Error Control Schemes for On-Chip Communication Links: The Energy–Reliability Tradeoff

Davide Bertozzi, Luca Benini, *Senior Member, IEEE*, and Giovanni De Micheli, *Fellow, IEEE*

***Abstract*—On-chip interconnection networks for future systems on chip (SoC) will have to deal with the increasing sensitivity of global wires to noise sources such as crosstalk or power supply noise. Hence, transient delay and logic faults are likely to reduce the reliability of across-chip communication. Given the reduced power budgets for SoCs, in this paper, we develop solutions for combined energy minimization and communication reliability control. Redundant bus coding is proved to be an effective technique for trading off energy against reliability, so that the most efficient scheme can be selected to meet predefined reliability requirements in a low signal-to-noise ratio regime. We model on-chip interconnects as noisy channels and evaluate the impact of two error recovery schemes on energy efficiency: correction at the receiver stage versus retransmission of corrupted data. The analysis is performed in a realistic SoC setting, and holds both for shared communication resources and for peer-to-peer links in a network of interconnects. We provide SoC designers with guidelines for the selection of energy efficient error-control schemes for communication architectures.**

***Index Terms*—Bus encoding, on-chip communication, power, reliability.**

I. Introduction

THE integration of a large number of functional blocks on the same silicon die is becoming technically feasible, and the design paradigm for systems on chip (SoC) is shifting from being device-centric to being interconnect-centric. Performance and energy consumption will be increasingly determined by the communication architecture. Under very high integration densities, on-chip realization of interconnection networks [networks on chip (NoC)] is emerging as the most efficient solution for communication. NoCs can be viewed as an adaptation of the wide-area network paradigm, well known to the communication community, to the deep submicron (DSM) IC scenario. In this context, micronetworks of interconnects can take advantage of local proximity and of a lower degree of nondeterminism, but have to meet new distinctive requirements such as design-time specialization and energy constraints [21].

Energy dissipation is a critical NoC design constraint, particularly in the context of battery-operated devices, and will be the focus of this paper. The International Technology Roadmap for Semiconductors [13] projects that power consumption can marginally scale up while moving from 90- to 35-nm technology. At the same time, projected clock frequency and number of devices on-chip are increased significantly. By the end of the decade, with a 50-nm technology about 4 billion transistors running at 10 GHz and operating below 1 V will be integrated into a single chip. It is considered that the power consumption of CMOS chips will steadily be increased as a natural result of device scaling and of the consequent on-chip device density.

In energy-constrained systems, low-power CMOS circuit techniques are needed to extend the battery and system lifetimes. The most widely used strategies include clock gating [4], dynamic voltage/frequency scaling [28], and low voltage design with variable/multiple $V_{\rm dd}/V_{\rm th}$ control [11].

Dynamic voltage scaling (DVS) dynamically adapts processor speed to current computational requirements. A transmission scheme applying DVS to chip-to-chip interconnection networks has been introduced in [30], while the extension to on-chip communication was done in [38], wherein adaptive voltage-swing signaling is used for interconnects without relying on *a priori* knowledge of working conditions.

Lowering supply voltage $V_{\rm dd}$ decreases dissipated energy quadratically but also results in a performance degradation. As a consequence, the threshold voltage $V_{\rm th}$ should be lowered as well, even though the side effect is an increase of the leakage current [14]. The tradeoff between energy and performance, and the optimal $V_{\rm dd} - V_{\rm th}$ operating region for CMOS circuits is investigated in [37].

Computation and storage energy greatly benefits from device scaling (smaller gates, smaller memory cells), but the energy for global communication does not scale down. On the contrary, projections based on current delay optimization techniques for global wires show that on-chip communication will require increasingly higher energy consumption. Hence, communication-related energy minimization will be a growing concern in future technologies, and will create novel challenges that have not yet been addressed by traditional high-performance network designers.

An effective approach to high-speed energy-efficient communication is low swing signaling [39]. Even though it requires the design of receivers with good adaptation to line impedance and high sensitivity (often achieved by means of simplified sense-amplifiers), power savings in the order of $10\times$ have been estimated with reduced interconnect swings of a few hundreds of mV in a 0.18-μm process [35].

The use of low swing signaling poses a critical challenge for NoC design: communication reliability has to be provided in

Manuscript received May 5, 2003; revised November 13, 2003 and May 17, 2004. This paper was recommended by Associate Editor M. Pedram.

D. Bertozzi is with the Department of Electrical Engineering, University of Bologna, 40136 Bologna, Italy (e-mail: dbertozzi@deis.unibo.it).

L. Benini is with the Department of Electronics and Computer Science, University of Bologna, 40136 Bologna, Italy (e-mail: lbenini@deis.unibo.it).

G. De Micheli is with the Computer Science Laboratory, Stanford University, Stanford, CA 94305 USA (e-mail: nanni@stanford.edu).

Digital Object Identifier 10.1109/TCAD.2005.847907

spite of the decreased noise margins and the strong technology limitations, under limited power budgets.

With present technologies, most chips are designed under the assumption that electrical waveforms can always carry correct information on chip. As we move to consider DSM NoCs, communication is likely to become inherently unreliable because of the increased sensitivity of interconnects to on-chip noise sources, such as crosstalk and power-supply noise.

As a consequence, solutions for combined energy minimization and communication reliability control have to be developed for NoCs. The contribution of this work is to investigate the energy–reliability tradeoff for a peer-to-peer link of a SoC communication architecture in a low SNR regime. With respect to the unencoded link (a shared system bus or a switch-to-switch link in a network of interconnects), we show that redundant bus encoding provides a degree of freedom for spanning the energy–reliability tradeoff. By means of extensive simulations on a realistic SoC setting, we come up with a comparison of the energy efficiency of different link coding schemes, assessing their ability to meet predefined communication reliability constraints with the minimum energy dissipation. To this purpose, we define a unique metric for characterizing each coding scheme, which takes into account the main parameters of interest: detection capability, average-energy-per-bit for encoder and decoder operation and for transfers across link lines.

The key point of our analysis is to model on-chip interconnects as noisy channels, and to exploit the error detection capability of the coding schemes, that would provide a link transfer reliability in excess with respect to the constraints, to decrease the voltage swing, resulting in an overall energy saving (compared to the unencoded link) in spite of the overhead associated with the code implementation. Our results also indicate that the energy efficiency of a code is tightly related to its error recovery technique, namely error correction or retransmission of corrupted data. This issue resembles the tradeoff investigation between forward error correction (FEC) and automatic repeat request (ARQ), well known to the communication community [40], but for on-chip communication networks this study is still in its early stage. The results derived throughout this paper provide SoC designers with guidelines for the selection of energy efficient error control schemes for communication architectures.

In Section II, the traditional approach to fault tolerance is described, while Sections III and IV introduce the role played by redundant bus encoding in the energy–reliability tradeoff for SoC communication. In Section V, an energy efficiency metric is defined and simulation results are reported in Sections VI and VII. In Section VIII, further developments about multihop NoCs are discussed, while conclusions are drawn in Section IX.

II. Traditional Approach to Fault Tolerance

Fault tolerance of computing systems is usually ensured by incorporating *redundancy* (i.e., additional resources) so that computation can be correctly carried out even in presence of faults [32].

Hardware redundancy has been widely adopted to implement reliable systems, and relevant examples include hardware duplication or triple modular redundancy [24]. However, under tight area and power constraints this solution might turn out to be impractical because of the large hardware overhead.

Alternatively, the addition of redundant information (information redundancy) to the original data can be used. As an example, error detecting and correcting codes append check bits to the data bits to enable detection and correction of erroneous bits.

Error-detecting codes are widely deployed for the implementation of self-checking circuits (SCCs), mainly because of design cost considerations and because they allow error recovery to be carried out either in hardware or in software [18]. Basically, a functional unit provides an information flow protected by an error detecting code, so that a checker can continuously verify the correctness of the flow and provide an error indication as soon as it occurs. Error correcting codes would on the contrary incur performance penalties related to additional correction circuitry.

Using information redundancy has less impact on hardware and is therefore of interest for on-chip realizations, even though the distinctive characteristics of the on-chip scenario with respect to multichip systems lead to the deployment of different error control schemes, as will be hereafter discussed.

Two frequently used codes in SCCs are the *parity check* and *two-rail codes*. The former adds only one parity bit to the information bits and detects all error patterns of an odd number of bits, but cannot detect double errors that can be relevant in a crosstalk-dominated scenario.

The two-rail code represents a signal as a pair of two complementary variables $(x_i, \bar{x}_i)$, thus doubling the number of bus lines. This overhead may not always be acceptable in spite of the high error detection efficiency provided by this code [23].

It has been observed that many faults in very large scale integration (VLSI) circuits cause unidirectional errors (i.e., 0–1 or 1–0 errors, provided the two kinds of errors do not occur simultaneously). Therefore, coding schemes targeting this kind of error are well-known to the testing community, such as the *m-out-of-n* and *Berger codes*.

In particular, the Berger code is the optimal separable all-unidirectional error-detecting code: no other separable code can detect all unidirectional errors with a fewer number of check bits [27]. The check bits are the binary representation of the number of 0s counted in the information bits.

