Configurable Error Control Scheme for NoC Signal Integrity*

Daniele Rossi Paolo Angelini Cecilia Metra
D.E.I.S. University of Bologna
Viale Risorgimento 2, 40136 Bologna, Italy
{drossi, cmetra}@deis.unibo.it paolo.angelini@studio.unibo.it

Abstract

In this paper we propose a novel error control scheme to cope with errors affecting the communication links of a NoC. Our scheme can be configured in Correction Mode, Detection Mode, and Mixed Mode, depending on the particular application, thus allowing to meet different Quality of Service (QoS) levels in terms of error control. For each configuration mode, we propose different error control policies and we consider SEC Hamming codes, SEC/DED Hsiao codes, and Symbol Error Correcting codes. We evaluate advantages and drawbacks of each approach, in terms of signal integrity, area overhead and impact on performance.

1. Introduction

As device geometry shrinks toward the nanometer scale, interconnects, both on-chip and off-chip, are becoming a critical bottleneck in meeting performance and power consumption requirements of chip/system design [1]. Recently, researchers have proposed packet based communication networks, known as Network on Chips (NoCs), to overcome this problem and, more generally, to address the challenges of increasing interconnects' complexity [2].

Transmission of information through wires is becoming unreliable, due to several noise sources. Among them, crosstalk is commonly recognized as the main contributor to the noise affecting the on-chip interconnects. Error control (detection or correction) coding mechanisms can protect the system from errors affecting the subsystems' communication [3 - 9].

Error control schemes developed for NoC communication systems can use end-to-end flow control (network level), or switch-to-switch flow control (link level). Both error detection followed by retransmission, and error correction have been proposed to cope with communication errors [5, 8, 10, 11]. Mixed schemes with combined error detection plus retransmission and error correction are also possible [8]. Because of the different error detection/correction capability, area-power overhead, and impact on performance of the various schemes, the choice of the error control scheme requires exploring different cost-performance-reliability tradeoffs.

In this paper, we propose a novel error control scheme which, depending on the particular application, can be configured by the user in three different operating modes: *Correction Mode*, *Detection Mode*, and *Mixed Mode*. This allows the NoC to meet different Quality of Service (QoS) levels in terms of error control. As an example, the proposed error control scheme has been applied to the Spidergon NoC [12], but its application can be easily extended to different NoC topologies.

For each configuration mode, we propose different error control policies (varying from the end-to-end to the switch-to-switch control policy) considering SEC Hamming codes, SEC/DED Hsiao codes, and Symbol Error Correcting codes [13]. Since these codes can be encoded/decoded by means of parallel networks, they allow minimizing the impact on performance of the encoding/decoding circuit. We evaluate the advantages and drawbacks of each approach, in terms of signal integrity, required hardware modifications, area over-head and impact on performance

The remainder of this paper is organized as follows. In Sect. 2 we give some preliminaries on NoCs. In Sect. 3 we present the proposed error control scheme, describing the different error control modes and error control policies. In Sect. 4 we introduce the derived router architectures. In Sect. 5 we show the implementation of the encoding/decoding blocks employed by our proposed scheme. In Sect. 6 we estimate the cost of our proposed scheme implementing Hamming, Hsiao and Symbol error control codes. Finally, in Sect. 7 we give some conclusive remarks.

2. Preliminaries on NoC

In NoC-based systems, the *Intellectual Property* (IP) cores of the network communicate with one another through high performance links and intelligent switches [2]. Packet-based communication is becoming

* Partially supported by AST-STMicroelectronics, Grenoble (France).

13th IEEE International On-Line Testing Symposium (IOLTS 2007)
0-7695-2918-6/07 $25.00 © 2007 IEEE

a standard for NoC systems, and recent trends show that wormhole switching is the solution of choice for NoCs. In packet-based NoC, each packet is split into data units called flits. In wormhole, the first flit of a packet (header) is sent to the routing mechanism (switch stage), and then transferred to the output queue of the target channel. Once the header flit has been processed by the router, a switching mechanism is defined to forward immediately all following packet flits to the buffer of the outgoing links of the target path to the destination node.

Fig. 1 shows the structure of the routers of the considered NoC. Each router has 4 channels: 3 channels communicate with the adjacent routers (from the right (R), left (L), and across (A) directions), while one channel communicates with the network interface (NI), which connects the respective IP with the NoC.

