An Information Theoretical Framework for Analysis and Design of Nanoscale Fault-Tolerant Memories Based on Low-Density Parity-Check Codes

Bane Vasic, *Senior Member, IEEE*, and Shashi Kiran Chilappagari, *Student Member, IEEE*

***Abstract*—In this paper, we develop a theoretical framework for the analysis and design of fault-tolerant memory architectures. Our approach is a modification of the method developed by Taylor and refined by Kuznetsov. Taylor and Kuznetsov (TK) showed that memory systems have nonzero computational (storage) capacity, i.e., the redundancy necessary to ensure reliability grows asymptotically linearly with the memory size. The restoration phase in the TK method is based on low-density parity-check codes which can be decoded using low complexity decoders. The equivalence of the restoration phase in the TK method and faulty Gallager B algorithm enabled us to establish a theoretical framework for solving problems in reliable storage on unreliable media using the large body of knowledge in codes on graphs and iterative decoding gained in the past decade.**

***Index Terms*—Faulty gates, reliable storage, low-density parity-check (LDPC) codes, message passing.**

I. Introduction

DURING the past four decades, the decrease in transistor size and the increase in integration factor have led to very small, fast, and power efficient chips. As the demand for power efficiency continues, a wide range of new nanoscale technologies (see [1]–[5]) is being actively investigated for processing and storage of digital data. Although it is difficult to discern which of these approaches will become a technological basis for computers in the future, it is widely recognized that due to their miniature size and variations in technological process, the nanocomponents will be inherently unreliable. Even in more traditional semiconductor technologies, reducing transistor size has already started affecting circuit reliability, and it is widely believed that transistor failures (both transient and permanent) will become one of the main technological obstacles as the trend of increasing the integration factor continues.

Traditional architectures that ensure fault tolerance are incapable of handling such increased unreliability, and with alternative nanostorage technologies on the industry roadmaps, the development of novel reliable computers is of critical importance. Generally, the unreliability problem can be solved on a device level, circuit level or on a system level. In the first two approaches the devices or circuits are made more reliable by the improved technological process or circuit design. Such approaches are not in the spirit of information theory. In this paper, we consider a different approach. We will assume that devices and circuits have finite reliability that cannot be improved, and our goal is to design a system of maximal possible reliability by using redundant hardware.

We give a theoretical framework for highly reliable memories and logic gates made of unreliable components and use it to characterize such devices in terms of their complexity and reliability. The main challenge is that in nanoscale systems both the storage elements and logic gates are faulty. It is in contrast to the state-of-the-art systems where only the memory elements are considered unreliable while logic gates are assumed to be reliable.

This unique feature of the nanosystems makes the problem of ensuring fault tolerance theoretically very important, because the process of error correction, which is performed by logic gates, is not error free as assumed in classical information theory [6]. Classical coding theory guarantees that the probability of decoding error during transmission or storage of information can be made arbitrarily small by increasing the strength of the error correcting code, i.e., by using very long codes. It can be shown that a tolerance to errors in memory systems can be achieved by linear increase in redundancy while the cost of ensuring reliability of logic gates is log-linear [7]. This is possible because in traditional models of memory and communication systems with error correction coding, it is assumed that the operations of an error correction encoder and decoder are deterministic and that the randomness (in the form of noise and/or errors) exists only in the channel. However, if digital logic in the encoder and decoder is built of faulty components (devices), then the errors and noise do effect the operations performed, and reduce the reliability of the whole system. Such encoders and decoders are referred to as noisy or faulty. Making error correcting codes stronger and transmitters and receivers more complex will not necessarily improve the performance of a system. It is likely that for a given failure mechanism, there is a trade off between receiver complexity and its performance.

The above problem can be condensed into the following question: given n memory cells and m universal logic gates which fail following a known random mechanism, what is the optimal memory and logic gate architecture which stores/processes the maximum number of information bits with arbitrary low probability of error? This complex problem can be divided and reformulated in many ways, but, interestingly, even some of the most

Manuscript received January 20, 2007. This work was supported by the National Science Foundation under Grant CCF-0634969. This paper was recommended by Guest Editor C. Lau.

