

A Fault Model Notation and Error-Control Scheme for Switch-to-Switch Buses in a Network-on-Chip

Heiko Zimmer
Darmstadt University of Technology,
Darmstadt, Germany
heiko.zimmer@ieee.org

Axel Jantsch
Royal Institute of Technology (KTH),
Stockholm, Sweden
axel@imit.kth.se

ABSTRACT

The reliability of a Network-on-Chip will be significantly influenced by the reliability of the switch-to-switch connections. Faults on these buses may cause disturbances on multiple adjacent wires, so that errors on these wires can no longer be considered as statistically independent from one another, as it is expected due to deep submicron effects. A new fault model notation for buses is proposed which can represent multiple-wire, multiple-cycle faults. An estimation method based on this notation is presented which can accurately predict error probabilities. This method is used to examine bus encoding schemes. Finally, an encoding scheme for four Quality-of-Service classes is proposed which can be dynamically selected for each packet.

Categories and Subject Descriptors

B.4.3 [Input/Output and Data Communications]: Interconnections (Subsystems); B.8.1 [Performance and Reliability]: Reliability, Testing, and Fault-Tolerance

General Terms

Design, Reliability

Keywords

Network-on-Chip, Fault Tolerance, Bus Encoding

1. INTRODUCTION

To efficiently use the billions of transistors that will soon be available on a single chip, new design methods will be necessary. While the reuse of components in building Systems-on-Chip (SoC) will continue, their interconnection becomes a major concern. Currently used shared buses and dedicated wires exhibit limited scalability.

A new approach to overcome these limitations is to implement a *Network-on-Chip* (NoC) to handle the on-chip

communication [2, 7, 8]. In such an architecture, communication can be decoupled from computation: The computational blocks called *resources* (which can be anything from processors or DSPs over FPGA, mixed signal or ASIC blocks to any kind of memory) are connected to an inter-resource communication network.

We assume a NoC-architecture that employs packet switching in a $n \times m$ mesh of switches. Adjacent switches are connected by wide buses so that a complete packet can be transmitted in parallel. By a *network interface*, one resource is connected to every switch. These switches must be as small and efficient as possible [10] in order to limit the overhead (in terms of area, power and delay) introduced by the NoC.

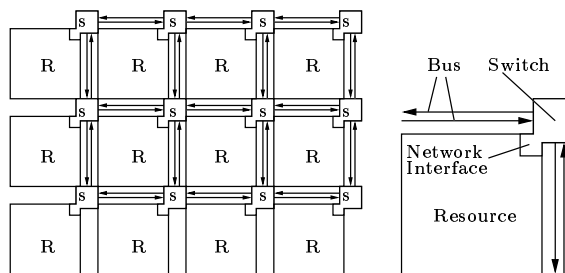


Figure 1: Complete NoC and detailed view of one resource/switch block

QoS Levels

The Network-on-Chip becomes the communication backbone to handle all data transmitted between components of a SoC. Due to the varying requirements of communication streams, it must implement transmission at multiple Quality-of-Service (QoS) levels (classified by parameters like bandwidth, reliability, latency etc.) [7]. The transport of multimedia data may require maximum bandwidth whereas data integrity must be guaranteed for transfers from and to memories. We propose four QoS levels offering different characteristics to applications:

Maximum Bandwidth: Since the amount of wires available for routing the inter-switch buses is limited, the redundancy introduced by bus encoding will directly influence the application payload that can be transported with every packet. Consequently, maximum bandwidth is available to the application when no encoding is applied.

Guaranteed Integrity: To ensure data integrity as far as possible, error detection methods can be used. Data correc-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CODES+ISSS'03, October 1-3, 2003, Newport Beach, California, USA.
Copyright 2003 ACM 1-58113-742-7/03/0010 ...\$5.00.

tion is not attempted since it might result in miscorrection. If erroneous data is detected, a flag can be set, a retransmission can be initiated or the whole packet can be dropped.

Minimum Latency: To achieve minimum latency, packets are always forwarded. Error correction is performed, the underlying assumption is that all errors can be corrected by the code used. This mode is well suited for applications that can tolerate rare errors (due to miscorrection) but depend on receiving data at a constant rate.