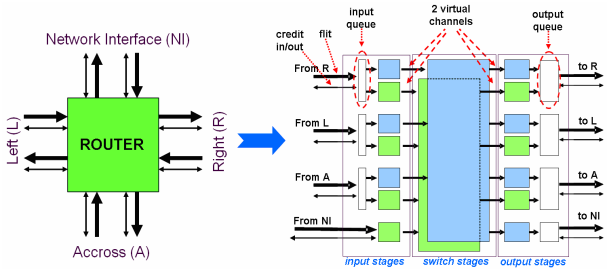

Fig. 1. Spidergon router structure [12].

3. Proposed Scheme: Configuration Modes and Error Control Policies

In NoC architectures, the error control mechanism can be distributed over multiple hops (switch-to-switch control policy), or concentrated at the end nodes (end-to-end control policy). The trade-off related to the localization of error detection and correction involves several figures of merit, such as latency, area, and power consumption. Furthermore, due to the different requirements of applications, the NoC must implement transmission at multiple QoS levels [8], with possibly different error protection requirements. Furthermore, it should be taken into account that different parts of the transmitted packet can require different levels of protection. In particular the header, carrying information about the destination, can require a higher protection than the payload**.**

Based on these considerations, we propose a novel error control coding scheme that can be configured by designers in *Correction Mode*, *Detection Mode*, and *Mixed Mode*. Depending on the particular application, it can meet different QoS levels in terms of error control. When the proposed scheme is configured in the *Correction Mode*, it fully exploits the correction ability of an implemented error correcting code that, allowing correcting possible errors, guarantees that corrected packets/flits are always forwarded, thus incurring minimum latency.

In the *Detection Mode*, the decoding section devoted to error correction is bypassed, and possibly disabled. If erroneous data are detected, a flag can be set, and then a retransmission of the erroneous data can be requested, or the whole packet/flit can be dropped. This configuration mode 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 only detected.

Error control coding mechanisms can use switch-to-switch (s-s) flow control, or end-to-end (e-e) flow control. The s-s approach can offer higher reliability, since the transmitted data are checked (and possibly corrected) at every hop in the path between the sender and the receiver IPs. As for the e-e approach, it certainly poses fewer constraints on decoding time, but can provide far less error protection compared to the s-s approach, since the error checking and possible correction is performed only when the flits reach the receiver. In this case, multiple errors could exceed the code error protection ability and lead to miscorrection.

Therefore, the following five error control policies can be identified and adopted to provide NoC architectures with data integrity: I. Header: correction s-s, Payload: correction s-s; II. Header: correction s-s, Payload: detection (+ retransmission) s-s; III. Header: correction s-s, Payload: detection (+ retransmission) e-e; IV. Header: detection (+ retransmission) s-s, Payload: detection (+ retransmission) s-s; V. Header: detection (+ retransmission) s-s, Payload: detection (+ retransmission) e-e.

4. Proposed Scheme: Router Architecture and Costs

In this section, we prescind from the particular error control coding technique applied, considering the general case of a generic (n, k) error control code, whose codewords are composed by k information bits, and $m = n - k$ check bits. Details about the possible implementation of different error control codes will be given in Sect. 5. Let us discuss how the router architecture can be modified, in order to incorporate the hardware blocks implementing the proposed error control scheme, considering the possible error control policies introduced in the previous section.

0-7695-2918-6/07 $25.00

4.1. End-to-End Error Control Policy

In the e-e control policy the flits are encoded at the transmitter side, while the error detection/correction is performed only at the receiver side. No error control is performed by the intermediate routers.

In Fig. 2, we show the possible internal structure of a router implementing an e-e error control policy. Only one encoder and one decoder are inserted in the channel coming from/going to NI, respectively, thus requiring low area overhead compared to the original solution (i.e., without any error control coding). The encoder and decoder can be included in the NI itself.

Consider the router interfaced with the transmitter IP. Flits coming from the NI are encoded: *m* check bits are added to the original *k* bits. The whole encoded flit, composed by *k*+*m* bits, is then routed to the proper output channel and stored in the output queue.

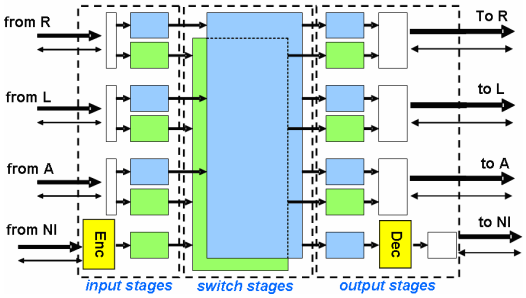

Fig. 2. Possible structure of a router implementing an end-to-end control policy.