The authors are with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721 USA (e-mail: vasic@ece.arizona.edu; shashic@ece.arizona.edu).

fundamental questions related to this problem are still unanswered.

The question that will be addressed in this paper is related to the following two fundamentally different approaches to ensure fault tolerance.

1) To improve reliability, the logic gate resources may be invested into a von Neumann multiplexing scheme. In this way, one can build highly redundant reliable networks that simulate the function of universal logic gates, and then use such better gates to build an error correction encoder and decoder.
2) Alternatively, the logic gate resources may be invested into building a more powerful error correcting code (i.e., decoder) capable of handling both memory elements as well as logic gates errors.

Which of these two approaches is optimal for a given failure mechanism? The purpose of this paper is to establish a theoretical framework for answering the above questions using results from modern coding theory, and to quantify advantages of the error correction approach.

The paper is organized as follows. Section II summarizes the state of the theoretical research in fault tolerance of memories and logic gates on a system level, and describes the approaches of von Neumann and Taylor and Kuznetsov. Section III gives a brief background on the failure model used in the paper. Section IV gives a brief introduction to low-density parity-check (LDPC) codes and the Gallager B algorithm. In Section V, we establish a connection between the Gallager B algorithm and the approach of Taylor and Kuznetsov. We then set up a framework for the analysis of faulty decoders in Section VI and present simulation results in Section VII. In Section VIII, we point out a few open problems and discuss the direction of future work and we conclude in Section IX.

II. Approaches of Von Neumann and Taylor and Kuznetsov

A. Multiplexing

The conventional approach to deal with faulty gates, in general, is to make them reliable by multiplexing. (Reliability is defined as the likelihood of a circuit to produce a desired output). The multiplexing technique originates from von Neumann's theory of computation by circuits with faulty gates [8]. Von Neumann showed that, under certain conditions, increased gate redundancy may lead to increased circuit reliability. His fault-tolerant circuit was comprised of the so-called computational organ consisting of many replicas of faulty gates and a restoration organ that performs a majority vote. Von Neumann's multiplexing technique and especially one of its variants known as triple modular redundancy (TMR) have been extensively studied in the past, and recent advances in microelectronics and the need for stronger signal protection in digital circuits have returned it to the focus of many research groups (see e.g., [9]–[12], [5], [13]). All fault-tolerant nano-architectures proposed in the literature so far are based on von Neumann's multiplexing (see e.g., [14], [15], [1], [16]). The attractiveness of multiplexing is its simplicity. Its shortcoming is that it typically leads to extremely large, sometimes prohibitively large, hardware redundancy.

B. Taylor and Kuznetsov Scheme

Taylor [17], [18] was the first to investigate the capacity and fault-tolerant architectures of storage systems built entirely from unreliable components. His results were refined by Kuznetsov [19]. The aim of Taylor and Kuznetsov was to derive results analogous to the ones derived by Shannon on the capacity of communication systems. In the Taylor–Kuznetsov (TK) model, the information is stored in coded form, i.e., the stored vector is a codeword of some (n, k) block code (precise definitions of coding concepts will be given in Section IV). The registers in which the bits are stored are assumed to be unreliable. These registers are connected to a correcting circuit (a restoration organ in von Neumann's terminology), which periodically updates the values in the registers according to some decoding scheme. The correcting circuit is also assumed to be built from faulty components.