High Reliability: Using codes capable of correcting errors and detecting uncorrectable errors at the same time combines the characteristics of the two previously described modes at the expense of lower bandwidth and higher latency.

Faults on DSM buses

During the next decade, deep submicron (DSM) technology will have to cope with more transient faults than ever before. The number of single-event-upsets (SEU) due to various reasons (charge injection by neutrons or α -particles, process variations, electro-magnetic interference) will increase along with crosstalk, mainly affecting long interconnects [1]. Technology scaling also gives rise to new error sources: For instance can the effects of neutrons (which have formerly only been a concern with memories) now cause upsets in logic elements and low power/low capacitance buses as well [9]. However, the main source for errors on buses will be crosstalk, which can lead to increased delay or cause voltage glitches on wires [4]. Furthermore, crosstalk can inherently cause multi-bit and bidirectional errors [6] (i.e., disturbing multiple wires, causing both $0 \rightarrow 1$ and $1 \rightarrow 0$ errors).

Since faults on the massively parallel buses between adjacent switches will mainly be transient and not necessarily confined to one wire, conventional fault models are not sufficient to accurately describe them. We have developed a new fault model notation for faults on on-chip buses. It is capable of describing a wide range of faults and as a significant feature, it supports modeling of faults that affect multiple wires and last multiple bus cycles. Furthermore, arbitrary effects of faults can be represented.

We show that accurate prediction of error probabilities can be achieved by an estimation method based on this notation which includes structural information (effect, duration and affected wires) about the faults. Evaluating various bus encoding schemes, we derive some general design guidelines to achieve higher reliability of transmissions over on-chip buses in the presence of multiple-wire faults. Eventually, a case study combines all aspects of this work and shows how to select encoding schemes for a fault-tolerant NoC supporting network traffic with non-uniform QoS requirements.

System-wide energy management is not considered. When a NoC is operated in different power-saving modes, we expect different fault scenarios. Our approach is compatible and complementary to power-saving techniques such as [12].

2. RELATED WORK

The importance of crosstalk faults for bus reliability has lead to the development of a crosstalk fault model which can be used to generate test vectors [4]. To evaluate the dependability of interconnects, a hardware-software co-design methodology has been used to compare bus encodings [9]. Also, the energy efficiency of transmission schemes was compared [3] and a combination of error detection with retransmission was found to be most energy efficient. This was extended to

an adaptive low-power transmission scheme for NoCs [12].

All these approaches lack a comprehensive fault model notation that is complete or can at least represent all expected fault types. Such a model must explicitly support faults spanning multiple wires and it must not be limited to one fault type. In the presence of crosstalk effects, statistical independence of errors on adjacent bus lines cannot be assumed. However, the effect of this on bus encoding schemes has not been discussed previously.

The main contribution of this paper is a high-level fault model notation for buses which can represent multiple-wire, multiple-cycle faults on single- or multi-layer on-chip bus architectures. Furthermore, we present an approach to adaptive encoding of on-chip buses with QoS-support.

3. MODELING FAULTS ON NOC-INTERCONNECTS

On a bus, arbitrary faults may occur. Of these, all faults that are caused by the same physical effect belong to one *fault type* f_i . For instance, crosstalk faults and faults caused by neutrons form two fault types. Faults of different types are statistically independent.

3.1 Fault Model Notation

The *fault model* $\mathcal{FM}(f_i)$ describes faults of type f_i in a given bus architecture. It is based on the faults' probability of occurrence (α_i), their characteristics (expressed in the matrix \mathbf{P}_i) and a distance metric DM_i influenced by the physical bus layout.

$$\mathcal{FM}: f_i \rightarrow (\alpha_i, \mathbf{P}_i, DM_i)$$

Probability of Occurrence

We define α_i as the probability of occurrence of a fault of type f_i per wire and cycle. Thus, $\alpha_i = \frac{1}{128}$ leads to one fault of type f_i occurring on a 128 bit wide bus every cycle or every 8 cycles on a 16 bit bus.