As technology scales toward DSM, traditional schemes used in SCCs may lose their detection effectiveness when applied to on-chip buses, because of the new noise sources that come into play. Unidirectional errors cannot efficiently describe the effects of these noise sources, and the detection capability of multiple bidirectional errors instead of unidirectional errors will be the distinctive feature of error control schemes for DSM NoCs. For instance, crosstalk causes bidirectional errors, when two coupled lines switch in the opposite direction and both transitions are delayed inducing sampling errors.

Many solutions have been proposed to overcome the detection capability limitation of traditional error control schemes with respect to multiple bidirectional errors:

1) Acting on the layout in both a code-independent way (i.e., spacing rules, shielding, line crossing, etc.) or

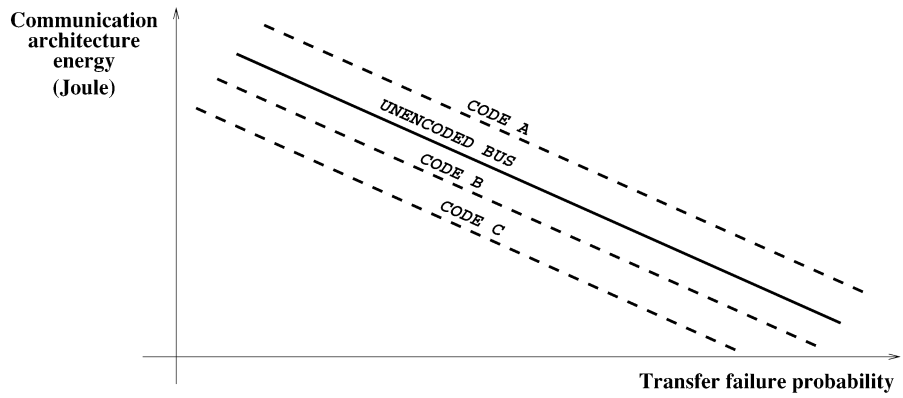

Fig. 1. Qualitative plot of communication energy versus reliability, for different error control schemes.

code-driven way (e.g., keeping the two complementary bits as far apart as possible in a two rail code). Alternatively, layout information can be exploited to come up with weight-based codes, i.e., extensions of Berger or m-out-of-n codes that are able to deal more efficiently with bidirectional errors [7], [8].

2) Acting at the electrical level (e.g., the probability of single errors can be increased with respect to that of bidirectional ones by unbalancing bus lines drivers).
3) Using suitable detectors capable to deal with the effects of specific errors (e.g., crosstalk induced errors), but they might not be available is some design styles.

The major drawback of the above-mentioned approaches is the need to have layout knowledge or to act at the electrical level. A more general approach could be desirable, wherein the proper course of action against multiple bidirectional errors can be taken early in the design stage, independent of the technology and the final layout, the knowledge of which is not generally available in advance.

Redundant bus encoding remains the most efficient approach for this purpose. Yet, new codes must be used, targeting a more general class of errors than unidirectional ones. Linear codes could be a viable solution, in that they target error multiplicity rather than error direction. Moreover, their codecs can exhibit very lightweight implementations and can provide optional correcting capabilities.

III. Energy-Reliability Tradeoff

In DSM SoCs, communication reliability cannot be tackled without considering the impact on energy efficiency. These two related issues are brought up by the scaling scenario and their interaction can be briefly summarized as follows:

1) Low swing signaling reduces signal-to-noise ratio [10], thus making interconnects inherently sensitive to on-chip noise sources such as cross-talk, power supply noise, electromagnetic interferences, soft errors, etc. [31] [2] [17] [1]. This sensitivity is increased by the reduction of receiving gates voltage noise margins as an effect of the decreased supply voltages.

   Therefore, a tradeoff exists between communication reliability and energy dissipation, as depicted in the qualitative plot of Fig. 1. The lower the probability of a codeword being received in error, the higher the energy cost that has to be sustained by the communication architecture. Very high detection capabilities have to be ensured by more complex codecs and wire voltage swings have to be kept high to preserve high SNR values.
2) Coupling capacitance between adjacent wires is becoming the dominant component of interconnect capacitance. This has an impact not only on signal integrity, but also on power consumption, as most power consumed by interconnects is associated with switching of coupling capacitances (*coupling power*).

With reference to on-chip buses, encoding strategies have been successfully exploited by the low-power design community to save energy by reducing the switching activity on long wires (*low power encoding*) [3], [25], [34]. As technology scales toward DSM, coupling effects become more significant. This consideration has led to the development of *energy-efficient* and *coupling-driven* bus encoding schemes, that consider coupling power in their power minimization framework [16].

Another approach is viable, that consists of employing linear codes to meet predefined requirements on communication reliability, and of minimizing energy consumption by reducing interconnect voltage swing [5], [6]. This idea was first proposed by Hegde and Shanbhag [9], who tried to achieve energy efficiency via noise-tolerant coding in presence of DSM noise.

The energy efficiency of a code is a measure of its ability to achieve a specified communication reliability level with minimum energy cost. It is traditionally expressed in terms of energy dissipated per bit [9].

Each coding scheme is able to meet the reliability constraints with different energy costs, according to its intrinsic characteristics, and this allows to search for the most efficient code from an energy viewpoint (see Fig. 1).

This paper takes the same approach as the theoretical study in [9], but comes up with a practical framework for the design of energy efficient and reliable communication architectures. In particular, the distinctive contribution of our paper is threefold.

1) The idea of minimizing communication energy by means of redundant bus encoding has been implemented and explored in the communication architecture of a realistic SoC setting.

2) Besides evaluating energy associated with bus transitions, we also consider the impact of encoder and decoder complexity and operation on energy consumption. For small wire load capacitances, this contribution is likely to make the difference.
3) We make a comparison between error correction and error detection mechanism with retry procedure, and come up with guidelines to select the most efficient error recovery technique from an energy viewpoint.

For the on-chip context, we propose to use some simple codes frequently employed in the communication domain so to meet the tight area and energy budgets.

Our code comparison framework is based on a detailed analysis of the energy dissipated by each component of the communication channel (encoder, link and decoder) and of their relative impact on the energy cost for transferring information bits across the communication subsystem. Our analysis targets energy efficiency of link-level error control mechanisms, and is therefore carried out at the level of the communication subsystem. In this way, fine-grained details can be observed, such as the assessment of whether energy associated with link-level retransmissions actually results in an inherently less efficient mechanism for reliable communication with respect to error correction from an energy viewpoint.

On the other hand, this kind of analysis ends up hiding system-level effects such as the impact of the adopted link-level error recovery techniques on application hard- or soft-real time constraints or on the energy dissipated by other system components. For example, retransmissions result in a longer execution time, and this causes other system components to be active for a longer time thus dissipating more energy. Such a system-level energy–reliability tradeoff investigation in presence of time constrained traffic is outside the scope of this paper, which aims at providing low level, accurate energy efficiency characterization of different reliable communication channels. The derived results can then be used within a more general analysis framework combining application, system and technology (e.g., bit error rate) information.

In the next section, details are given about a class of linear codes (Hamming codes), whose flexibility and optimality in the number of parities make it suitable for micronetwork applications. In addition, we also take cyclic codes into consideration: they represent a class of linear codes characterized by highly efficient hardware implementations and by a high resilience to burst errors.

IV. Linear Codes

In block coding, the binary information sequence is segmented into message blocks of fixed length; each message block, denoted by u, consists of k information digits. There are a total of 2^k distinct messages. The encoder, according to certain rules, transforms each input message u into a binary n-tuple v with $n > k$ (n is said to be the codeword length and $n - k$ represents the number of check bits). This binary n-tuple v is referred to as the code word of the message u (see Fig. 2). Therefore, corresponding to the 2^k possible messages, there are 2^k code words. This set of 2^k code words is called a block code. For a block code to be useful, the 2^k code words must be distinct. A binary block code is *linear* if and only if the modulo-2 sum of two code words is also a code word [20]. The block code given in Table I is a $(n, k) = (7, 4)$ linear code, and is reported as an example.

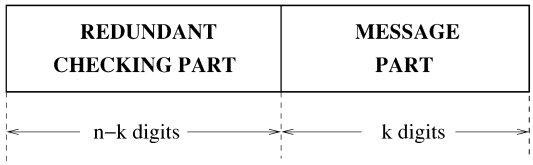

Fig. 2. Systematic format of a code word, wherein message and checking part are kept separate.

TABLE I
Linear Block Code With $k = 4$ and $n = 7$

0 0 0 0	0 0 0 0 0 0 0
1 0 0 0	1 1 0 1 0 0 0
0 1 0 0	0 1 1 0 1 0 0
1 1 0 0	1 0 1 1 1 0 0
0 0 1 0	1 1 1 0 0 1 0
1 0 1 0	0 0 1 1 0 1 0
0 1 1 0	1 0 0 0 1 1 0
1 1 1 0	0 1 0 1 1 1 0
0 0 0 1	1 0 1 0 0 0 1
1 0 0 1	0 1 1 1 0 0 1
0 1 0 1	1 1 0 0 1 0 1
1 1 0 1	0 0 0 1 1 0 1
0 0 1 1	0 1 0 0 0 1 1
1 0 1 1	1 0 0 1 0 1 1
0 1 1 1	0 0 1 0 1 1 1
1 1 1 1	1 1 1 1 1 1 1

A. Hamming Codes

Hamming codes have been the first class of linear codes devised for error correction and have been widely employed for error control in digital communication and data storage systems.