TK showed that faulty memory systems have nonzero computational capacity (or rather storage capacity). Roughly, the computational capacity of a Boolean function is the inverse ratio of the minimal number of faulty gates needed to evaluate the function with an arbitrary small probability of error to the number of perfect gates needed to obtain a reliable result. In other words, the redundancy necessary to ensure reliability of a memory grows asymptotically linearly with the memory size. This is in contrast to general computing systems made of faulty gates, which require redundancy asymptotically log-linear in number of inputs, and therefore have zero computational capacity [7]. Taylor also proved that no decoding scheme other than iterative LDPC decoding can achieve nonzero capacity, and derived bounds on memory failure probability under certain assumptions (similar to those used by Gallager in [20]). In the TK scheme, a memory failure is said to occur if the contents in the registers correspond to an error pattern uncorrectable by a noiseless decoder in I iterations, where I is the number of independent iterations in an iterative decoder.

C. Previous Work on LDPC Codes

LDPC codes have been the focus of much research over the past decade. Proposed by Gallager [20] in 1960s, they were largely forgotten until they were rediscovered by MacKay [21]. LDPC codes belong to a class of codes which can be described by graphs. One of the key results in codes on graphs comes from Kschischang *et al.* [22] who observed that iterative decoding algorithms developed for these codes are instances of probability propagation algorithms that operate in a graphical model of the code. The belief propagation algorithms and graphical models have been developed in the expert systems literature, by Pearl [23] and Lauritzen [24]. MacKay and Neal [25], and McEliece *et al.* [26] showed that Gallager's algorithm [20] for decoding LDPC codes proposed in early sixties is essentially an instance of Pearl's algorithm. Extensive simulation results of MacKay and Neal showed that Gallager codes could perform nearly as well as earlier developed Turbo codes [27]. They have observed that Turbo

decoding is an instance of "belief" propagation and provided a description of Pearl's algorithm, as well as made explicit the connection to the basic Turbo decoding algorithm described in [27]. Wiberg in [28] showed that graphs introduced by Tanner [29] twenty years ago to describe a generalization of Gallager codes provide a natural setting to describe and study iterative soft-decision decoding techniques. Forney [30] generalized Wiberg's results and explained connections of various two-way propagation algorithms with coding theory. Richardson and Urbanke [31] have analyzed the performance of ensemble of codes using density evolution. Richardson *et al.*, [32] gave methods for the design of capacity approaching irregular LDPC codes.

The current state of theoretical research in systems made of unreliable components can be compared to that in the area of communications before the renaissance of LDPC codes and iterative decoding algorithms [21]. The spirit and methodology of TK's work [17], [19] is similar to Gallager's results [20] on LDPC codes. The bounds on probability of error are given for an ensemble of regular random LDPC codes of infinitely large length used in the restoration organ. They are obtained under the assumptions that the bits in memory cells remain independent during the process of restoration, i.e., under the assumption that the girth (the length of the shortest cycle) of the Tanner graph (precise definition is given in Section IV) corresponding to a code is infinitely large. The situation is similar on the code design front. Although several authors, most notably Pippenger [7], suggested the potential of structured LDPC codes, there was not much progress in this direction, and the research in LDPC codes for faulty memories has not moved further from Margulis codes [33] and codes based on Ramanujan graphs [34]. The large body of knowledge in iterative decoding gained in past decade, especially techniques developed for code construction and optimization as well as techniques for analysis of iterative decoding algorithms have not been exploited so far to improve reliability of memory systems. Bridging this gap is precisely the main goal of this paper.

At the end of this section, we would like to point out that in information theory community, the TK result has hardly been recognized as fully as it deserves. It is also worth noting that Spielman obtained the best result for a general model of computation [35], by marrying the ideas of von Neumann with Reed–Solomon (RS) codes. Spielman's scheme has logarithmic redundancy (in terms of code length), and has zero computational capacity. An excellent overview of the state of research in fault tolerant computation is given by Pippenger [7].