Fault Characteristics

When a fault of type f_i occurs, it has a probability $p_i(w, d, e)$ to affect w wires for a duration of d time units (typically clock or transfer cycles) with the effect e . The *effect* of a fault is selected from a list of all possible effects. See table 1 for an incomplete list including symbolic names.

Effect e	Description
1 Inv	logic value on wires inverted
2 Set0	wires forced to logic 0
3 Set1	wires forced to logic 1
4 SetRand	wires forced to random logic value
5 Bridge	wire w_i forced to value of wire w_{i-1}
6 De1	delay: wire set to previous value
7

Table 1: (Numerical) representation of faults' effects

All possible combinations of $p_i(w, d, e)$ are represented in the normalized matrix \mathbf{P}_i of the dimensions $w_{max_i} \times (d_{max_i} + 1) \times e_{max}$. w_{max_i} and d_{max_i} are the maximum values for the number of wires affected and the fault's duration respectively. These maximum values can be different for every fault type f_i . While d_{max_i} can take any positive value, w_{max_i} must not exceed the bus width w_{max} of the

interconnects between the switches: $w_{max_i} \leq w_{max} \cdot e_{max}$ is the number of different effects which can be described.

E.g., when setting $e_{max} = 1$ and thus restricting the model to the effect **Inv**, the normalized matrix \mathbf{P}_i is written as

$$\mathbf{P}_i = \begin{pmatrix} p_i(1, 0, \text{Inv}) & \cdots & p_i(1, d_{max_i}, \text{Inv}) \\ p_i(2, 0, \text{Inv}) & \cdots & p_i(2, d_{max_i}, \text{Inv}) \\ \vdots & \ddots & \vdots \\ p_i(w_{max_i}, 0, \text{Inv}) & \cdots & p_i(w_{max_i}, d_{max_i}, \text{Inv}) \end{pmatrix} \begin{array}{l} \uparrow \\ \text{no. of wires} \\ \downarrow \end{array}$$

$\xrightarrow{\text{fault duration}}$

A fault generally lasts at least one clock cycle ($d \geq 1$). The elements of \mathbf{P}_i with the index $d = 0$ are used to indicate the probability of permanent faults.

Since the matrix elements describe the probabilities of different characteristics of one fault occurrence, their values must satisfy the following condition:

$$\sum_{a=1}^{w_{max_i}} \sum_{b=0}^{d_{max_i}} \sum_{c=1}^{e_{max}} p_i(a, b, c) = 1$$

The manifestation of a fault of type f_i affecting w wires for d cycles with effect e is determined by the probability of occurrence and the normalized value $p_i(w, d, e)$ to a probability of $\alpha_i \cdot p_i(w, d, e)$.

EXAMPLE 1 (FAULT TYPE f_1). *The characteristics of a single-event-upset that is always confined to one wire, lasts only one cycle and has the effect **Inv** are represented by matrix element $p_1(1, 1, \text{Inv}) = 1$. Since for this fault type $d_{max_1} = w_{max_1} = 1$, the resulting matrix is*

$$\mathbf{P}_1 = (p_1(1, 0, \text{Inv}) \quad p_1(1, 1, \text{Inv})) = (0 \quad 1)$$

meaning that if a fault of type f_1 occurs, it will always have the characteristics $w = 1$, $d = 1$ and $e = \text{Inv}$. The probability that a fault of this type occurs is determined by its occurrence probability α_1 .

EXAMPLE 2 (FAULT TYPE f_2). \mathbf{P}_2 characterizes a fault that can affect multiple wires ($e = \text{Inv}$) for multiple cycles.

$$\mathbf{P}_2 = \begin{pmatrix} 0 & 0.65 & 0.1 \\ 0 & 0.2 & 0.05 \end{pmatrix}$$

meaning that a fault of type f_2 will be confined to one wire and one clock (or transfer) cycle in 65% of all occurrences. Another 20% of these faults will disturb two wires for one clock cycle. Only in 15% can the effects be noted for two cycles (on one (10%) or two wires (5%) respectively).

Note that in both examples the elements of the first column are zero, indicating that faults of type f_1 or f_2 will never lead to permanent errors.

