., MATTINA, M., MIAO, C.-C., RAMEY, C., WENTZLAFF, D., ANDERSON, W., BERGER, E., FAIRBANKS, N., KHAN, D., MONTENEGRO, F., STICKNEY, J., AND ZOOK, J. 2008. TILE64 processor: A 64-core SoC with mesh interconnect. In *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC)*. 87–90.

BERTOZZI, D., BENINI, L., AND DE MICHELI, G. 2002. Low power error resilient encoding for on-chip data buses. In *Proc. Design, Automation and Test in Europe Conf. and Exhibition*. 102–109.

BERTOZZI, D., BENINI, L., AND DE MICHELI, G. 2005. Error control schemes for on-chip communication links: The energy-reliability tradeoff. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD) 24,* 6, 818–831.

BJERREGAARD, T. AND MAHADEVAN, S. 2006. A survey of research and practices of network-on-chip. *ACM Computing Surveys 38,* March, 1–51.

BOBDA, C., AHMADINIA, A., MAJER, M., TEICH, J., FEKETE, S., AND VAN DER VEEN, J. 2005. Dynoc: A dynamic infrastructure for communication in dynamically reconfigurable devices. In *Proc. Int Field Programmable Logic and Applications Conf*. 153–158.

BOGDAN, P., DUMITRAŞ, T., AND MARCULESCU, R. 2007. Stochastic communication: A new paradigm for fault-tolerant networks-on-chip. *VLSI Design 2007*, 17.

BOLOTIN, E., CIDON, I., GINOSAR, R., AND KOLODNY, A. 2007. Routing table minimization for irregular mesh nocs. In *Proc. Design, Automation & Test in Europe Conf. & Exhibition DATE '07*. 1–6.

BONDAVALLI, A., CHIARADONNA, S., GIANDOMENICO, F. D., AND GRANDONI, F. 2000. Threshold-based mechanisms to discriminate transient from intermittent faults. *IEEE Transactions on Computers 49,* 4, 230–245.

BOPPANA, R. V. AND CHALASANI, S. 1995. Fault-tolerant wormhole routing algorithms for mesh networks. *IEEE Transactions on Computers 44,* 7, 848–864.

BORKAR, S. 2005. Designing reliable systems from unreliable components: the challenges of transistor variability and degradation. *Micro, IEEE 25,* 6 (Nov.-Dec.), 10–16.

BORKAR, S. 2007. Thousand core chips: a technology perspective. In *Proceedings of the 44th annual Design Automation Conference*. DAC '07. ACM, New York, NY, USA, 746–749.

BOYAN, J. AND LITTMAN, M. 1994. Packet routing in dynamically changing networks: A reinforcement learning approach. *Advances in Neural Information Processing Systems 6*, 671–678.

BREUER, M., GUPTA, S., AND MAK, T. 2004. Defect and error tolerance in the presence of massive numbers of defects. *Design Test of Computers, IEEE 21,* 3 (may-june), 216 – 227.

CHEN, C.-L. AND CHIU, G.-M. 2001. A fault-tolerant routing scheme for meshes with nonconvex faults. *IEEE Transactions on Parallel and Distributed Systems 12,* 5, 467–475.

CONCATTO, C., MATOS, D., CARRO, L., KASTENSMIDT, F., SUSIN, A., COTA, E., AND KREUTZ, M. 2009. Fault tolerant mechanism to improve yield in nocs using a reconfigurable router. In *SBCCI '09: Proceedings of the 22nd Annual Symposium on Integrated Circuits and System Design*. ACM, New York, NY, USA, 1–6.

CONSTANTINESCU, C. 2003. Trends and challenges in vlsi circuit reliability. *Micro, IEEE 23,* 4 (July-August), 14 – 19.

CONSTANTINIDES, K., PLAZA, S., BLOME, J., ZHANG, B., BERTACCO, V., MAHLKE, S., AUSTIN, T., AND ORSHANSKY, M. 2006. Bulletproof: a defect-tolerant cmp switch architecture. In *Proc. Twelfth Int High-Performance Computer Architecture Symp*. 5–16.

COTA, E., KASTENSMIDT, F., CASSEL, M., HERVE, M., ALMEIDA, P., MEIRELLES, P., AMORY, A., AND LUBASZEWSKI, M. 2008. A high-fault-coverage approach for the test of data, control and handshake interconnects in mesh networks-on-chip. *IEEE Transactions on Computers 57,* 9 (sept.), 1202 –1215.