If the router belongs to a routing node, the encoded flits are routed to the proper output channel and do not have to pass through the decoding logic.

When encoded flits arrive at the router of the destination IP, first they are routed to the output channel going to NI, then they are decoded, and finally the *k* original information bits are stored in the output queue, whose registers have a width equal to *k*.

As for area overhead, this control policy implies the addition of the following hardware: 1 encoder at the input channels and 1 decoder at the output channels; queue registers for *k*+*m* bits, but for the register to the NI, which has to store only the *k* information bits; switch stage for *k*+*m* bits.

As for delay, the additional delay per flit is $\Delta T_{D\text{-}Flit} = T_{Enc} + T_{Dec}$, where T_{Enc} and T_{Dec} denote the encoder and decoder input-output delay, respectively.

4.2 Switch-to-Switch Error Control Policies

In the s-s control policy, the flits are checked for errors at any hop. We propose two possible schemes: the first one allows minimizing the impact on area overhead, but incurs high latency; conversely, the second one allows minimizing the impact on performance, but at the cost of a high area overhead.

4.2.1 Solution s-s LA (Low Area Overhead). Fig. 3 shows the properly modified router structure. Each channel of the router, but for those from/to the NI, must be provided with an encoding/ decoding block: each output channel must be provided with an encoder, while a decoder has to be placed in each input channel.

In the router interfaced with the transmitter IP, flits coming from NI, composed by the *k* information bits, are routed to the proper output channel. Here they are encoded (*m* check bits are added to the original *k* bits) and the whole encoded flit, composed by *k*+*m* bits, is stored in the output queue. In this phase, a switch stage for only *k* bits is required.

Encoded flits passing through a routing node are first decoded, and then the *k* information bits provided by the decoder are stored in the channel queue and routed to the proper output channel. Here flits are encoded again and the whole encoded flit is stored in the output queue. Analogously to the previous case, also in this phase a switch stage for only *k* bits is required.

Finally, flits going to NI are decoded, properly routed, and then transferred to the destination IP (switch stage for *k* bits).

As for area overhead, this control policy implies the addition of: 3 encoders and 3 decoders at the output and at the input channels, respectively; queue registers of the input stages for k bits, queue registers of the output stages for *k*+*m* bits, queue registers to the NI for *k* bits; switch stage for *k* bits.

As for delay, the additional delay per flit (worst case estimation) is: $\Delta T_{D\text{-}Flit} = N_{max} \times (T_{Enc} + T_{Dec})$, where N_{max} is the maximum number of hops made by a flit.

4.2.2 Solution s-s HP (High Performance). Fig. 4 shows the properly modified router structure. In this scheme, only the input channels must be provided with an encoder (the channel from NI), or a decoder (the channels from R, L, A). Differently from the s-s LA solution, also the check bits are checked for errors and corrected. This way, no encoding scheme in the output stages is required. This advantage is counterbalanced by the need for a switch stage for *k*+*m* bits instead of *k*.

In the router interfaced with the transmitter IP, flits coming from NI, composed by the *k* information bits, are first encoded (*m* check bits are added to the original *k* bits); then the whole *(k+m)*-bit encoded flit is routed to the proper output channel and stored in an output queue. The switch stage and output queue registers

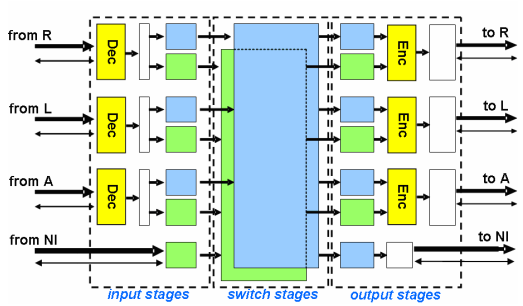

Fig. 3. Possible structure of a router implementing a La s-s error control policy.

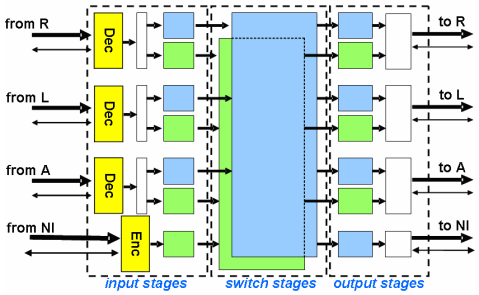

Fig. 4. Possible structure of a router implementing a La s-s error control policy.

have to be designed to route and store $k+m$ bits.