EXAMPLE 3 (PERMANENT FAULT). *A fault that causes a permanent defect on one wire whenever it occurs, can be described by a \mathbf{P} -matrix whose only non-zero elements are:*

$$p_{\text{permanent}}(1, 0, \text{Set0}) = 0.5 \quad p_{\text{permanent}}(1, 0, \text{Set1}) = 0.5$$

meaning that faults of this type will cause a permanent error on one wire; stuck-at-0 and stuck-at-1 are equally probable.

Distance Metric

One parameter of the fault characteristics is the number of

wires that are affected by a fault. This does not include the information which wires these are. Without loss of generality, we say that a fault affects w adjacent wires and describe the relative distance between wires in a *distance metric* DM_i . This distance metric may be different for every fault type and bus layout. We use a weighted, directed acyclic graph (DAG), examples are given in figure 2.

The w wires affected by a fault will be those with the lowest distance from the wire where the fault occurs. If multiple wires have the same distance, the choice among them is random.

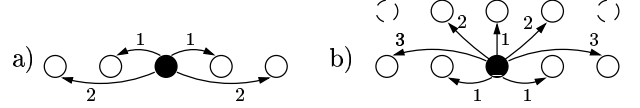


Figure 2: Example for distance relation between wires of a bus running on one or two layers

3.2 Fault Scenario

For a complete representation of all faults that occur in a given environment, a set of fault models is necessary. This set is referred to as a *fault scenario* and is defined as:

$$\mathcal{S} = \{ \mathcal{FM}(f_1), \mathcal{FM}(f_2), \dots, \mathcal{FM}(f_n) \}$$

When using a *fault scenario* in simulation, at every time step it will be checked for each wire if one or multiple faults have occurred (based on the probability of occurrence). If a fault of type f_i has occurred, \mathbf{P}_i and DM_i are evaluated to determine the fault's duration, effect and the affected wires.

Used Fault Scenario

The fault scenario used throughout this paper is a set of three fault models:

$$\mathcal{S} = \{ \mathcal{FM}(f_1), \mathcal{FM}(f_2), \mathcal{FM}(f_3) \}$$

$$= \{ (\alpha_1, \mathbf{P}_1, DM_1), (\alpha_2, \mathbf{P}_2, DM_2), (\alpha_3, \mathbf{P}_3, DM_3) \}$$

All three fault models were assigned the same probability of occurrence $\alpha_1 = \alpha_2 = \alpha_3 = \alpha = 10^{-9}$ and a distance metric equivalent to that from figure 2a was used. While the first two fault models are those from examples 1 and 2, the third one was extracted from a more detailed experiment:

Since the inter-wire capacitance C_I between adjacent wires of a bus has a significant influence on the total capacity which has to be charged during a bus transition, the signal delay depends strongly on the bus transition pattern.

Type	Transitions	Type	Transitions
4C	$v_1 \overline{v_1} v_1 \rightarrow \overline{v_1} v_1 \overline{v_1}$	2C	$v_1 v_2 v_3 \rightarrow v_1 \overline{v_2} v_3$
3C	$v_1 \overline{v_1} v_2 \rightarrow \overline{v_1} v_1 v_2$	1C	$v_1 v_2 v_2 \rightarrow v_1 \overline{v_2} v_2$
	$v_2 v_1 \overline{v_1} \rightarrow v_2 \overline{v_1} v_1$		$v_2 v_2 v_1 \rightarrow \overline{v_2} v_2 v_1$

Table 2: Classification of crosstalk sequences

Considering three adjacent wires, transition patterns can be classified in 1C...4C sequences (table 2) depending on how much they slow down the transition of the middle signal [5]. Out of a large number of random data patterns, we identified all 4C sequences (causing maximum signal delay)

and marked the middle wire as fault location. Translating this into our fault model using $e = \text{Inv}$, we got \mathbf{P}_3 .