The decoder for a Hamming code is reported in Fig. 3. When the transmitted codeword is received, an error detecting stage takes care of checking parity bits. If a correction stage is applied, the exact location of the error can be identified so that the corrupted bit can be restored. The Hamming code is exactly a distance-3 code (e.g., the smallest distance between two distinct codewords is at least three bits), and is therefore a single error correcting code [20].

The whole circuit of Fig. 3 can be implemented as an EXOR tree. Note that the final correction stage is optional, and this makes Hamming code very flexible from an implementation standpoint. However, the use of error correction ends up limiting the detecting capability of the code and seriously affecting implementation complexity, as will be hereafter described. The design of different decoder schemes led us to the realization of several versions of the Hamming code:

Single Error Correcting Code—SEC: This is a basic implementation of a (38, 32) Hamming code, with single error correction capability. The purpose of this scheme is to highlight the characteristics of a recovery strategy based on correction.

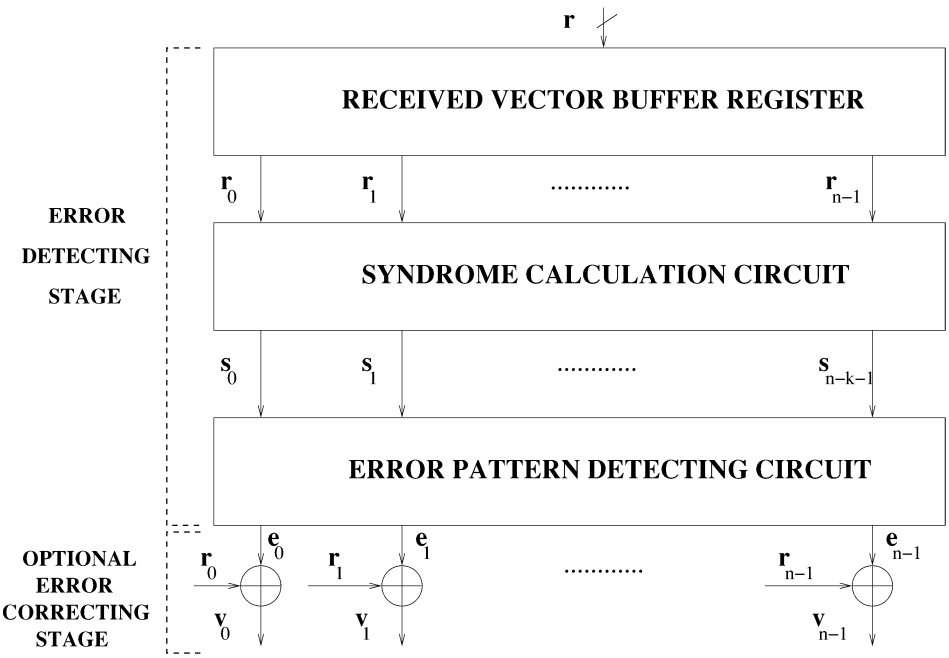

Fig. 3. Decoder for a Hamming code.

The decoder is more complex than the encoder, because of the correction circuitry. Six additional check bits are required.

SEC and Double Error Detecting Code—SECDED: A distance-3 Hamming code, like that implemented in SEC, can be easily modified to increase its minimum distance to 4, adding one more check bit, chosen so that the parity of all of the bits, including the new one, is even [36]. This version of the Hamming code, that features seven check bits instead of the six of the previous version, is traditionally used for single error correction and double error detection. Yet it allows to detect also all error patterns of an even number of errors, even though the double ones are the most meaningful for their higher probability. In case a double or multiple error is detected, the recovery action to take would be the retransmission of the wrong data word. Here, the codec is slightly more complex than that of SEC because of the combined approach of correction and retransmission.

Error Detecting Code—ED: Using the Hamming code for detection purposes only, it is possible to exploit its full detection capability, which includes not only all single and double errors, but also a large amount of multiple errors. An (n, k) linear code is capable of detecting $2^n - 2^k$ error patterns of length n. The undetectable error patterns are $2^k - 1$, and they are identical to the nonzero code words. This scheme exhibits the same encoder as SEC but a more simplified decoder, because it only has to compute and check the syndrome bits. We select ED to highlight the characteristics of a retransmission oriented approach.

Hamming codes are promising for application to on-chip micro-networks because of their implementation flexibility, low codec complexity and multiple bidirectional error detecting capability. Note however that when correction is carried out, the detection capability of the code is reduced, because restrictive assumptions have to be made on the nature of the error. This explains why for a linear code the probability of a decoding error is much higher than the probability of an undetected error [20].

B. Cyclic Codes

Cyclic codes are a class of linear codes with the property that any code word shifted cyclically (an end-around carry) will also result in a code word. For example, if $c_{n-1}, c_{n-2}, \ldots, c_1, c_0$ is a code word, then $c_{n-3}, \ldots, c_0, c_{n-1}, c_{n-2}$ is also a code word.

Cyclic redundancy check (CRC) codes are the most widely used cyclic codes (e.g., in computer networks), and the new DSM scenario could raise the interest for their on-chip implementation, thanks to their properties and to their low complexity codec.

Components of a code word are usually treated as the coefficients of a polynomial in order to develop the algebraic properties of a cyclic code. An (n, k) cyclic code is completely specified by its nonzero code polynomial of minimum degree $g(x)$ (the *generator polynomial*). In fact, the code word for k message bits can be obtained by multiplying the message polynomial by $g(x)$, and the degree of $g(x)$ is equal to the number of parity-check digits of the code [20].

An (n, k) cyclic code is capable of detecting any error burst of length $n-k$ or less, including the end-around bursts [20]. The length of a burst is the span from first to last error, inclusive.

Unlike the Hamming code, CRC code exhibits the additional feature to target both random error patterns and error bursts. A Hamming code relies on its capability to detect multiple errors arbitrarily located all over the codeword, while CRC code is particularly suitable to deal with errors affecting lines that are close to each other (within the detectable burst length). With respect to the application to on-chip reliable communication, CRC code is effective whenever the communication failure mechanism affects a certain number of contiguous lines. DSM technologies are particularly sensitive to these effects, in that the shrinking of geometries makes the relative distance between interconnects smaller, therefore even localized noise sources are likely to have an impact on multiple contiguous bus lines. Fig. 4(a) shows a relevant example thereof.

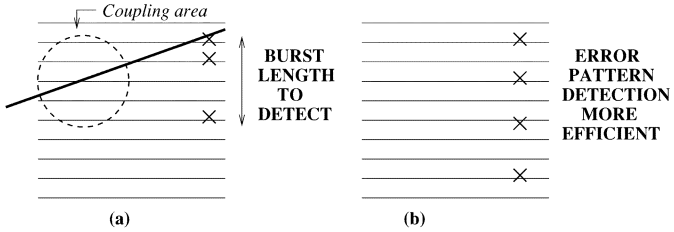

Fig. 4. (a) Upper layer wire crosses a bus, and capacitive coupling effects are likely to induce errors on a group of contiguous lines. CRC codes can be efficiently used in this context. (b) Example of error pattern induced by interwire coupling capacitances. Errors are sparse all over the bus, and could be effectively detected by a Hamming code.

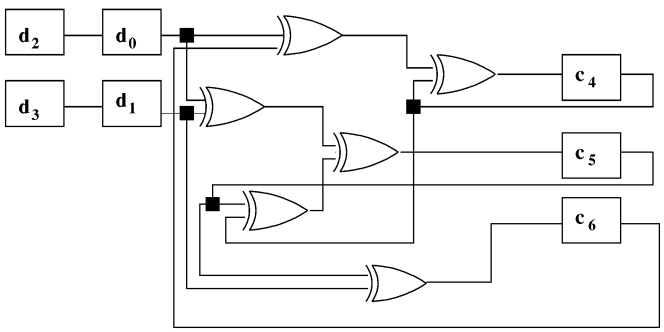

Fig. 5. Serial-parallel implementation of a (7, 4) cyclic code executing in two clock cycles.

On the contrary, Hamming codes can be efficiently used whenever errors are spread all over the bus and the detection of error patterns is more effective than that of error bursts (see Fig. 4). This is the case, for instance, of crosstalk faults induced by capacitive coupling among neighboring wires (interwire crosstalk): multiple errors might affect the same codeword on the bus as an effect of the presence of multiple crosstalk sensitive patterns in the transmission of two successive codewords (e.g., 000–101).

The process of encoding cyclic codes essentially involves the calculation of the parity check bits for a given message word. In general, this can be achieved in two ways [33].

1) Use a linear feedback shift register to generate check bits.
2) Calculate the check bits directly using parity check equations.