CUVIELLO, M., DEY, S., BAI, X., AND ZHAO, Y. 1999. Fault modeling and simulation for crosstalk in system-on-chip interconnects. In *Proc. Digest of Technical Papers Computer-Aided Design 1999 IEEE/ACM Int. Conf*. 297–303.

DALIRSANI, A., HOLST, S., ELM, M., AND WUNDERLICH, H. 2011. Structural test for graceful degradation of noc switches. In *Proc. European Test Symposium (ETS)*. 183–188.

DE MICHELI, G. AND BENINI, L. 2006. *Networks On Chips: Technology and Tools*. Morgan Kaufmann Publishers.

DODD, P. AND MASSENGILL, L. 2003. Basic mechanisms and modeling of single-event upset in digital microelectronics. *Nuclear Science, IEEE Transactions on 50,* 3 (june), 583 – 602.

DUAN, X., ZHANG, D., AND SUN, X. 2009. Fault-tolerant routing schemes for wormhole mesh. In *Proc. IEEE Int Parallel and Distributed Processing with Applications Symp*. 298–301.

DUATO, J., LYSNE, O., PANG, R., AND PINKSTON, T. 2005. Part i: A theory for deadlock-free dynamic network reconfiguration. *IEEE Transactions on Parallel and Distributed Systems 16,* 5, 412–427.

DUBROVA, E. 2008. *Fault-Tolerant Design: An Introduction*. Kluwer Academic Publishers.

DUMITRAS, T. AND MARCULESCU, R. 2003. On-chip stochastic communication [soc applications]. In *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'03)*. 790–795.

DUTTA, A. AND TOUBA, N. 2007. Reliable network-on-chip using a low cost unequal error protection code. In *Defect and Fault-Tolerance in VLSI Systems, 2007. DFT '07. 22nd IEEE International Symposium on*. 3 –11.

EGHBAL, A., YAGHINI, P. M., PEDRAM, H., AND ZARANDI, H. R. 2010. Designing fault-tolerant network-on-chip router architecture. *International Journal of Electronics 97,* 10, 1181–1192.

EJLALI, A., AL-HASHIMI, B. M., ROSINGER, P., AND MIREMADI, S. G. 2007. Joint consideration of fault-tolerance, energy-efficiency and performance in on-chip networks. In *Proceedings of the conference on Design, automation and test in Europe*. DATE '07. EDA Consortium, San Jose, CA, USA, 1647–1652.

ELAKKUMANAN, P., PRASAD, K., AND SRIDHAR, R. 2006. Time redundancy based scan flip-flop reuse to reduce ser of combinational logic. In *Proceedings of the 7th International Symposium on Quality Electronic Design*. ISQED '06. IEEE Computer Society, Washington, DC, USA, 617–624.

ERNST, D., KIM, N. S., DAS, S., PANT, S., RAO, R., PHAM, T., ZIESLER, C., BLAAUW, D., AUSTIN, T., FLAUTNER, K., AND MUDGE, T. 2003. Razor: a low-power pipeline based on circuit-level timing speculation. In *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-36)*. 7–18.

FENG, C., LU, Z., JANTSCH, A., LI, J., AND ZHANG, M. 2010a. FoN: Fault-on-neighbor aware routing algorithm for networks-on-chip. In *International SOC Conference*. Las Vegas, Nevada.

FENG, C., LU, Z., JANTSCH, A., LI, J., AND ZHANG, M. 2010b. A reconfigurable fault-tolerant deflection routing algorithm based on reinforcement learning for networks-on-chip. In *Proceedings of the International Workshop on Network on Chip Architectures (NoCArc)*. Atlanta, Georgia.

FICK, D., DEORIO, A., CHEN, G., BERTACCO, V., SYLVESTER, D., AND BLAAUW, D. 2009. A highly resilient routing algorithm for fault-tolerant nocs. In *Proc. Design Automation and Test in Europe (DATE)*. 21–26.

FICK, D., DEORIO, A., HU, J., BERTACCO, V., BLAAUW, D., AND SYLVESTER, D. 2009. Vicis: a reliable network for unreliable silicon. In *DAC '09: Proceedings of the 46th Annual Design Automation Conference*. ACM, New York, NY, USA, 812–817.

FIORIN, L., MICCONI, L., AND SAMI, M. 2011. Design of fault tolerant network interfaces for nocs. In *Proceedings of the 14th Euromicro Conference on Digital System Design*. 393–400.