Due to the lack of a realistic fault scenario for future DSM technology, all further discussion and results are based on the described fault scenario. I.e., transmission over a planar bus in presence of the three fault types is considered.

$$\mathbf{P}_3 = \begin{pmatrix} 0 & 0.6768 & 0.0836 & 0.0102 & 0.0013 \\ 0 & 0.1521 & 0.0195 & 0.0028 & 0.0003 \\ 0 & 0.0353 & 0.0047 & 0.0007 & 0 \\ 0 & 0.0090 & 0.0009 & 0.0002 & 0 \\ 0 & 0.0019 & 0.0002 & 0 & 0 \\ 0 & 0.0003 & 0.0001 & 0 & 0 \\ 0 & 0.0001 & 0 & 0 & 0 \end{pmatrix}$$

4. ERROR-CONTROL CODING FOR NOC COMMUNICATION BUSES

In this section, possibilities are explored how error-control coding can be used on the switch-to-switch buses of a NoC to implement a network supporting different QoS levels.

Knowing that errors on adjacent wires of these buses are not always statistically independent, coding techniques to enhance the bus reliability are discussed. In a NoC running with gigahertz clock frequency, consisting of 100 switches and about 20.000 switch-to-switch wires, we have to take faults with a probability of 10^{-20} (one every 10^7 sec.) into account to assess reliability over four months. Since simulation cannot be used and to efficiently compare different coding schemes, a method for the estimation of error probabilities is presented. This method is finally used to compare coding schemes providing different QoS-characteristics.

4.1 Coding Techniques for NoC-Buses

A code protecting data on the switch-to-switch buses must allow for fast decoding because decoding has to be completed before the switch can make a routing decision. Additionally, area constraints motivate a switch design with as little buffers as possible. We assume a NoC architecture that requires very low area overhead and high performance from the switches. Thus, we consider only codes that can process 100-200 bit wide buses in a single clock cycle and that can be implemented in a few gates of hardware for each bit. The requirement of fast decoding can be fulfilled by combinatorial logic circuits of low logic depth. Therefore, parity-based codes (e.g., Hamming or Hsiao codes [11]) are considered. These can be employed at various coding schemes: A single-error correcting (SEC) code can correct all single errors. All other errors lead to miscorrection. A double-error detecting (DED) code, on the other hand, detects all single and double errors. In fact, a DED code detects even more errors: If there are 2^k valid codewords of length n , $2^n - 2^k$ error patterns are detected. The properties of these two operating modes are combined in SEC-DED codes which require one more bit of redundancy to encode the same amount of information. Using codes with the ability to detect/correct more than two random errors usually makes decoding slower because arithmetic decoding is necessary [11].

Therefore, we focus on SEC, DED and SEC-DED codes and propose methods that help to enhance the capabilities of these codes in the presence of multiple-bit errors.

One possibility is to divide the network packet into several smaller blocks encoded separately. This *parallel coding*

increases the redundancy, but enhances the overall error detection/correction capabilities and reduces the length of the critical path in the decoder. This approach also enables using different codes for different parts of the packet.

If the data is split into multiple blocks encoded separately, the n bits of a data block can be assigned to n adjacent wires of the bus or they can be interleaved, so that they will be assigned to the wires $x, x + \delta, x + 2\delta, \dots, x + (n - 1)\delta$. The distance δ between two wires that belong to the same block is called *interleaving degree*.

To compare the capabilities of different coding schemes, we use the probability of uncorrected and undetected errors in a packet, $P_{err,UC}$ and $P_{err,UD}$ respectively.

It can be seen from figure 5 that in the presence of multi-bit errors, interleaving can lead to the reduction of those probabilities by several orders of magnitude. At small interleaving degrees, multi-wire faults may easily affect multiple bits of one block. With increasing interleaving degree, the probability that a fault affects multiple wires of one block decreases. In the fault scenario used, faults can affect up to 7 adjacent wires. This is why in figure 5, $P_{err,UC}$ and $P_{err,UD}$ constantly decrease with increasing interleaving degree. For interleaving degrees greater than 7, no further improvement can be achieved.

4.2 Estimation of Error Probabilities

To assess coding schemes, simulation is not feasible for faults with occurrence probabilities as low as 10^{-20} .