III. Failure Model

In both von Neumann and TK models, a faulty component, generally a logic gate or a memory element is subject to transient faults, i.e., faults that manifest themselves at particular time steps but do not necessarily persist for later times [36]. It is also assumed that gates fail independently of each other, and that the defects are not permanent, i.e., a gate that malfunctioned at some point in time may give correct output subsequently and that failure occurs by flipping the correct result with some probability, i.e., if the correct result is "1," the gate gives "0" and vice versa. Such failure mechanism is referred to as von Neumann type of error. We note that the probability of "1" flipping to "0" and the probability of "0" flipping to "1" do not need to be the same, i.e., the failure can be input dependent. We also note that this model can readily be extended to nonbinary logic gates [37], [38] and memory elements [39]–[41]. From a theoretical point of view, both von Neumann type of error as well as the transient error models assuming dependent gate failures [42] fall into the class of Pippenger's ϵ-admissible failure model [9]. Notice that the failure of the connection between gates may be incorporated into the error of the gate at the end of connection, as proposed by Dobrushin and Ortyukov in [43]. It should be noted that transient faults in logic gates and memory elements can be modeled as a binary channel where a fault results in the flipping of the output of a logic gate or content of register from "0" to "1" and vice versa. A well known simple channel model is the binary symmetric channel (BSC) in which the probability of flipping from "0" to "1" is same as the probability of flipping from "1" to "0." This probability of error is generally denoted as α.

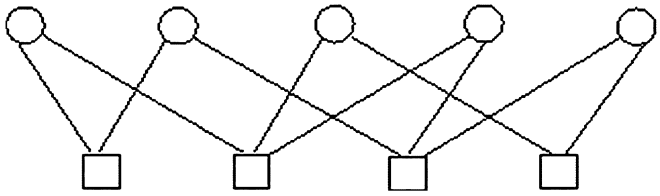


Fig. 1. Tanner graph.

IV. Introduction to LDPC Codes

Let $\mathcal{C}$ be an LDPC code of length n and dimension (the number of information bits) k. Let H, an $(m \times n)$ matrix, represent a parity check matrix of $\mathcal{C}$. The Tanner graph of an LDPC code, $\mathcal{G}$, is a bipartite graph with two sets of nodes: variable (bit) nodes and check (constraint) nodes. The check nodes (variable nodes) connected to a variable node (check node) are referred to as its neighbors. The degree of a node is the number of its neighbors. In a (γ, ρ) regular LDPC code, each variable node has degree γ and each check node has degree ρ. The girth g is the length of the shortest cycle in $\mathcal{G}$. A check node is said to be satisfied if the sum of all incoming messages has even parity and unsatisfied otherwise. Fig. 1 illustrates a simple Tanner graph with five variable nodes and four check nodes. The corresponding H matrix is given by

$$H = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

The Gallager B algorithm on the the BSC operates by passing binary messages along the edges of the Tanner graph of an LDPC code. Every round of message passing (iteration) starts with sending messages from variable nodes and ends by sending messages from check nodes to variable nodes. For a variable

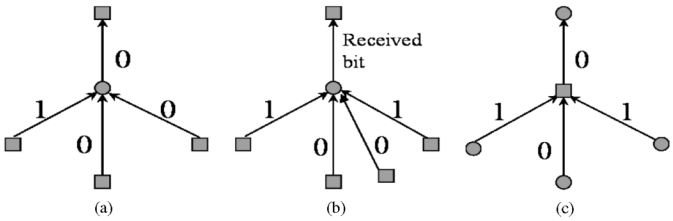


Fig. 2. Illustration of message passing. (a) Variable to check message. (b) Variable to check message in case of tie. (c) Check to variable message.

node v (check node c), let $E(v)$ $(E(c))$ denote the edges incident on v (c). Also, let $r(v)$ denote the received value of node v. Let the degree of a variable node be j. A threshold $b_{i,j}$ is determined for every iteration i and variable degree j. Let $\overrightarrow{m_i}\ (e)$ and $\overleftarrow{m_i}\ (e)$ represent the messages passed on an edge e from variable node to check node and check node to variable node in iteration i respectively. Then for each node v, the Gallager B algorithm passes the following messages in ith iteration.