Method 1 is generally relatively simple but can become very slow, whereas method 2 is relatively fast but prohibitively complex when a large number of check bits are required. However, alternative circuit implementations do exist, that derive from the theory of series-parallel m-sequence generation and provide the desired tradeoff between the speed and complexity factors of methods 1 and 2. Compact encoder implementations like that reported in Fig. 5 can be obtained (it encodes four data bits generating three check bits in two clock cycles), resulting in a very limited on-chip area occupancy.

For our analysis, we have chosen two among the simplest possible CRC codes. The choice of the generator polynomial was mostly driven by the need for a lightweight implementation. In particular, we considered a CRC8 code generated by a standard LRCC-8 generator polynomial:

$$g(x) = x^8 + 1 \tag{1}$$

and a CRC4 code with generator polynomial of degree 4

$$g(x) = x^4 + 1. \tag{2}$$

The CRC8 generator polynomial has a degree of 8 and is a factor of $1 + x^{40}$ and, therefore, generates a (40, 32) cyclic code with eight parity-check bits and is therefore able to detect all error bursts of length equal to or less than 8 [20]. Instead, polynomial $x^4 + 1$ divides $1 + x^{36}$ and generates a (36, 32) cyclic code with 4 parity-check bits. The burst error detection capability includes all error bursts of length less than or equal to 4.

Both CRC4 and CRC8 can detect single errors (because their generator polynomials consist of more than one term) and any odd number of errors $(1, 3, 5, \ldots)$. Unfortunately, the selected polynomials are not primitive, therefore, no closed-form analytical models can be derived to express the detection capability of double errors [22]. Therefore, we opted for the worst-case assumption that they can detect only single and burst errors, which is usually enough to motivate the adoption of CRC codes from a practical viewpoint.

In order to accurately assess efficiency of linear codes used to span the energy–reliability tradeoff for on-chip communication links, we proceeded with the synthesis of encoder and decoder for the considered coding schemes and for a 32 bit link, coming up with the results reported in Table II. In addition, a simple *single parity bit code* (PAR) is considered for comparison, as it involves minimum redundancy in terms of link lines (only one check bit). Its detection capability includes all errors in an odd number of bits.

All of the error control schemes use retransmission as error recovery strategy, except for SEC and SECDED that correct single errors. A 0.25-μm synthesis library has been used, with a supply voltage of 2.5 V. Average power figures for the codecs have been obtained by means of back-annotation of switching activity of gates' internal nodes from very high speed hardware language (VHDL) simulation. Then, Synopsys Power Compiler was used for power estimation. Note that VHDL simulation is performed (for all codes) for the different cases of reliable communication: one simulation run includes only error-free transfers, the next one only transfers affected by (artificially generated) single errors (if single errors are detected), and so on. One hundred iterations of the Dhrystone benchmark were used to stimulate transfers across the encoded/unencoded bus.

It can be observed that CRC codes exhibit the most lightweight implementations, comparable to that of a single parity check code. Area and power metrics for CRC4 and CRC8 are sometimes counterintuitive, because the final circuit realization is strongly dependent on the generator polynomial.

V. Energy Efficiency Metric

As the focus here is on the inherent energy efficiency of the coding schemes (including their error recovery techniques) used for reliable communication, we employ the *average energy per useful bit* as assessment and comparison metric.

We make the assumption that a common predefined constraint on communication reliability has to be met by all of the schemes, and the metric expresses the energy cost incurred by each of them to achieve this goal. The different features of each code (detection capability, redundancy, error recovery technique) are all translated into a different contribution to the energy-per-bit. For instance, the energy overhead associated with redundant parity lines is ascribed to the information (i.e., useful) lines, thus considering the impact of coding efficiency on energy efficiency.

We also assume that whenever an undetectable error occurs or a decoding error is made, the system crashes. Therefore, for each coding scheme, we have to consider only those cases wherein the system works properly. As an example, Hamming SEC works correctly both in the error-free case and in the single error case. For each case of functionally correct operation, we consider: 1) the probability of occurrence p (analytically derived) and 2) the average energy consumption per information bit $\bar{E}$, derived as the sum of three contributions

$$\bar{E} = \bar{E}b + \bar{E}e + \bar{E}d \tag{3}$$

where $\bar{E}b$ is the average energy per useful bit spent for link transitions and $\bar{E}e$ and $\bar{E}d$ express the average energy consumption of encoder and decoder, respectively.

For the sake of mathematical formulation, the cases of correct operation have been indexed. Thus, for Hamming SEC, an index i ranging from 0 to 1 has been introduced that identifies the cases wherein communication reliability is preserved. So, for Hamming SEC, the error-free case occurs with probability p_0 and the average communication and codec-related energy consumption per useful transferred bit is $\bar{E}_0$, while in the single error case (probability p_1) the average energy is $\bar{E}_1$. As already observed in Table II, the energy contribution of the decoder is larger for the single error case than the error-free case because of the overhead for error correction. For schemes using retransmission instead of correction, we observe a more significant contribution of the link-related energy.

Finally, we derive average energy per useful bit by means of a weighted average of the energies consumed in the different cases. For Hamming SEC, we get

$$\bar{E}_{\mathrm{ub}} = \frac{p_0\bar{E}_0 + p_1\bar{E}_1}{p_0 + p_1}. \tag{4}$$

More in general, the *average energy per useful bit* can be defined as

$$\bar{E}_{\mathrm{ub}} = \frac{\sum_i p_i\bar{E}_i}{\sum_i p_i} \tag{5}$$

where p_i is the probability of having $i = 0, 1, 2, \ldots, n$ simultaneous errors affecting the transfer of a codeword (of length n), and $\bar{E}_i$ is the average energy consumed by the coding scheme implementation in that case. Obviously, not all values of i are considered in the metric, but only those ones corresponding to the number of errors that can be handled (detected or corrected) by a certain error control code. For example, all even values of i are not considered for PAR, as a single parity check code cannot detect errors in an even number of bits. All average energies are referred to useful information bits.

In the above-defined metric, the denominator can be thought of as the probability of correct operation of the system, and it has the same value for all schemes that have to be compared, as it represents the common predefined communication reliability requirement. The probability of a decoding or undetected error will be hereafter referred to as *residual error probability* (REP) *for a codeword transfer*, and its inverse is proportional to the *mean time to failure* (MTTF).

Energy values needed to compute the metric are partly derived from the results of Table II. In particular, average power values of the codecs can be translated into energy-per-bit values by considering latency for bus transfers (with correction or retransmission in presence of errors) and the exact number of bus transitions. In order to make link transitions' contribution to energy-per-bit explicit in the metric, the voltage swing used across interconnects needs to be derived.

Its value is tightly related to the communication reliability, and the simple model proposed by Hegde and Shanbhag [9] will be used in this paper. They make the assumption that every time a transfer occurs across a wire, it can make an error with a certain probability ϵ. The parameter ϵ depends on the knowledge of different noise sources and their dependence on the voltage swing, and is therefore difficult to estimate. So, for purpose of statistical analysis, the sum of several uncorrelated noise sources affecting each line of a point-to-point link is modeled as a single Gaussian noise source, and the value of ϵ depends on the voltage swing V_{sw} and the variance σ_N^2 of the noise voltage V_N

$$\epsilon = Q\left(\frac{V_{\mathrm{sw}}}{2\sigma_N}\right) \tag{6}$$

where $Q(x)$ is the Gaussian pulse

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}}\, dy. \tag{7}$$

This model is based on the assumption of uncorrelated link lines, and accounts for the decrease of noise margins (and hence for an increase of the line flipping probability ϵ) caused by a decrease of the voltage swing across a line. Overall, the model allows an investigation of the energy–reliability tradeoff for the introduced linear codes applied to on-chip communication links.

VI. Simulated SoC Setting

The energy–reliability tradeoff is analyzed for different bus coding schemes, each one working at different voltage swings depending on their error detection capability. This study has been conducted on the communication architecture of a real SoC setting, dealing with real implementation issues and exploiting the power analysis of synthesized blocks. We also consider the impact on energy efficiency of the error recovery technique, and this forced us to work on communication architectures featuring error recovery capabilities. As we did not want to come up with innovative communication infrastructures, we focused on a standard AMBA bus implementation, and we modified it to support retransmissions of corrupted data in a fully transparent way with respect to the AMBA specification. To this purpose, AMBA built-in mechanisms were exploited.