We have therefore developed an estimation method to predict error probabilities which is based on the proposed fault model notation. Thereby, errors spanning multiple adjacent wires are taken into account. In order to be able to assess block codes, we compute error probabilities for each block. For instance, a single error in a block of 4 wires of a bus occurs if exactly one wire is erroneous. Single errors are caused by faults affecting only one of the wires within the block or if a wire at the block's border is affected by multiple-bit errors. Thus, the probability that a single error is caused by an erroneous value of a border wire is higher compared to a wire in the middle of the block. Figure 3 shows errors due to two different faults, both leading to a single error within the considered block.

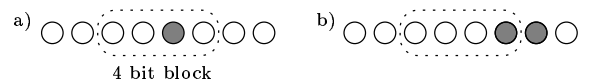


Figure 3: Faults affecting one and two adjacent wires, both leading to single errors in the 4 bit block

Figure 4 shows the probability of single and double errors when transferring data words of different width over a bus on adjacent wires or with interleaving. In these figures, estimations derived by the method proposed below are compared with simulation results and a second estimation based only on the fault occurrence probability. The simulation results were generated using a SystemC model for a bus which introduced errors according to the fault scenario from section 3.2. While results for a fault occurrence probability $\alpha = 10^{-4}$ are shown, other magnitudes of α yielded comparable accuracy for both interleaved and adjacent alignment.

The algorithm to estimate the probability of x -bit errors for blocks of n bits with an interleaving degree of δ is:

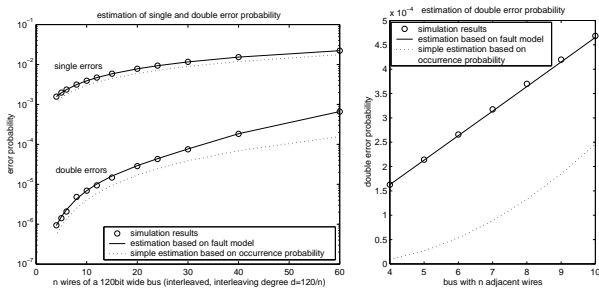


Figure 4: Estimation and simulation results

1. Compute the average probability that a wire is not affected by any fault: P_{ok}
2. Compute the probabilities P_{b_1}, \dots, P_{b_x} that $1 \dots x$ adjacent bits on the border of the considered block are erroneous due to one fault. Combinations of different faults leading to multiple erroneous bits are not counted.
3. In a similar way, compute the probabilities that $1 \dots x$ adjacent bits in the middle of the considered block are erroneous. (P_{m_1}, \dots, P_{m_x})
4. From all possible error combinations leading to a x -bit error in the considered block (e.g., x single errors or one double error and $x - 1$ single errors), the overall probability of x -bit errors is computed using P_{ok} , P_{b_1}, \dots, P_{b_x} and P_{m_1}, \dots, P_{m_x} .

Using this approach, the probability P_{de} of double errors in a block of n bits is for instance computed as

$$P_{de} = P_{ok}^{(n-2)} \cdot (2 \cdot P_{b_2} + P_{b_1}^2 + 2 \cdot P_{b_1} \cdot P_{m_1} \cdot (n-2) + (n-2-1) \cdot P_{m_2} + P_{m_1}^2 \cdot \binom{n-2}{2})$$

The influence of the interleaving degree δ is contained in the values computed in the steps 1.-3.

The current implementation accurately predicts error probabilities for faults on a planar bus whose effect is to invert the logic value. It seems feasible to extend it to incorporate faults with different effects and other bus layouts.

4.3 Multiple QoS-levels in a NoC

The NoC must provide reliable communication services to resources. To guarantee packet delivery, the header (including destination address) of a packet must be particularly strong protected. On the other hand the protection level of the payload should be tunable by the application.

Given a fixed number of physical wires, we propose to select the encoding scheme for the header first, so that the minimum reliability constraint is met. The number of wires required for header encoding limits those remaining for payload transmission. Payload encoding schemes for the desired service levels have to be selected so that the remaining wires are used efficiently. The selection is guided by trading off reliability versus bandwidth. We show this approach in the following example where we assume that 128 wires are available and that a 16 bits wide header is used.