FLICH, J., MEJIA, A., LÓPEZ, P., AND DUATO, J. 2007. Region-based routing: An efficient routing mechanism to tackle unreliable hardware in network on chips. In *Proc. Symposium on Networks-on-Chip (NOCS)*. Princeton, NJ, USA, 183–194.

FLICH, J., SKEIE, T., MEJIA, A., LYSNE, O., LOPEZ, P., ROBLES, A., DUATO, J., KOIBUCHI, M., ROKICKI, T., AND SANCHO, J. 2012. A survey and evaluation of topology-agnostic deterministic routing algorithms. *IEEE Transactions on Parallel and Distributed Systems 23,* 3 (March), 405–425.

FORNEY, G. D. 1973. The viterbi algorithm. *Proceedings of the IEEE 61,* 3, 268–278.

FRANTZ, A., KASTENSMIDT, F., CARRO, L., AND COTA, E. 2006a. Dependable network-on-chip router able to simultaneously tolerate soft errors and crosstalk. In *Proceedings of the IEEE International Test Conference (ITC)*. 1–9.

FRANTZ, A. P., CASSEL, M., KASTENSMIDT, F. L., COTA, E., AND CARRO, L. 2007. Crosstalk- and seu-aware networks on chips. *IEEE Design & Test of Computers 24,* 4, 340–350.

FRANTZ, A. P., KASTENSMIDT, F. L., CARRO, L., AND COTA, E. 2006b. Evaluation of seu and crosstalk effects in network-on-chip switches. In *Proc. Symposium on Integrated Circuits and Systems Design (SBCCI)*.

FU, B. AND AMPADU, P. 2009. On hamming product codes with type-ii hybrid arq for on-chip interconnects. *IEEE Transactions on Circuits and Systems I: Regular Papers 56,* 9, 2042–2054.

FUKUSHIMA, Y., FUKUSHI, M., AND HORIGUCHI, S. 2009. Fault-tolerant routing algorithm for network on chip without virtual channels. In *Proc. 24th IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems DFT '09*. 313–321.

FURBER, S. 2006. Living with failure: Lessons from nature? In *Proc. European Test Symposium (ETS)*. 4–8.

GADLAGE, M., AHLBIN, J., NARASIMHAM, B., BHUVA, B., MASSENGILL, L., REED, R., SCHRIMPF, R., AND VIZKELETHY, G. 2010. Scaling trends in set pulse widths in sub-100 nm bulk cmos processes. *IEEE Transactions on Nuclear Science 57,* 6, 3336–3341.

GANGULY, A., PANDE, P. P., AND BELZER, B. 2009. Crosstalk-aware channel coding schemes for energy efficient and reliable noc interconnects. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems 17,* 11 (November), 1626 – 1639.

GANGULY, A., PANDE, P. P., BELZER, B., AND GRECU, C. 2007. Addressing signal integrity in networks on chip interconnects through crosstalk-aware double error correction coding. In *Proc. IEEE Computer Society Annual Symp. VLSI ISVLSI '07*. 317–324.

GIZOPOULOS, D., PSARAKIS, M., ADVE, S. V., RAMACHANDRAN, P., HARI, S. K. S., SORIN, D., BISWAS, A. M. A., AND VERA, X. 2011. Architectures for online error detection and recovery in multicore processors. In *Proceedings of the Design Automation and Test Europe Conference (DATE)*. Embedded tutorial.

GLASS, C. J. AND NI, L. M. 1993. Fault-tolerant wormhole routing in meshes. In *Proc. Twenty-Third Int Fault-Tolerant Computing FTCS-23. Digest of Papers. Symp*. 240–249.

GRECU, C., IVANOV, A., PANDE, R., JANTSCH, A., SALMINEN, E., OGRAS, U., AND MARCULESCU, R. 2007. Towards open network-on-chip benchmarks. In *Proc. 1st International Symposium on Networks-on-Chip (NOCS)*.

GRECU, C., IVANOV, A., SALEH, R., AND PANDE, P. P. 2006. Noc interconnect yield improvement using crosspoint redundancy. In *Proc. 21st IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems DFT '06*. 457–465.

GRECU, C., IVANOV, A., SALEH, R., SOGOMONYAN, E., AND PANDE, P. P. 2006. On-line fault detection and location for noc interconnects. In *Proceedings of the 12th IEEE International On-Line Testing Symposium (IOLTS'06)*. 6pp.

HAZUCHA, P., KARNIK, T., MAIZ, J., WALSTRA, S., BLOECHEL, B., TSCHANZ, J., DERMER, G., HARELAND, S., ARMSTRONG, P., AND BORKAR, S. 2003. Neutron soft error rate measurements in a 90-nm cmos process and scaling trends in sram from 0.25-mu;m to 90-nm generation. In *Electron Devices Meeting, 2003. IEDM '03 Technical Digest. IEEE International*. 21.5.1 – 21.5.4.