$$\overrightarrow{m_i}\ (e) = r(v), \qquad i = 1.$$

For $i > 1$

$$\overrightarrow{m_i}\ (e) = \begin{cases} 1, & \sum_{e' \in \widetilde{E_e(v)}} \overleftarrow{m_{i-1}}\ (e') \geq b_{i,j} \\ 0, & \sum_{e' \in \widetilde{E_e(v)}} \overleftarrow{m_{i-1}}\ (e') \leq j - 1 - b_{i,j} \\ r(v), & \text{otherwise} \end{cases}$$

where $\widetilde{E_e(v)} = E(v) \backslash \{e\}$.

For each check node c, the messages passed in ith iteration are

$$\overleftarrow{m_i}\ (e) = \left(\sum_{e' \in E(c) \backslash \{e\}} \overrightarrow{m_i}\ (e') \right) \bmod 2.$$

Fig. 2 illustrates the message passing rules for different cases. For more details on iterative decoding on Tanner graphs see [44].

V. Proposed Method

As mentioned earlier, the basis of our approach is the TK method [17], [19]. This section explains the main idea, namely the equivalence of the restoration phase in the TK method and a "faulty" Gallager B algorithm. This equivalence will enable us to design and analyze memories using methodology developed for codes on graphs and iterative decoding. First, we describe the restoration process in a basic TK scheme and in a modified scheme that we propose, and then we establish the relation of the modified TK scheme with the Gallager B algorithm for iterative decoding [20].

In the TK scheme [17], [19], the information to be stored is first encoded by a regular binary LDPC code of length n and dimension k. The stored codeword $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ consist of bits $x_i, 1 \leq i \leq n$ referred also as variables. Each variable bit $x_i, 1 \leq i \leq n$ is involved in J parity-check equations by $\mathbf{x}H^T = \mathbf{c}$, where H is an $(m \times n)$ parity check matrix and all operations are in binary field. The degree of each check node is K. The vector $\mathbf{c} = (c_1, c_2, \ldots, c_m)$ is called syndrome and c_j corresponds to the value of jth parity-check sum for $1 \leq j \leq m$. Parity check c_j is said to be satisfied if $c_j = 0$ and unsatisfied if $c_j = 1$. A set of parity checks involving bit x_i is $\{c_i^{(1)}, c_i^{(2)}, \ldots, c_i^{(J)}\}$. After encoding, every coded bit x_i is replaced with J bit-copies of itself $\{x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(J)}\}$ and stored in J registers (Fig. 3). All bit-copies initially have the same value.

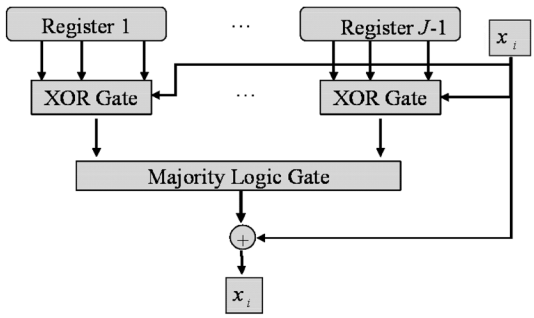


Fig. 3. TK memory.

Registers are made from memory cells that are considered to be unreliable. New estimates of each of these copies are obtained by using one combination of $J-1$ checks. Note that there are exactly $\binom{J}{J-1} = J$ combinations. The estimates are obtained as follows.

1) Evaluate parity checks for each bit-copy (exclude one distinct parity check from the original set of checks for each bit-copy).
2) Flip the value of a particular bit-copy if half or more of the parity checks are unsatisfied.
3) Iterate (1) and (2).