Header Protection

Since the header protection should correct as many errors

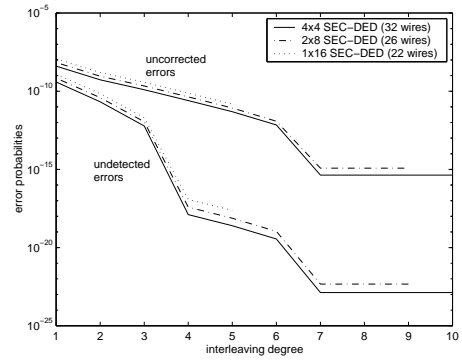


Figure 5: Reduction of error probabilities by interleaving blocks on a 128 bit bus

as possible while at the same time leaving as little errors as possible undetected, the obvious choice is a SEC-DED code.

The header (16 bits) can be encoded as one block (1×16 SEC-DED), or as 2 blocks of 8 bits each (2×8 SEC-DED) or as 4 blocks (4×4 SEC-DED). The notation $b \times k$ means b blocks with k useful information bits each.

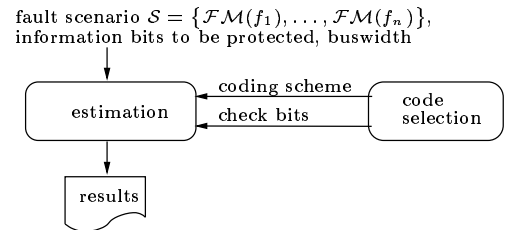


Figure 6: Code quality assessment

The estimation method we presented above was used to compare the properties of different encoding schemes. Figure 6 illustrates how this method was used to derive table 3 which shows the amount of wires required for the dif-

Code	Wires	Max. interl.	Crit. path	$P_{err,UC}$	$P_{err,UD}$
1x16	22	5	9	$1.47 \cdot 10^{-11}$	$2.37 \cdot 10^{-18}$
2x8	26	9	8	$1.21 \cdot 10^{-15}$	$4.63 \cdot 10^{-23}$
4x4	32	16	6	$4.33 \cdot 10^{-16}$	$1.32 \cdot 10^{-23}$

Table 3: Comparison of header protection with different SEC-DED codes

ferent encoding schemes along with information about the expected decoder logic depth and the probabilities of uncorrected and undetected errors ($P_{err,UC}$ and $P_{err,UD}$). As a comparison: When transmitting 16 bits over adjacent wires without encoding, the error probability is $5.33 \cdot 10^{-8}$.

Based on the information from table 3 and complementing experiments, the 2×8 SEC-DED code was chosen as header encoding scheme for this example. It is a reasonable trade-off between error probability and decoder speed, leaving 102 of the 128 wires to transmit the payload.

Payload Protection

Regarding the services discussed at the beginning of this

paper, the *Maximum Bandwidth (MB)* mode is inherent in every implementation. In this example, the bandwidth of this mode is 102 bits/packet and the probability of an erroneous transfer is $3.36 \cdot 10^{-7}$.

Both the *Guaranteed Integrity (GI)* and *Minimum Latency (ML)* modes can be implemented with one code, using the property that every SEC code can also work as DED code: The same codewords are used for transmission, the encoder is identical and the DED code just uses the first part of the decoder block. Thus, these two services offer the same bandwidth to the application, but different protection characteristics. The possibility of this efficient implementation has influenced the selection of examined codes.

The selection of an appropriate coding scheme is guided by reliability and bandwidth constraints. Parameters are the number of parallel encoded blocks and their interleaving degree. This trade-off is shown in table 4. Out of these coding schemes, the appropriate one can be chosen.

Code	Max. interl.	Avail. bandw.	used as SEC $P_{err,UC}$	used as DED $P_{err,UD}$
1x95	1	95	$5.68 \cdot 10^{-8}$	$6.13 \cdot 10^{-9}$
2x45	2	90	$3.77 \cdot 10^{-9}$	$1.71 \cdot 10^{-10}$
3x28	3	84	$5.88 \cdot 10^{-10}$	$3.20 \cdot 10^{-12}$

Table 4: Error probabilities for different operation modes of SEC (or DED) codes, 102 physical wires

Since the *High Reliability (HR)* mode is based on a SEC-DED code, it will require different encoder/decoder logic in parallel to that used for the other modes. Note that SEC-DED codes offer one information bit less per block compared to SEC codes occupying the same amount of wires.