HEGDE, R. AND SHANBHAG, N. 2000. Toward achieving energy efficiency in presence of deep submicron noise. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 8,* 4 (aug), 379 –391.

HERNANDEZ, C., FEDERICO, F., SANTONJA, V., AND DUATO, J. 2009. A new mechanism to deal with process variability in noc links. In *Proc. Int'l Parallel and DIstributed Processing Symposium (PDPS)*. 1–11.

HOSKOTE, Y., VANGAL, S., SINGH, A., BORKAR, N., AND BORKAR, S. 2007. A 5-ghz mesh interconnect for a teraflops processor. *IEEE Micro 27,* 5, 51–61.

HU, J. AND MARCULESCU, R. 2004. Dyad - smart routing for networks-on-chip. In *Proceedings of the 41st Design Automation Conference (DAC)*. 260–263.

HUFFMAN, W. C. AND PLESS, V. 2003. *Fundamentals of Error-Correcting Codes*. Cambridge University Press.

INTEL LABS. 2010. The SCC platform overview. Tech. Rep. Revision 0.7, Intel Corporation. May.

ITRS 2009. International technology roadmap for semiconductors. Tech. rep., ITRS Technology Working Group.

JANTSCH, A., LAUTER, R., AND VITKOWSKI, A. 2005. Power analysis of link level and end-to-end data protection in networks on chip. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'05)*. 1770–1773Vol.2.

JOVANOVIC, S., TANOUGAST, C., WEBER, S., AND BOBDA, C. 2009. A new deadlock-free fault-tolerant routing algorithm for noc interconnections. In *Proc. International Conference on Field Programmable Logic (FPL)*. 326–331.

KAKOEE, M. R., BERTACCO, V., AND BENINI, L. 2011a. A distributed and topology-agnostic approach for on-line noc testing. In *Proceedings of the Network on Chip Symposium*.

KAKOEE, M. R., BERTACCO, V., AND BENINI, L. 2011b. Relinoc: A reliable network for priority-based on-chip communication. In *Proceedings of the Design Automation and test Europe Conference (DATE)*.

KEANE, J. AND KIM, C. 2011. An odometer for CPUs. *IEEE Spectrum 48,* 5 (May), 26–31. Online version appeared as "Transistor Aging".

KEANE, J., KIM, T.-H., AND KIM, C. H. 2007. An on-chip nbti sensor for measuring pmos threshold voltage degradation. In *Proceedings of the International Symposium on Low Power Electronics and Design*.

KIM, J., NICOPOULOS, C., PARK, D., NARAYANAN, V., YOUSIF, M. S., AND DAS, C. R. 2006. A gracefully degrading and energy-efficient modular router architecture for on-chip networks. In *Proc. International Symposium on Computer Architecture (ISCA)*. Boston, MA, USA, 4 – 15.

KIM, J., PARK, D., NICOPOULOS, C., VIJAYKRISHNAN, N., AND DAS, C. 2005. Design and analysis of an noc architecture from performance, reliability and energy perspective. In *Proceedings of the Symposium on Architecture for Networking and Communications Systems (ANCS)*.

KIM, Y. B. AND KIM, Y.-B. 2007. Fault tolerant source routing for network-on-chip. In *Proc. 22nd IEEE Int. Symp. Defect and Fault-Tolerance in VLSI Systems (DFT)*. 12–20.

KOHLER, A. AND RADETZKI, M. 2009. Fault-tolerant architecture and deflection routing for degradable noc switches. In *Proc. 3rd ACM/IEEE International Symposium on Networks-on-Chips (NOCS)*. 22–31.

KOHLER, A., SCHLEY, G., AND RADETZKI, M. 2010. Fault tolerant network on chip switching with graceful performance degradation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 20,* 6, 883–896.

KOIBUCHI, M., MATSUTANI, H., AMANO, H., AND MARK PINKSTON, T. 2008. A lightweight fault-tolerant mechanism for network-on-chip. In *Proc. Second ACM/IEEE Int. Symp. Networks-on-Chip NoCS 2008*. 13–22.

KOUPAEI, F. K., KHADEMZADEH, A., AND JANIDARMIAN, M. 2011. Fault-tolerant application-specific network-on-chip. In *Proceedings of the World Congress on Engineering and Computer Science*.