TABLE II
CHARACTERISTICS OF SYNTHESIZED CODECS FOR DIFFERENT HAMMING AND CRC CODES, AND THEIR AVERAGE PERFORMANCE IN SOME CASES OF INTEREST, WHEREIN COMMUNICATION COMPLETES SUCCESSFULLY

System		Parity	Area		Power (uW)		Delay (ns)	
Scheme	Error	Lines	Enc.	Dec.	Enc.	Dec.	Enc.	Dec.
UNENCODED	Free	-	-	-	-	-	-	-
SEC	Free	6	5,022	11,034	153	233	1.61	4.56
	Single				153	279		
SECDED	Free	7	6,588	14,238	205	308	2.31	4.85
	Single				205	360		
	Double				254	363		
ED	Free	6	5,022	5,049	153	146	1.61	1.73
	Single				190	144		
	Double				190	148		
	Triple				190	152		
PAR	Free	1	2,538	2,592	61	63	1.40	1.45
	Single				77	62		
	Triple				77	66		
CRC-4	Free	4	2,376	2,637	54	62	0.65	1.02
	Single				68	61		
	Burst				68	65		
CRC-8	Free	8	2,160	2,700	47	58	0.39	0.88
	Single				59	59		
	Burst				59	62		

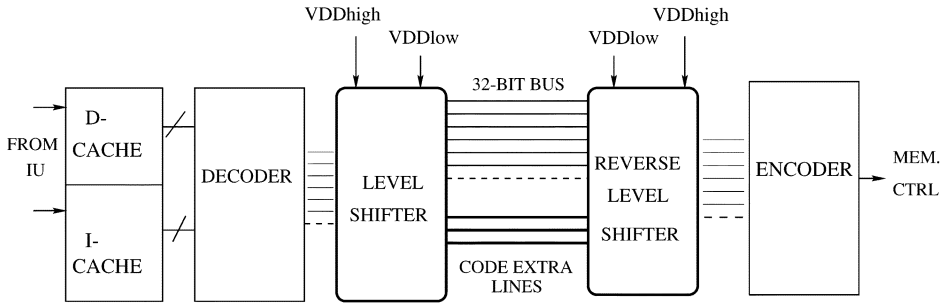

Fig. 6. SoC setting wherein the energy–reliability tradeoff is investigated.

More precisely, we worked on the setting depicted in Fig. 6. We considered a subset of the VHDL model of a 32-bit processor compliant to the SPARC V8 architecture (called Leon [19]), and provided it with noise tolerance. The Leon processor, designed for embedded systems, has a full implementation of the AMBA bus. We focused on the read data bus (Fig. 6), accessed by the I- and D-cache whenever an instruction/data miss occurs. The cache accesses the bus by means of a hardware interface that generates the AMBA bus control signals in compliance with its timing requirements. The request for data is transmitted to the memory controller, that in turn accesses the off-chip memory and forward read data back to the cache.

Redundant encoding has been implemented on this communication channel, consisting of an encoder, a decoder and some additional code-dependent parity lines. The link has been provided with retransmission capability, preserving compliance with the AMBA protocol (see Section VI-A).

Encoder and decoder are powered at standard voltage levels, while voltage level translators allow wires to work at a reduced swing. This setup points out the energy efficiency of the codes under test and provides useful indications for SoC designers because it can be thought of as a shared communication resource or as a point-to-point connection in an NoC (e.g., between the network interface and a switch, or between two switches). For the purpose of our analysis, the major difference with respect to a multihop NoC scenario is that this latter makes use of data packetization, and this also affects the way retransmissions are carried out, as will be discussed in Section VII.

A. AMBA-Compliant Retransmissions

In this section, changes applied to the AMBA bus implementation are described. We show that retransmission based recovery procedures can be carried out while preserving compliance with the AMBA specification.

In AMBA infrastructures, a slave can indicate that the transfer in progress cannot be completed successfully. In this case, only higher priority masters will gain access to the bus, and the present master should continue to retry the transfer until it completes. This mechanism is referred to as the slave *retry* response, and is described in Fig. 7. In the penultimate cycle of a read transfer, the slave drives *HRESP* to indicate *RETRY*, while driving *READY* low to extend the transfer for an extra cycle. This signal configuration is sampled by the rising edge of the clock, and it causes the next cycle to be *IDLE*. During the idle cycle, *HRESP* is still set to *RETRY*, and this prevents the data on the bus from being sampled by both the cache and the integer unit of the Leon processor

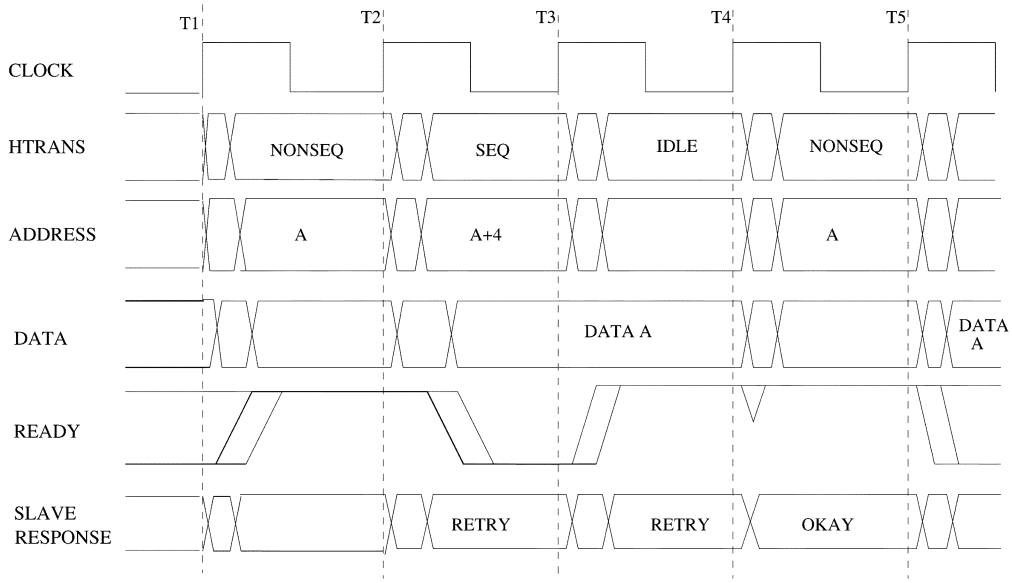

Fig. 7. Waveforms of the retry response on an AMBA bus, exploited for retransmissions. The slave response is given by means of the *HRESP* lines.

on the next rising edge of the clock. It also causes the processor to retry the transfer at the same memory location, provided the control of the bus is not lost.

This mechanism has been used to implement retransmissions of incorrectly received data. The assumption we make is that before the rising edge *T4* of the clock, which should sample data as the *READY* signal is high, the decoder evaluation has completed. This can be accounted by adding its delay (as well as the encoder delay) to the memory access time, and can result in one additional wait state. If the decoder response notifies errors, the master takes the same course of action that would be triggered by the *HRESP* signal in case of a slave retry response. That is, the internal signal allowing data sampling on *T4* at the cache and at the integer unit is not asserted, while another signal is activated, that forces the processor to repeat the last transfer. The activation time of these signals should be considered in the computation of the needed wait states as well.

In this implementation, the *HRESP* signal is ignored by the master, and the memory controller just serves two successive access requests to the same memory location. The entire mechanism is master-controlled. Note that because of the pipelined nature of the AMBA bus, by the time the master issues a retry procedure, then the address for the following transfer has already been broadcast onto the bus (see address $A+4$ in Fig. 7). This involves additional transitions on the address bus to restore the address of the data word to be retransmitted. Their impact on energy dissipation will be considered in the experimental results.

The timing overhead incurred by a data-word retransmission is one extra idle cycle plus those cycles associated with the repetition of the basic read/write transfer. This information could be used for a system-level latency analysis that, based on the knowledge of application quality of service constraints and of system and technology parameters (power of system components, bit error rate, etc.), could assess whether the retransmission overhead is tolerable or not. Such a system-level view goes beyond the scope of this paper, which focuses on the efficiency of the error control mechanism and not on its system level implications.

However, it is worth observing that deep submicron integrated circuits exhibit a lower degree of nondeterminism than other contexts (such as communication over wireless channels in wide area networks). For instance, if we assume a pessimistic bit error rate for on-chip communication of 10^{-6}, we have to deal with a corrupted transfer every 26 315 transferred codewords (for a 32-bit link and six parity bits). An error correcting decoder introduces a significant overhead (decoder delay and power) for each transfer, while an error detecting decoder scheduling retransmissions incurs a much smaller overhead at each transfer and an additional overhead of a few cycles only for corrupted transfers. If the retransmission penalty amounts to one idle cycle plus two cycles to repeat the AMBA transfer plus two wait states, this results in an overall overhead of 0.019% with respect to the total number of transfer cycles.

Finally, note that contrary to [9], in our implementation retransmissions are carried out by exploiting bus protocol features (AMBA retry response) and not adding retransmission request signals, for which full swing signaling should be used for reliability purposes. In fact, the assumption we are making here is that all control lines work at full swing (i.e., the swing of an unencoded bus). Only encoded (data) links are allowed to work at lower swings.

VII. Energy Efficiency Results

Given the requirement on communication reliability for the considered link, we can derive the maximum tolerable bus line flipping probability corresponding to that constraint [variable ϵ in (6)], as illustrated in Fig. 8. This was done by using simple probability theory for uncorrelated events. For example, recalling that SEC requires 38 lines (32 data bits and 6 check bits), the value of ϵ for SEC was derived from the following equation by replacing REP with the reliability requirement:

$$\mathrm{REP} = 1 - \sum_{i=0}^{1} \binom{38}{i} \epsilon_{\mathrm{SEC}}^{i} (1-\epsilon_{\mathrm{SEC}})^{38-i}. \qquad (8)$$

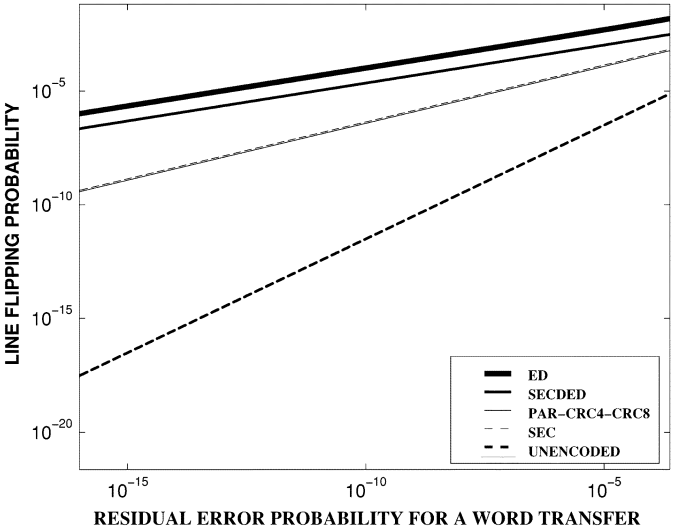

Fig. 8. Line flipping probability as a function of the requirement on communication reliability.