Encoded flits passing through a routing node are first decoded. Differently from the s-s LA solution, also the check bits are corrected (if erroneous) and provided as outputs by the decoders. Then, the whole encoded flit is stored in the input queue and routed to the proper output channel, where it is stored in the output queue.

Similarly, flits going to NI are first decoded; then only the k information bits are properly routed and stored in the output queue of the channel to NI, and finally transferred to the destination IP.

As for area overhead, this control policy implies the addition of: 3 decoders and one encoder at the input stages; queue registers of the input stages for $k+m$ bits, queue registers of the output stages for $k+m$ bits, queue registers to NI for k bits; switch stage for $k+m$ bits.

As for delay, the additional delay per flit (worst case estimation) is $\Delta T_{D\text{-}Flit} = T_{Enc} + N_{max} \times T_{Dec}$.

5. Proposed Error Control Scheme: Encoding/Decoding Blocks

The proposed error control scheme is represented schematically in Fig. 5.

The encoding block (employed for both the e-e and the s-s error control policies) is a standard check bit generator that, receiving as input the original flits

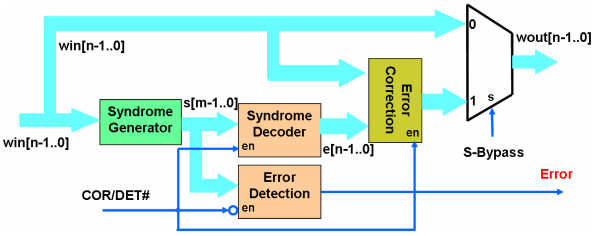

Fig. 5. Router decoding scheme.

composed by k bits *din[k-1..0],* generates m check bits *c[m-1..0*]. The m check bits, along with the original k-bit flits, form the whole ($n=k+m$)-bit encoded flits *wout[n-1..0].*

As for the decoding block, we must distinguish between the s-s and e-e error control policies. In the first case, we have the scheme shown in Fig. 5. During the decoding procedure, the whole encoded flit *win[n-1..0]* feeds the Syndrome Generator block. This computes the error syndrome *s[m-1..0],* which allows detecting whether or not the input word is erroneous.

In the *Correction Mode* (COR/DET# =1), the syndrome is decoded by the Syndrome Decoder, which generates an *Error Vector e[k-1..0]* containing all 0s, apart from the error bits corresponding to the data bits in error (*e.g.,* if only the data bit d_j is erroneous, it is $e_j = 1$ and $e_i = 0$, $\forall i \in [0..k\text{-}1]$, with $i \neq j$). The correction is performed by simply ex-oring the error vector with the input data. The output mux, controlled by the signal S-Bypass = 0, allows data coming from the correction path to be forwarded.

In the *Detection Mode* (COR/DET# =0), the Error Detection block has to determine only whether the syndrome is an all 0s vector, or not. The Error Detection block generates an *Error* flag, which can be employed by the receiving switch to generate a retransmission request, or simply to decide whether to propagate, or drop, the received flit. In this case, the output mux directly forwards the original data (S-Bypass = 1). Besides distinguishing between the *Correction Mode* and the *Detection Mode* when an s-s error control policy is applied, the S-Bypass signal is employed to differentiate between e-e and s-s control policies. It can be obtained as: S-Bypass = COR/DET# + SS/EE#.

The decoding scheme employed in the e-e error control mode is shown in Fig. 6. In this case, as described in Sect. 3, only the *Detection Mode* is considered. The syndrome generator and error detection blocks can be disabled when an s-s control policy is pursued, by asserting the control signal SS/EE#. Analogously to the *Detection Mode* described above (for the s-s policy), the *Error* flag generated by the detection block can be used to instantiate a retransmission phase.

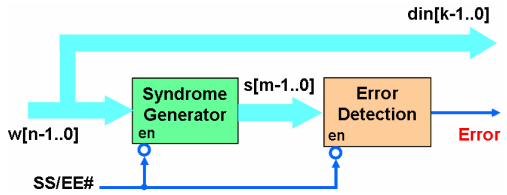

Fig. 6. NI decoding scheme.

6. Cost Evaluation and Comparison

We have evaluated the impact on area and performance of our configurable error control scheme implementing Hamming SEC code, Hsiao SEC/DED code and a symbol S_2SC code [13].

As for the s-s error control policies I, II and IV presented in Sect. 3, we have considered the solution s-s LA. In fact, as introduced in Sect. 4, this solution requires less area then the s-s HP solution, while implying a higher delay penalization. As for the mixed error control policies III and V, we have considered the s-s HP solution, along with the discussed e-e solution.