KUHN, K., KENYON, C., KORNFELD, A., LIU, M., MAHESHWARI, A., KAI SHIH, W., SIVAKUMAR, S., TAYLOR, G., VANDERVOORN, P., AND ZAWADZKI, K. 2008. Managing process variation in Intel's 45nm CMOS technology. *Intel Technology Journal 12,* 2 (June).

LEE, H., CHANG, N., OGRAS, U., AND MARCULESCU, R. 2007. On-chip communication architecture exploration: A quantitative evaluation of point-to-point, bus, and network-on-chip approaches. *ACM T 12,* 3.

LEHTONEN, T., LILJEBERG, P., AND PLOSILA, J. 2007a. Analysis of forward error correction methods for nanoscale networks-on-chip. In *Nano-Net '07: Proceedings of the 2nd international conference on Nano-Networks*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 1–5.

LEHTONEN, T., LILJEBERG, P., AND PLOSILA, J. 2007b. Online reconfigurable self-timed links for fault tolerant noc. *VLSI Design 2007*, 13.

LEHTONEN, T., WOLPERT, D., LILJEBERG, P., PLOSILA, J., AND AMPADU, P. 2010. Self-adaptive system for addressing permanent errors in on-chip interconnects. *IEEE Transactions on VLSI Systems 18,* 4, 527–540.

LIN, S.-Y., SHEN, W.-C., HSU, C.-C., CHAO, C.-H., AND WU, A.-Y. 2009. Fault-tolerant router with built-in self-test/self-diagnosis and fault-isolation circuits for 2d-mesh based chip multiprocessor systems. In *Proc. Int. Symp. VLSI Design, Automation and Test (VLSI-DAT)*. 72–75.

LYSNE, O., PINKSTON, T., AND DUATO, J. 2005. Part ii: A methodology for developing deadlock-free dynamic network reconfiguration processes. *IEEE Transactions on Parallel and Distributed Systems 16,* 5, 428–443.

MAJER, M., BOBDA, C., AHMADINIA, A., AND TEICH, J. 2005. Packet routing in dynamically changing networks on chip. In *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium*. 154b–154b.

MALKIN, G. AND STEENSTRUP, M. 1995. Distance-vector routing. In *Routing in Communication Networks*, M. Steenstrup, Ed. Prentice Hall, 83–98.

MARCULESCU, R., OGRAS, U., PEH, L.-S., JERGER, N., AND HOSKOTE, Y. 2009. Outstanding research problems in noc design: System, microarchitecture, and circuit perspectives. *IEEE Transactions on Computer 28,* 1, 3–21.

MCPHERSON, J. 2006. Reliability challenges for 45nm and beyond. In *Design Automation Conference, 2006 43rd ACM/IEEE*. 176 –181.

MEDIRATTA, S. D. AND DRAPER, J. 2007. Performance evaluation of probe-send fault-tolerant network-on-chip router. In *Proceedings of the IEEE International Conf. on Application-specific Systems, Architectures and Processors (ASAP07)*. 69–75.

MEJIA, A., FLICH, J., DUATO, J., REINEMO, S.-A., AND SKEIE, T. 2006. Segment-based routing: an efficient fault-tolerant routing algorithm for meshes and tori. In *Proc. 20th Int. Parallel and Distributed Processing Symposium (IPDPS)*.

MEJIA, A., PALESI, M., FLICH, J., KUMAR, S., LOPEZ, P., HOLSMARK, R., AND DUATO, J. 2009. Region-based routing: A mechanism to support efficient routing algorithms in nocs. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems 17,* 3, 356–369.

MINTARNO, E., SKAF, J., ZHENG, R., VELAMALA, J. B., CAO, Y., BOYD, S., DUTTON, R. W., AND MITRA, S. 2011. Self-tuning for maximized lifetime energy-efficiency in the presence of circuit aging. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD) 30,* 5 (May), 760 – 773.

MIRANDA, E. AND SUNE, J. 2004. Electron transport through broken down ultra-thin sio2 layers in mos devices. *Microelectronics Reliability 44,* 1, 1 – 23.

MITRA, S., ZHANG, M., WAQAS, S., SEIFERT, N., GILL, B., AND KIM, K. S. 2006. Combinational logic soft error correction. In *Test Conference, 2006. ITC '06. IEEE International*. 1 –9.

MOY, J. 1995. Link-state routing. In *Routing in Communication Networks*, M. Ste, Ed. Prentice Hall, 135–157.