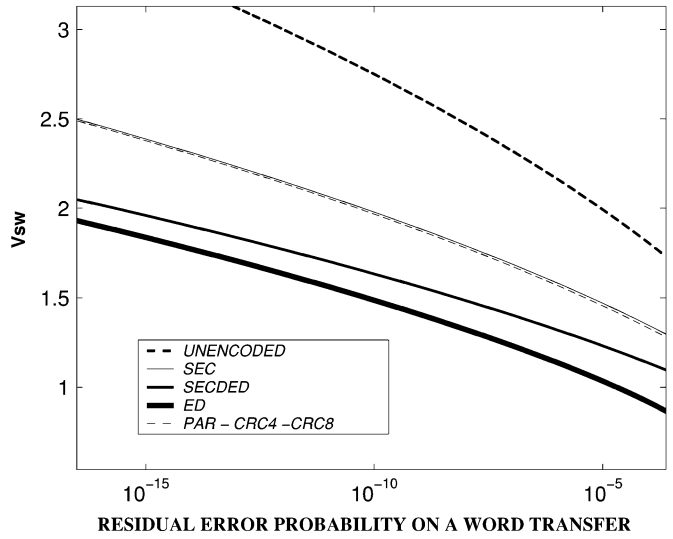

Fig. 9. Minimum voltage swing needed by each coding scheme to meet a predefined communication reliability requirement.

Note that $i = 0$ indicates the fault-free case while $i = 1$ the single error case, i.e., those cases wherein communication across the encoded bus does not fail. Fig. 8 points out that the higher the detection capability of a code, the higher its maximum tolerable error on bus lines.

By inverting (6) of the Hegde-Shanbhag model, we get the minimum wire voltage swing $V_{\rm sw}$ that each error control code has to use to meet the common reliability constraint, as reported in Fig. 9.

The unencoded link has of course to use the highest swing, while the other encoding schemes can rely on their error detection capability, independently of the error recovery action. Note that retransmission-based techniques, however, require lower swings than correction oriented ones, because they do not have to make restrictive assumptions on the nature of errors in order to be able to correct them. CRC codes have the same requirements as SEC and PAR, as they have the common characteristic of detecting single errors (having the highest occurrence probability with the used model) and not all double ones.

In general, the larger the detection capability of a coding scheme, the lower the voltage swing that can be used across interconnects. In fact, the lower SNR is counterbalanced by the ability of the decoder to detect a large number of error patterns and to take the proper course of recovery action.

The next step is to evaluate whether such a swing reduction is beneficial in terms of energy dissipation. The energy overhead associated with error recovery must not make up for the low-swing related savings.

The impact of error recovery techniques can be assessed by computing the energy efficiency metric for each code. Results in Fig. 10 refer to a bit line load capacitance C_L of 5 pF (a wire of about 1 cm in a 0.18-μm technology). This could be the case of a shared bus, wherein the growing number of connected cores in DSM technologies progressively increases the associated load capacitance.

Such a high capacitive load makes the energy cost for bus transitions dominant with respect to codec-related energy overhead. Retransmission-based strategies (ED, PAR, CRC) perform better than SEC because they can work at lower voltage swings thanks to their higher detection capability, and this makes the difference independent of the increased number of transitions on link lines. Note that SECDED gives satisfactory results, in that it uses a mixed approach: correction is used for single errors, retransmission for double ones. The left-most part of the graph is the one of interest, because it corresponds to a MTTF of about 1 year (bus clocked at 500 MHz). In the right-most part, MTTF is hundreds of nanoseconds.

Fig. 11 shows the same curves plotted for a C_L of 0.5 pF (a few millimeter long wires in a 0.18-μm technology). This could be the case of a switch-to-switch link of an on-chip micronetwork of interconnects (NoC), characterized by reduced length and with only two switches connected to it.

Here, transitions on the link lines play a minor role, while the contribution of codec complexity becomes relevant. This explains why the gap between SEC, SECDED, and the other schemes becomes more relevant: correction circuitry at the decoder side makes the difference. Among retransmission-oriented schemes, PAR outperforms ED, and CRC4 and CRC8 become competitive.

Since projections show that capacitance of global wires does not scale down with technology, we have plotted results over a wide range of interconnect capacitances to capture the trend as technology scales down. In fact, shrinking of geometries will make communication power dominant with respect to computation power. Under these assumptions, the effectiveness of bus encoding combined with low swing signaling is better described by Fig. 10. The proposed approach will increasingly contribute to energy-efficient communication, and although retransmission-based techniques are more effective in state-of-the-art implementations, the gap with error correction will be progressively bridged due to the cost for switching heavily loaded global interconnects.

On the other hand, communication architectures are evolving to evolutionary solutions (e.g., crossbars) and (in the long run) to

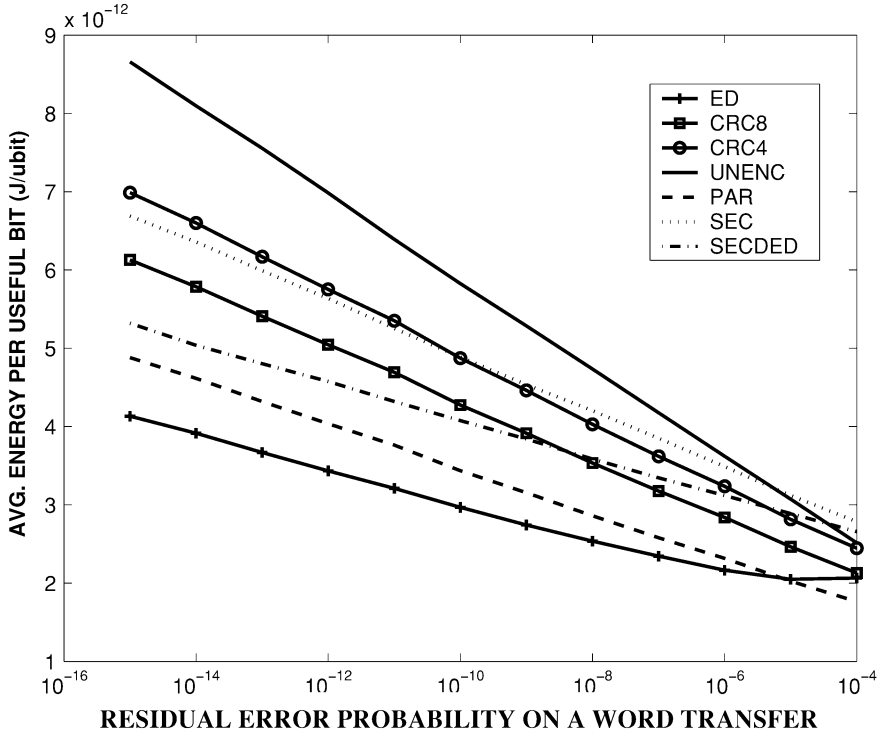

Fig. 10. Energy efficiency of coding schemes for on-chip communication links. Wire lengths are in the order of centimeters.