For the 128 bit bus of our example, we have selected the coding schemes from table 5. The selection of services and

Mode	Header protect.	Payload protect.	Payload
MB	2x8 SEC-DED	none	102
GI	2x8 SEC-DED	2x45 DED	90
ML	2x8 SEC-DED	2x45 SEC	90
HR	2x8 SEC-DED	3x27 SEC-DED	81

Table 5: Selected coding schemes

coding schemes a NoC supports is a design-time decision which depends on the intended application and environment, whereas each packet can be encoded with one of these codes during run-time. Since subsequent packets can be encoded differently, the information about the payload encoding scheme is transmitted as part of the header information.

5. CONCLUSIONS

We have presented a fault model notation for faults occurring on on-chip buses that can represent a wide range of faults due to deep submicron effects, among them multiple-wire faults. We showed that error probabilities can be accurately predicted by this model in situations where the statistical independence of errors on bus lines cannot be assumed. This is important because crosstalk faults in future technology generations will most likely affect multiple wires.

Based on simulation and estimation results, we derived design guidelines for bus encoding schemes, namely splitting the data into multiple, separately encoded blocks and

interleaving them. Using these simple methods, the probability of erroneous transfers can be reduced significantly.

The tradeoff between required reliability and the necessary overhead to achieve it was explored for different application requirements, leading to a proposal of a NoC that supports network traffic at four QoS levels. From these, each application can dynamically select the most appropriate one for its communication. An application can even adapt its policy during run-time when too many errors are encountered.

The specific constraints assumed for NoC-switches influenced the selection of considered codes which we believe is a good compromise when the critical path in the decoder should be kept as short as possible. With different requirements or when pipelining is possible, alternatives should be considered.

6. REFERENCES

- [1] International Technology Roadmap for Semiconductors, 2001.
- [2] L. Benini and G. De Micheli. Networks on chips: A new SoC paradigm. *IEEE Computer*, pages 70–78, January 2002.
- [3] D. Bertozzi, L. Benini, and G. De Micheli. Low power error resilient encoding for on-chip data buses. In *Proc. Design, Automation and Test in Europe*, pages 102–109, March 2002.
- [4] M. Cuvillo, S. Dey, X. Bai, and Y. Zhao. Fault modeling and simulation for crosstalk in system-on-chip interconnects. *IEEE/ACM Int. Conf. on Computer-Aided Design*, pages 297–303, 1999.
- [5] C. Duan, A. Tirumala, and S. P. Khatri. Analysis and avoidance of cross-talk in on-chip buses. *Hot Interconnects 9, 2001*, pages 133–138, 2001.
- [6] M. Favalli and C. Metra. Bus crosstalk fault-detection capabilities of error-detecting codes for on-line testing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pages 392–396, September 1999.
- [7] K. Goossens, J. van Meerbergen, A. Peeters, and P. Wielage. Networks on silicon: Combining best-effort and guaranteed services. In *Proc. Design, Automation and Test in Europe*, March 2002.
- [8] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Öberg, K. Tiensyrjä, and A. Hemani. A network on chip architecture and design methodology. In *Proc. of IEEE Computer Society Annual Symposium on VLSI*, April 2000.
- [9] M. Lajolo, M. S. Reorda, and M. Violante. Early evaluation of bus interconnects dependability for system-on-chip designs. *14th Int. Conf. on VLSI Design*, pages 371–376, 2001.
- [10] E. Nilsson, M. Millberg, J. Öberg, and A. Jantsch. Load distribution with the proximity congestion awareness in a network on chip. In *Proc. Design, Automation and Test in Europe*, pages 1126–1127, March 2003.
- [11] T. Rao and E. Fujiwara. *Error-Control Coding for Computer Systems*. Prentice-Hall International, 1989.
- [12] F. Worm, P. Thiran, P. Jenne, and G. De Micheli. An adaptive low-power transmission scheme for on-chip networks. In *Proc. of the 15th International Symposium on System Synthesis*, October 2002.