MURALI, S., ATIENZA, D., BENINI, L., AND DE MICHELI, G. 2006. A multi-path routing strategy with guaranteed in-order packet delivery and fault-tolerance for networks on chip. In *Proc. 43rd ACM/IEEE Design Automation Conference (DAC)*. 845–848.

MURALI, S., THEOCHARIDES, T., VIJAYKRISHNAN, N., IRWIN, M., BENINI, L., AND DEMICHELI, G. 2005. Analysis of error recovery schemes for networks on chips. *IEEE Design & Test of Computers 22,* 5, 434–442.

NICOLAIDIS, M. 1999. Time redundancy based soft-error tolerance to rescue nanometer technologies. In *VLSI Test Symposium, 1999. Proceedings. 17th IEEE*. 86 –94.

OGRAS, U., HU, J., AND MARCULESCU, R. 2005. Key research problems in noc design: A holistic perspective. In *Proceedings of the International Conference on Hardware/Sofware Codesign and System Synthesis (CODES+ISSS)*.

OWENS, J., DALLY, W., HO, R., JAYASIMHA, D., KECKLER, S., AND PEH, L.-S. 2007. Research challenges for on-chip interconnection networks. *Micro, IEEE 27,* 5 (September - October), 96 –108.

PALESI, M., KUMAR, S., AND CATANIA, V. 2010. Leveraging partially faulty links usage for enhancing yield and performance in networks-on-chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 29*, 426–440.

PANDE, P. P., GANGULY, A., FEERO, B., BELZER, B., AND GRECU, C. 2006. Design of low power & reliable networks on chip through joint crosstalk avoidance and forward error correction coding. In *Proc. 21st IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems DFT '06*. 466–476.

PARIKH, R. AND BERTACCO, V. 2011. Formally enhanced runtime verification to ensure noc functional correctness. In *Proc. 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. 410–419.

PARK, D., NICOPOULOS, C., KIM, J., VIJAYKRISHNAN, N., AND DAS, C. R. 2006. Exploring fault-tolerant network-on-chip architectures. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN'06)*. IEEE Computer Society, Washington, DC, USA, 93–104.

PATOOGHY, A. AND MIREMADI, S. G. 2008. Ltr: A low-overhead and reliable routing algorithm for network on chips. In *Proc. Int. SoC Design Conf. ISOCC '08*. Vol. 01.

PATOOGHY, A., MIREMADI, S. G., AND SHAFAEI, M. 2010. Crosstalk modeling to predict channel delay in network-on-chips. In *2010 IEEE International Conference on Computer Design (ICCD)*. 396–401.

PIRRETTI, M., LINK, G. M., BROOKS, R. R., VIJAYKRISHNAN, N., KANDEMIR, M. T., AND IRWIN, M. J. 2004. Fault tolerant algorithms for network-on-chip interconnect. In *Proceedings of the International Symposium on VLSI (ISVLSI '04)* (2005-04-14). IEEE Computer Society, 46–51.

PUENTE, V., GREGORIO, J. A., VALLEJO, F., AND BEIVIDE, R. 2008. Immunet: Dependable routing for interconnection networks with arbitrary topology. *IEEE Transactions on Computers 57,* 12, 1676–1689.

RADETZKI, M. 2011. Fault-tolerant differential q routing in arbitrary noc topologies. In *Proc. Embedded and Ubiquitous Computing (EUC)*. 33–40.

RAIK, J., UBAR, R., AND GOVIND, V. 2007. Test configurations for diagnosing faulty links in noc switches. In *Proceedings of the 12th IEEE European Test Symposium (ETS)*. 29–34.

RANTALA, V., LEHTONEN, T., LILJEBERG, P., AND PLOSILA, J. 2009. Multi network interface architectures for fault tolerant network-on-chip. In *Proceedings of International Symposium on Signals, Circuits and Systems*. 1–4.

RAVINDRAN, D. K. 2009. Structural fault-tolerance on the noc circuit level. Tech. rep., Institut für Technische Informatik, Universität Stuttgart. June.

RODRIGO, S., FLICH, J., DUATO, J., AND HUMMEL, M. 2008. Efficient unicast and multicast support for cmps. In *Proc. 41st IEEE/ACM International Symposium on Microarchitecture (MICRO-41)*. 364–375.

RODRIGO, S., FLICH, J., ROCA, A., MEDARDONI, S., BERTOZZI, D., CAMACHO, J., SILLA, F., AND DUATO, J. 2010. Addressing manufacturing challenges with cost-efficient fault tolerant routing. In *Proc. Fourth ACM/IEEE Int Networks-on-Chip (NOCS) Symp*. 25–32.