Area overhead and additional delay are expressed in terms of *equivalent gates* (EqGs), where for EqG we mean the area/delay of a 2-input AND/OR gate; XOR/XNOR gates are weighted as 2 EqGs for area, and 1.5 EqG for delay estimation. Besides the extra area required by the encoding/decoding blocks, also the size of queue registers and crossbars need to be increased when the flits are encoded.

As an example, in our estimation we have assumed the original size of header and payload flits equal to 16 and 32 bits, respectively. For these values of flits' parallelism, and assuming no implemented coding technique, we obtain an area of the routers equal to 7686 EqGs. This value is computed by neglecting the area due to the routing decision logic in the router [8].

Table 1 reports area overhead (expressed in EqGs) due to the implementation of the considered error control codes for the five different error control policies. Furthermore, it shows the area overhead express as a percentage over the area of the original routers (columns labeled as Δ%).

In the error control policies I, II and IV there are 3 encoders included in the routers, while the error control policies III and V requires only one encoder in the NI.

As for the decoders, error control policies I, II and IV requires 3 decoders in the routers, whose areas differ because of the provided error control abilities are different. In the first error control policy, whose decoders have the largest area, errors affecting both header and payload flits must be corrected; in the second error control policy errors affecting the header has to be corrected, while those affecting the payload

Table 1. Area overhead for the proposed error control policies

Area Overhead (EqG)							
Error Control Policy		(38, 32) SEC Hamming Code		(39, 32) SEC/DED Hsiao Code		(38, 32) S_2SC Code	
Header	Payload	EqG	Δ%	EqG	Δ%	EqG	Δ%
Cor. s-s LA	Cor. s-s LA	1464	19	1552.5	20	1686	22
Cor. s-s LA	Det. s-s	1111.5	14	1378.5	18	1440	19
Cor. s-s HP	Det. e-e	801.5	10	1030	13	979	13
Det. s-s LA	Det. s-s LA	951	12	1128	15	1155	15
Det. s-s HP	Det. e-e	530	7	679	9	628	8

flits requires only detection; the fourth error control policy requires that both header and payload errors are only detected. Error control policies III and V are characterized by a different error control for header and payload: header flit requires correction, and are checked and corrected (in case of errors) by the three decoders included in the router; errors in the payload flits have only to be detected by the decoder in the NI.

For all three error control codes considered, the error control policy I, requiring correction for both the header and the payload flits is that incurring the highest are overhead. Conversely, error control policy V is that requiring the lower area overhead, but is also that with the lower error protection ability. Error control policies II, III and IV can represent good trade-offs in terms of area and error protection. As expected, Hamming codes is that requiring less area overhead, since it provides the lowest error protection. Hsiao and S_2SC codes, although having different error protection characteristics, introduces comparable area overhead.

Table 2 reports the additional delay introduced by the considered error control policies. Of course, it depends on N_{max}. As an example, the last column shows the additional delay introduced in the propagation of a flit for $N_{max} = 3$. Analogously to the case of area overhead, error control policy I implies the highest additional delay, while the error control policy V is that requiring the lowest additional delay.

Furthermore, in case of an error control policy requiring a retransmission phase, we should consider also the additional delay penalization and area overhead introduced by the retransmission procedure. These contributions depend critically on whether we perform an s-s or an e-e control policy, and are not considered in our analyses.

Finally, let us now analyze the retransmission error recovery ability in case of crosstalk-induced errors. We can reasonably assume that the transition of a line (victim) experiences its worst case crosstalk-induced delay principally when the adjacent signals (aggressors) switch oppositely. If we assume that the crosstalk-induced delay (leading to an error) in the

Table 2. Additional delay per flit for the proposed error control policies.

Additional Delay per Flit (EqG)										
Error Control Policy		(38, 32) SEC Hamming Code			(39, 32) SEC/DED Hsiao Code			(38, 32) S_2SC Code		
Header	Payload	Enc	Dec	Tot (N_{max}=3)	Enc	Dec	Tot (N_{max}=3)	Enc	Dec	Tot (N_{max}=3)
Cor. s-s LA	Cor. s-s LA	$6N_{max}$	$11N_{max}$	51	$6N_{max}$	$12N_{max}$	54	$7.5N_{max}$	$15N_{max}$	22
Cor. s-s LA	Det. s-s	$6N_{max}$	$10N_{max}$	48	$6N_{max}$	$12N_{max}$	54	$7.5N_{max}$	$15N_{max}$	19
Cor. s-s HP	Det. e-e	6	$8.5N_{max}$	31.5	6	$12N_{max}$	42	7.5	$13.5N_{max}$	13
Det. s-s LA	Det. s-s LA	$6N_{max}$	$9N_{max}$	45	$6N_{max}$	$9N_{max}$	45	$7.5N_{max}$	$10.5N_{max}$	15
Det. s-s HP	Det. e-e	6	$7.5N_{max}$	28.5	6	$9N_{max}$	33	7.5	$9N_{max}$	8