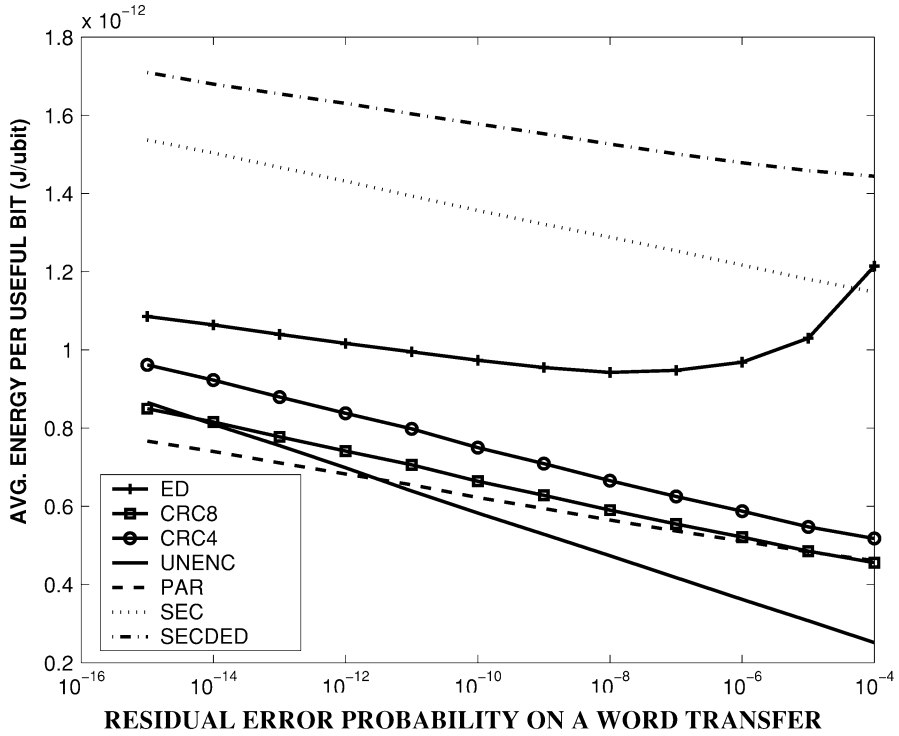

Fig. 11. Energy efficiency of coding schemes for on-chip communication links. Wire lengths are in the order of millimeters.

revolutionary NoC-based solutions. In general, advanced interconnects tend to have shorter point-to-point links, therefore, the effectiveness of our technique for low-power communication on advanced infrastructures can be better understood by looking at Fig. 11. Still, our approach will provide energy savings for MTTFs of at least one year. In this context, retransmissions will no doubt be an energy-efficient error recovery technique. Overall, depending on the particular technology and communication architecture topology, the point of interest in the investigated range of wire load capacitances can be selected.

The illustrated results also point out that the detection capability of a code plays a major role in determining its energy ef-

ficiency, because it is directly related to the wire voltage swing that can be used to ensure a predefined level of communication reliability. Moreover, error correction is beneficial in terms of delay penalty for data recovery, but has two main drawbacks: 1) it limits the detection capability of a code and 2) it makes use of high-complexity decoders. On the contrary, when the higher recovery delay of retransmission mechanisms can be tolerated, they provide a higher energy efficiency, thanks to the lower swings and simpler codecs (pure error detecting circuits) they can use while preserving communication reliability. Mixed approaches such as SECDED could be a tradeoff solution. Further work at the system level needs to be done to confirm these results, for instance accounting for the energy dissipation of system components during retransmission cycles. However, if the bit error rate can be kept low by means of physical design techniques, we envision that such a retransmission overhead can be considered almost negligible.

Finally, these results are referred to the selected error control codes (Hamming and CRC), which we consider of interest for on-chip realizations. Although these codes represent a very small subset of the code space, their characterization suggests that it is a nontrivial task to find an error correcting code that achieves a residual error rate comparable to that of an error detecting code with the same level of complexity, thus allowing the use of the same wire voltage swing. Even though this were possible, if the bit error rate for on-chip communication is low, error correcting decoders significantly degrade performance and energy efficiency at each transfer, as opposed to retransmission-based coding schemes that only occasionally affect on-chip communication in a significant way. However, the overall performance degradation as perceived from an application is likely to be marginal for this latter case, unless very high bit error rates occur.

VIII. Further Developments: Multihop NoCs

As SoC communication architectures evolve from shared buses to micronetworks of interconnects, the relevant issue of packetization comes to the forefront, and additional parameters have to be considered to estimate energy efficiency of error recovery techniques. In general, each message to be transmitted is partitioned into fixed-length *packets*. Packets in turn are often broken into message flow control units or *flits*. In the presence of channel width constraints, multiple physical channel cycles may be used to transfer a single flit. A *phit* is the unit of information that can be transferred across a physical channel in a single step or cycle. Flits represent logical units of information, as opposed to phits which correspond to physical quantities, i.e., the number of bits that can be transferred in a single cycle. In many implementations, a flit is set to be equal to a phit.

Communication reliability can be guaranteed at different levels of granularity. We might refer control bits (i.e., a checksum) to an entire packet, thus minimizing control bits overhead. Though this would prevent us from stopping the propagation of corrupted flits, as routing decisions would be taken in advance with respect to the data integrity check. In fact, control bits would be transmitted as the last flit of the packet. In this scenario, the cost for error control would be paid in the time domain. The alternative solution is to provide reliability at the flit level, thus refining control granularity but paying for redundancy in the space domain (additional wiring resources for check bits).

The considerations that follow will be referred to this latter scenario, wherein two different solutions are viable.

1) The error recovery strategy can be *distributed* over the network. Each communication switch is equipped with error detecting/correcting circuitry, so that error propagation can be immediately stopped. This is the only way to avoid routing errors: Should the header get corrupted, its correct bit configuration could be immediately restored, preventing the packet from being forwarded across the wrong path to the wrong destination. Unfortunately, retransmission-oriented schemes need power-hungry buffering resources at each switch, so their advantage in terms of higher detection capability has to be paid with circuit complexity and power dissipation.
2) Alternatively, an *end-to-end* approach to error recovery is feasible: Only end-nodes are able to perform error detection/correction. In this case, retransmission may not be convenient at all, especially when source and destination are far apart from each other, and retransmitting corrupted packets would stimulate a large number of transitions, beyond giving rise to large delays. For this scenario, error correction is the most efficient solution, even though proper course of action has to be taken to handle incorrectly routed packets (retransmission time-outs at the source node, deadlock avoidance, etc.).

Another consideration regards the way retransmissions are carried out in a NoC. Traditional shared bus architectures can be modified to perform retransmissions in a "stop and wait" fashion: the master drives the data bus and waits for the slave to carry out sampling on one of the following clock edges. If the slave detects corrupted data, a feedback has to be given back to the master, scheduling a retransmission. To this purpose, an additional feedback line can be used, or built-in mechanisms of the bus protocol can be exploited (like the approach of this paper). In packetized networks, data packets transmitted by the master can be seen as a continuous flow, so the retransmission mechanism must be either "go-back-N" or "selective repeat". In both cases, each packet (or flit) has to be acknowledged (ACK), and the difference lies in the receiver (switch or network interface) complexity. In a "go-back-N" scheme, the receiver sends a not ACK (NACK) to the sender relative to a certain incorrectly received packet. The sender reacts by retransmitting the corrupted packet as well as all other following packets in the data flow. This alleviates the receiver from the burden to store packets received out of order and to reconstruct the original sequence.

On the contrary, when this capability is available at the receiver side (at the cost of further complexity), retransmissions can be carried out by selectively requiring the corrupted packet without the need to retransmit also successive packets. The tradeoff here is between switch and network interface complexity and number of transitions on the link lines.

IX. Conclusion

In this paper, the energy–reliability tradeoff has been investigated for the basic building blocks of SoC communication channels, providing SoC designers with guidelines for the selection of energy efficient error control schemes. We provide quantitative results about the energy cost that has to be sustained to guarantee predefined MTTFs in an on-chip point-to-point link. Implementation issues have been discussed with reference to a real SoC scenario.

Specific contributions of this paper can be described as follows.

1) Error-control coding can significantly enhance communication reliability while reducing energy-per-bit dissipation. The energy overhead introduced by redundant parity lines is counterbalanced by the reduced voltage swings.
2) The optimal encoding scheme for a point-to-point link is not unique but depends on link load conditions and therefore on technology and communication architecture.
3) With state of the art technology and for the considered subset of codes (Hamming and CRC), retransmission turns out to be more efficient than correction from an energy viewpoint. However, as an effect of the IC scaling scenario, communication energy is likely to largely overcome computational energy, and for very DSM technologies error correction might bridge the gap.

Interesting research and development opportunities lie ahead: Future NoCs could implement error correction and error detection at various levels of the communication stack, using dedicated hardware, even coupled with application-level software support. In general, communication reliability is likely to become a top-level design constraint. and its relationships with energy efficiency will become one of the most promising exploration directions in future designs.

References

[1] K. Aingaran, F. Klass, C. M. Kim, C. Amir, J. Mitra, E. You, J. Mohd, and S. K. Dong, "Coupling noise analysis for VLSI and ULSI circuits," in *Proc. IEEE ISQED*, Mar. 2000, pp. 485–489.

[2] L. Anghel and M. Nicolaidis, "Cost reduction and evaluation of temporary faults detecting technique," in *Proc. DATE*, Mar. 2000, pp. 591–598.

[3] L. Benini, A. Macii, E. Macii, M. Poncino, and R. Scarsi, "Architectures and synthesis algorithm for power efficient bus interfaces," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 19, no. 9, pp. 969–980, Sep. 2000.

[4] L. Benini, P. Siegel, and G. De Micheli, "Automatic synthesis of gated clocks for power reduction in sequential circuits," *IEEE Design Test Comput.*, no. 6, pp. 32–40, Dec. 1994.

[5] D. Bertozzi, L. Benini, and G. De Micheli, "Low power error resilient encoding for on-chip data buses," *Proc. DATE*, pp. 102–109, Mar. 2002.

[6] D. Bertozzi, L. Benini, and B. Ricco', "Energy-efficient and reliable low-swing signaling for on-chip buses based on redundant encoding," in *Proc. ISCAS*, vol. I, May 2002, pp. 93–96.