ROSSI, D., ANGELINI, P., AND METRA, C. 2007. Configurable error control scheme for noc signal integrity. In *Proc. International On-Line Testing Symposium (IOLTS)*. Crete, Greece, 43 – 48.

SAHA, S. 2010. Modeling process variability in scaled cmos technology. *Design Test of Computers, IEEE 27,* 2 (march-april), 8 –16.

Sanyo Semiconductors Visitied in June 2011. Quality and reliability handbook ver 3. Online Website by Sanyo Semiconductor Co., Ltd., http://semicon.sanyo.com/en/reliability/.

SCHROEDER, M. D., BIRRELL, A. D., BURROWS, M., MURRAY, H., NEEDHAM, R. M., RODEHEFFER, T. L., SATTERTHWAITE, E. H., AND THACKER, C. P. 1991. Autonet: a high-speed, self-configuring local area network using point-to-point links. *9,* 8, 1318–1335.

SCHÄFER, M., HOLLSTEIN, T., ZIMMER, H., AND GLESNER, M. 2005. Deadlock-free routing and component placement for irregular mesh-based networks-on-chip. In *Proc. International Conference on Computer Aided Design (ICCAD)*. 238–245.

SCHÖNWALD, T., ZIMMERMANN, J., BRINGMANN, O., AND ROSENSTIEL, W. 2007. Fully adaptive fault-tolerant routing algorithm for network-on-chip architectures. In *Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD)*. 527–534.

SHAMSHIRI, S., GHOFRANI, A., AND CHENG, K.-T. 2011. End-to-end error correction and online diagnosis for on-chip networks. In *International Test Conference*.

SHIVAKUMAR, P., KISTLER, M., KECKLER, S. W., BURGER, D., AND ALVISI, L. 2002. Modeling the effect of technology trends on the soft error rate of combinational logic. In *Proceedings of the International Conference on Dependable Systems and Networks*.

SHOOMAN, M. L. 2002. *Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design*. John Wiley & Sons.

SONG, W., EDWARDS, D., NUNEZ-YANEZ, J., AND DASGUPTA, S. 2009. Adaptive stochastic routing in fault-tolerant on-chip networks. In *3rd ACM/IEEE International Symposium on Networks-on-Chip (NoCS)*. 32–37.

SRIDHARA, S. AND SHANBHAG, N. 2005. Coding for system-on-chip networks: A unified framework. *IEEE Transactions on VLSI Systems 13,* 6, 655–667.

STRANO, A., BERTOZZI, D., TRIVINO, F., SANCHEZ, J. L., ALFARO, F. J., AND FLICH, J. 2012. Osr-lite: Fast and deadlock-free noc reconfiguration framework. In *Proc. International Conference on Embedded Computer Systems: Architectures, Modelling and Simulation*.

TAKEDA, E. AND YANG, C. 1995. *Hot-Carrier Effects in MOS Devices*. Academic Press.

TAMHANKAR, R., MURALI, S., AND DE MICHELI, G. 2005. Performance driven reliable link design for networks on chips. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC)*. Vol. 2. 749–754Vol.2.

VANGAL, S., HOWARD, J., RUHL, G., DIGHE, S., WILSON, H., TSCHANZ, J., FINAN, D., IYER, P., SINGH, A., JACOB, T., JAIN, S., VENKATARAMAN, S., HOSKOTE, Y., AND BORKAR, N. 2007. An 80-tile 1.28tflops network-on-chip in 65nm cmos. In *Digest of Technical Papers of the IEEE International Solid-State Circuits Conference (ISSCC)*. 98–589.

VITERBI, A. J. 1971. Convolutional codes and their performance in communication systems. *IEEE Transactions on Communication Technology 19,* 5, 751–772.

VITKOVSKI, A., JANTSCH, A., LAUTER, R., HAUKILAHTI, R., AND NILSSON, E. 2008. Low-power and error protection coding for network-on-chip traffic. *IET Computers and Digital Techniques 2,* 6, 483–492.

VITKOVSKIY, A., SOTERIOU, V., AND NICOPOULOS, C. 2010. A fine-grained link-level fault-tolerant mechanism for networks-on-chip. In *Proc. IEEE Int Computer Design (ICCD) Conf*. 447–454.

WALKER, M. 2000. Modeling the wiring of deep submicron ics. *Spectrum, IEEE 37,* 3 (mar), 65 –71.