victim is mainly impacted by the status of its two adjacent wires, in case of s-s retransmission the probability that an analogous error affecting occurs is given by $(½)^3$, and therefore the probability to have a successful retransmission is 87.5%.

Conversely, if a whole packet is retransmitted, as might be when an e-e error control policy is pursued, the erroneous flit encounters the same initial conditions that led to the error (since the flits sequence is the same). Therefore, in this case the probability that the error occurs again, leading to an unsuccessful retransmission, is very high.

7. Conclusions

In this paper we have proposed a novel error control scheme to cope with errors due to transient and crosstalk faults, affecting the communication links of a NoC. Our scheme can be configured in different operating modes (*Correction Mode*, *Detection Mode*, and *Mixed Mode*), thus allowing meeting different QoS levels in terms of error control. For each mode, we have proposed different error control policies (based on e-e and/or s-s control policies). As error control codes, we have considered SEC Hamming, SEC/DED Hsiao, and Symbol Error Correcting codes. We have evaluated advantages and drawbacks of each approach, in terms of signal integrity, required hardware modifications (router architecture and encoders/decoders), area overhead and impact on performance.

References

[1] International Technology Roadmap for Semiconductors, 2005, *http://www.itrs.net*.

[2] L. Benini, G. De Micheli, "Network on Chips: A New SoC Paradigm", *IEEE Computer*, Jan. 2002, pp. 70-78.

[3] M. Favalli, C. Metra, "Bus Crosstalk Fault-Detection Capabilities of Error Detecting Codes for On-Line Testing", *IEEE Trans. on VLSI Systems*, Sept. 1999, pp. 392-396.

[4] H. Zimmer, A. Jantsch, "A Fault-Model Notation and Error-Control Scheme for Switch-to-Switch Buses in a Network-on-Chip", *Proc. ACM CODES+ISSS'03*, 2003, pp. 188-193.

[5] P. Vellanki, N. Banerjee, K.S. Chatha, "Quality-of-Service and Error Control Techniques for Network-on-Chip Architectures", *Proc. GLSVLSI'04*, 2004, pp. 45-50.

[6] D. Rossi, A.K. Nieuwland, A. Katoch, C. Metra, "Exploiting ECC Redundancy to Minimize Crosstalk Impact", *IEEE Design & Test of Computers*, Jan.-Feb. 2005, pp. 59-70.

[7] R. R. Tamhankar, S. Murali, G. De Micheli, "Performance Driven Reliable Link Design for Networks on Chips", *ASP-DAC* 2005, pp. 749-754.

[8] S. Murali, *et al.*, "Analysis of Error Recovery Schemes for Networks on Chips", *IEEE Design &Test of Computers*, Sep.-Oct. 2005, pp. 434-442.

[9] P. P. Pande, *et al.*, "Design of Low Power & Reliable Networks on Chip Through Joint Crosstalk Avoidance and Forward Error Correction Coding", in *Proc. of IEEE Int'l Symp. on Defect and Fault Tolerance in VLSI Systems (DFT'06)*, 2006, pp. 466-476.

[10] D. Park, *et al.*, "Exploring Fault-Tolerant Network-on-Chip Architectures", in *IEEE Proc. of 2006 Int'l Conf. on Dependable Systems and Networks*, 2006, pp. 93-104.

[11] A. Frantz, F. Kastensmidt, L. Carro, E. Cota, "Dependable Network-on-Chip Router Able to Simultaneously Tolerate Soft Errors and Crosstalk", in *Proc. of IEEE Int'l Test Conf.*, 2006, pp. 1-9.

[12] M. Coppola, R. Locatelli, G. Maruccia, L. Pieralisi, A. Scandurra, "Spidergon: A Novel On-Chip Communication Network", *IEEE Proc. Int'l Symp. on System-on-Chip*, 2004, 16-18 Nov., 2004, p. 15.

[13] T. Rao, E. Fujiwara, *Error Control Coding for Computer Systems*, Prentice-Hall, 1989.