[7] D. Das and N. Touba, "Weight-based codes and their applications to concurrent error detection of multilevel circuits," in *Proc. IEEE VLSI Test Symp.*, 1999, pp. 370–376.

[8] M. Favalli and C. Metra, "Optimization of error detecting codes for the detection of crosstalk originated errors," in *Proc. DATE*, Mar. 2001, pp. 290–296.

[9] R. Hegde and N. R. Shanbhag, "Toward achieving energy efficiency in presence of deep submicron noise," *IEEE Trans. VLSI Syst.*, vol. 8, no. 4, pp. 379–391, Aug. 2000.

[10] R. Ho, K. W. Mai, and M. A. Horowitz, "The future of wires," *Proc. IEEE*, vol. 89, no. 4, pp. 490–504, Apr. 2001.

[11] I. Hyunsik, T. Inukai, H. Gomyo, T. Hiramoto, and T. Sakurai, "VTCMOS characteristics and its optimum conditions predicted by a compact analytical model," in *Proc. ISLPED*, Aug. 2001, pp. 123–128.

[12] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1987, pp. 50–58.

[13] Int. Technology Roadmap for Semiconductors (2001). [Online]. Available: http://public.itrs.net/files/2001itrs/home.htm

[14] C. Kim and K. Roy, "Dynamic vth scaling scheme for active leakage power reduction," in *Proc. DATE*, Mar. 2002, pp. 163–167.

[15] C. Grecu, P. P. Pande, A. Ivanov, and R. Saleh, "A scalable communication-centric SoC interconnect architecture," in *Proc. Int. Symp. Quality Electron. Design*, Mar. 2004, pp. 343–348.

[16] K. Kim, K. Baek, N. Shanbhag, C. Liu, and S. Kang, "Coupling-driven signal encoding scheme for low-power interface design," *Proc. IEEE/ACM ICCAD*, pp. 318–321, Nov. 2000.

[17] A. Krstic, Y. M. Jiang, and K. T. Cheng, "Pattern generation for delay testing and dynamic timing analysis considering power-supply noise effects," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 20, no. 3, pp. 416–425, Mar. 2001.

[18] P. Lala, *Self-Checking and Fault-Tolerant Digital Design*. San Francisco, CA: Morgan Kaufmann, 2000.

[19] Gaisler Research Website [Online]. Available: http://www.gaisler.com

[20] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1983.

[21] B. Luca and D. M. Giovanni, "Networks on chips: A new SOC paradigm," *IEEE Comput.*, vol. 35, no. 1, pp. 70–78, Jan. 2002.

[22] J. Mazo and B. Saltzberg, "Error-burst detection with tandem CRC's," *IEEE Trans. Commun.*, vol. 39, no. 8, pp. 1175–1178, Aug. 1991.

[23] C. Metra and M. Favalli, "Bus crosstalk fault-detection capabilities of error-detecting codes for on-line testing," *IEEE Trans. VLSI Syst.*, vol. 7, no. 3, pp. 392–396, Sep. 1999.

[24] S. Mitra and E. McCluskey, "Word-voter: A new voter design for triple modular redundant systems," in *Proc. IEEE VLSI Test Symp.*, May 2000, pp. 465–470.

[25] E. Musoll, T. Lang, and J. Cortadella, "Working-zone encoding for reducing the energy in microprocessor address bus," *IEEE Trans. VLSI Syst.*, vol. 6, no. 4, pp. 568–572, Dec. 1998.

[26] A. Popplewell, J. O'Reilly, and S. Williams, "Architectures for fast encoding and error detection of cyclic codes," *Proc. IEEE*, vol. 139, no. 3, pp. 340–348, Jun. 1992.

[27] D. Pradhan, *Fault Tolerant Computing: Theory and Techniques*. Englewood Cliffs, NJ: Prentice-Hall, 1986.

[28] G. Qu, "What is the limit of energy saving by dynamic voltage scaling?," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, 2001, pp. 560–563.

[29] S. Roman, *Introduction to Coding and Information Theory*. New York: Springer Verlag, 1997.

[30] N. Shang and P. Li Shiuan, "Power-efficient interconnection networks: Dynamic voltage scaling with links," *Comput. Archit. Lett.*, vol. 1, no. 2, pp. 1–4, 2002.

[31] K. L. Shepard and V. Narayanan, "Noise in deep submicron digital design," in *Proc. IEEE/ACM ICCAD*, Nov. 1996, pp. 524–531.

[32] D. Siewiorek and R. Swarz, *Reliable Computer Systems: Design and Evaluation*. Natick, MA: A. K. Peters, Oct. 1998.

[33] A. Sobski and A. Albicki, "Partitioned and parallel cyclic redundancy checking," *Proc. 36th Symp. Circuits Syst.*, vol. 1, pp. 538–541, Aug. 1993.

[34] M. Stan and W. Burleson, "Bus-invert coding for low-power i/o," *IEEE Trans. VLSI Syst.*, vol. 3, no. 1, pp. 49–58, Mar. 1995.

[35] C. Svensson, "Optimum voltage swing on on-chip and off-chip interconnect," *IEEE J. Solid State Circuits*, vol. 36, no. 7, pp. 1109–1112, Jul. 2001.

[36] J. F. Wakerly, *Digital Design: Principles and Practices*, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 2000.

[37] A. Wang, A. Chandrakasan, and S. Kosonocky, "Optimal supply and threshold scaling for subthreshold CMOS circuits," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI-ISVLS*, 2002, pp. 5–9.

[38] F. Worm, P. Ienne, P. Thiran, and G. De Micheli, "An adaptive low-power transmission scheme for on-chip networks," in *Proc. Int. Symp. Syst. Synthesis*, 2002, pp. 92–100.

[39] H. Zhang, V. George, and J. M. Rabaey, "Low swing on-chip signaling techniques: Effectiveness and robustness," *IEEE Trans. VLSI Syst.*, vol. 8, no. 3, pp. 264–272, Jun. 2000.

[40] M. Zorzi, "Some results on error control for burst-error channels under delay constraints," *IEEE Trans. Veh. Technol.*, vol. 50, no. 1, pp. 12–24, Jan. 2001.

Davide Bertozzi received the electrical engineering and Ph.D. degrees from the University of Bologna, Bologna, Italy, in 1999 and 2003, respectively.

He currently holds a Postdoctorate position in the Department of Electrical Engineering, University of Bologna, and has held several Visiting Researcher positions at international universities (Stanford University) and semiconductor industries (NEC Labs. America, Philips Labs, Samsung Electronics, and STMicroelectronics). His main research interests concern system-level design and communication architectures for multiprocessor systems on chip, devoting special attention to the power, performance, and reliability tradeoff.

Luca Benini (S'94–M'97–SM'04) received the B.S. degree (*summa cum laude*) in electrical engineering from the University of Bologna, Bologna, Italy, in 1991 and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1994 and 1997, respectively.

Currently, he is an Associate Professor in the Department of Electronics and Computer Science, University of Bologna. He also holds Visiting Researcher positions at Stanford University and Hewlett-Packard Laboratories, Palo Alto, CA. He has published more than 200 papers in international journals and conferences, two books, and several book chapters. His research interests are in all aspects of computer-aided design of digital circuits, with special emphasis on low-power applications, and in the design of portable systems.

Prof. Benini is the Technical Program Chair of the IEEE Design Automation Test in Europe 2004 Conference. He is a Member of the organizing committee of the International Symposium on Low Power Electronics and Design. He is a Member of the Technical Program Committee for several technical conferences, including DAC, DATE, ISLPED, and ISSS-CODES.

Giovanni De Micheli (S'79–M'79–SM'80–F'94) received the nuclear engineer degree from Politecnico di Milano, Milan, Italy, in 1979 and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley, in 1980 and 1983, respectively.

He is a Professor of electrical engineering and, by courtesy, of computer science at Stanford University, Stanford, CA. Previously, he held positions at the IBM T. J. Watson Research Center, Yorktown Heights, NY, in the Department of Electronics of the Politecnico di Milano, Milan, Italy, and at Harris Semiconductor, Melbourne, FL. He was a founding member of the ALaRI Institute at Universita' della Svizzera Italiana (USI), Lugano, Switzerland, where he is currently a Scientific Counselor. He is the author of *Synthesis and Optimization of Digital Circuits* (New York: McGraw–Hill, 1994) and coauthor and/or coeditor of five other books and of over 270 technical articles. His research interests include several aspects of design technologies for integrated circuits and systems, with particular emphasis on synthesis, system-level design, hardware/software codesign, and low-power design.

Prof. De Micheli is a Fellow of the ACM. He received the 2003 IEEE Emanuel Priore Award for contributions to computer-aided synthesis of digital systems. He received the Golden Jubilee Medal for outstanding contributions to the IEEE CAS Society in 2000. He received the 1987 IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS Best Paper Award and two Best Paper Awards at the Design Automation Conference, in 1983 and 1993. He was the Program and General Chair of the Design Automation Conference (DAC) from 1996 to 1997 and 2000, respectively. He was the Program and General Chair of the International Conference on Computer Design (ICCD) in 1988 and 1989, respectively. He is a Past President of the IEEE Circuits and Systems Society. He was the Editor in Chief of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS from 1987 to 2001.