WITTMANN, R., PUCHNER, H., HINH, L., CERIC, H., GEHRING, A., AND SELBERHERR, S. 2005. Simulation of dynamic NBTI degradation for a 90nm CMOS technology. In *Proceedings of the Nanotechnology Conference*.

WU, E., LAI, W., NOWAK, E., MCKENNA, J., VAYSHENKER, A., AND HARMON, D. 2001. Interplay of voltage and temperature acceleration of oxide breakdown for ultra-thin oxides. *Microelectronic Engineering 59*, 25–31.

WU, J. AND WANG, D. 2002. Fault-tolerant and deadlock-free routing in 2-d meshes using rectilinear-monotone polygonal fault blocks. In *Proceedings of the International Conference on Parallel Processing*. 247–254.

XINMING, D. AND XUEMEI, S. 2010. Fault-tolerant routing in a prdt(2,1)-based noc. In *Proc. 2nd Int Computer Engineering and Technology (ICCET) Conf*.

YAGHINI, P. M., EGHBAL, A., PEDRAM, H., AND ZARANDI, H. R. 2011. Investigation of transient fault effects in synchronous and asynchronous network on chip router. *Journal of Systems Architecture 57,* 1, 61 – 68.

YANG, Y. 2010. Issues of ESD protection in nano-scale cmso. Ph.D. thesis, George Mason University, Fairfax, Virginia, USA.

YU, A. J. AND LEMIEUX, G. G. 2005. Fpga defect tolerance: Impact of granularity. In *FPT*. 189–196.

YU, Q. AND AMPADU, P. 2008. Adaptive error control for noc switch-to-switch links in a variable noise environment. In *Defect and Fault Tolerance of VLSI Systems, 2008. DFTVS '08. IEEE International Symposium on*. 352–360.

YU, Q. AND AMPADU, P. 2010. Transient and permanent error co-management method for reliable networks-on-chip. In *Proc. Fourth ACM/IEEE Int Networks-on-Chip (NOCS) Symp*. 145–154.

YU, Q. AND AMPADU, P. 2011. A dual-layer method for transient and permanent error co-management in noc links. *IEEE Transactions on Circuits and Systems II: Express Briefs 58,* 1, 36–40.

YU, Q. AND AMPADU, P. 2012. Dual-layer adaptive error control for network-on-chip links. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems 20,* 7, 1304–1317.

YU, Q., CANO, J., FLICH, J., AND AMPADU, P. 2012. Transient and permanent error control for high-end multiprocessor systems-on-chip. In *Proc. Sixth IEEE/ACM International Symposium on Networks on Chip (NoCS)*. 169–176.

YU, Q., ZHANG, B., LI, Y., AND AMPADU, P. 2010. Error control integration scheme for reliable noc. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*. 3893 –3896.

YU, Q., ZHANG, M., AND AMPADU, P. 2011. Exploiting inherent information redundancy to manage transient errors in noc routing arbitration. In *Proceedings of the IEEE Network on Chip Symposium (NoCS)*.

ZHANG, B. AND ORSHANSKY, M. 2008. Modeling of nbti-induced pmos degradation under arbitrary dynamic temperature variation. In *Quality Electronic Design, 2008. ISQED 2008. 9th International Symposium on*. 774 –779.

ZHANG, M. AND SHANBHAG, N. 2006. Soft-error-rate-analysis (sera) methodology. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 25,* 10 (oct.), 2140 –2155.

ZHANG, Y. AND JIANG, J. 2008. Bibliographical review on reconfigurable fault-tolerant control systems. *Annual reviews in Control 32*, 229–252.

ZHANG, Y., LI, H., AND LI, X. 2009. Selected crosstalk avoidance code for reliable network-on-chip. *Journal of Computer Science and Technology 24,* 6, 1074–1085.

ZHANG, Y., PARIKH, D., SANKARANARAYANAN, K., SKADRON, K., AND STAN, M. 2003. Hotleakage: A temperature-aware model of subthreshold and gate leakage for architects. Tech. Rep. CS-2003-05, University of Virgiania, Department of Computer Science. March.

ZHANG, Z., GREINER, A., AND TAKTAK, S. 2008. A reconfigurable routing algorithm for a fault-tolerant 2d-mesh network-on-chip. In *Proceesings of the Design Automation Conference (DAC)*. 441–446.

ZIMMER, H. AND JANTSCH, A. 2003. A fault model notation and error-control scheme for switch-to-switch buses in a network-on-chip. In *Hardware/Software Codesign and System Synthesis, 2003. First IEEE/ACM/IFIP International Conference on*. 188–193.

Paper received ...