There are a total of $J-1$ parity checks for each bit copy. The decision element in this case is a *majority logic* gate whose output is 1 if half or more of the parity checks are nonzero. The correction is accomplished by XOR gate that has as inputs the output of the majority logic gate and the previous value of the bit-copy. Note that in this iterative decoding scheme, each parity check requires $J-1$ input bits (other than the bit-copy we are trying to estimate). Since each of these input bits has J different copies, there are J copies for each particular bit that can be used for estimating $x_i^{(j)}$, jth copy of bit x_i. When estimating a copy $x_i^{(j)}$ of bit x_i using an estimate of bit x_l, we use the bit-copy $x_l^{(s)}$ in whose evaluation the parity check involving x_i is omitted.

The above explanation gives rise to defining the whole scheme as a modified Gallager B decoder [20]. As explained in Section IV, the Gallager B algorithm on the BSC operates by passing binary messages along the edges of the Tanner graph of an LDPC code. Every round of message passing (iteration) starts with sending messages from variable nodes and ends by sending messages from check nodes to variable nodes. In the modified scheme, the J estimates of a particular bit x_i correspond to the outgoing messages along the edges of the Tanner graph incident on x_i (there are J such edges). This is

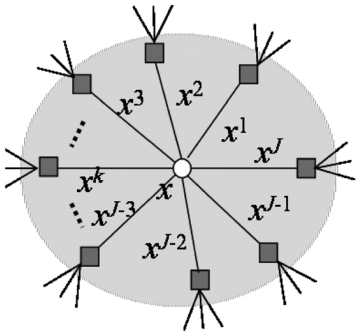


Fig. 4. Representation of the TK method as a faulty Gallager B algorithm. $\{x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(J)}\}$, the J copies of bit x_i, are assigned to the edges emanating from a variable node corresponding to the bit x_i.

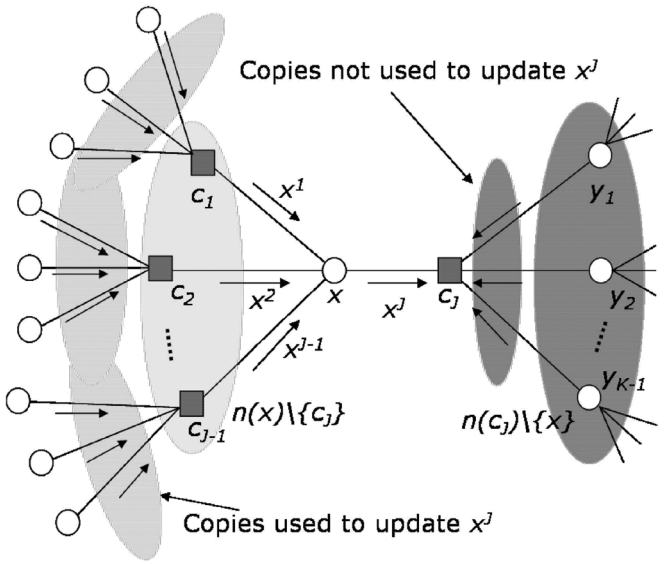


Fig. 5. Message passing in Tanner graph: updating the value of message on edge e connecting the node x and check c_J.

illustrated in Fig. 4. The outgoing message along an edge e of a variable node is updated based on the incoming messages along all edges incident on x_i excluding e (Fig. 5). The original value is used to resolve ties.

For an edge e, let the corresponding check node be c_e. The incoming message along e is the XOR of all other incoming messages along the edges incident on c_e other than e. The scheme differs from Gallager B decoder in only one aspect, that of breaking ties. In a memory system, there is no way to store the original value to resolve ties. So, the ties will be broken based on the previous value in the register (previous message along the edge).

In the TK scheme, the parity checks give an indication of whether the parity check is satisfied or not. If a bit is in error, we expect majority of the checks in which the bit is a part to be unsatisfied thereby leading to its correction. However, if we exclude the bit we are trying to estimate and calculate the parity, then the resulting value is an estimate of the bit rather than the parity. A particular copy is now updated based on $J-1$ estimates rather than $J-1$ parity values, i.e., the updated value is majority of the $J-1$ estimates and possibly makes use of previous value to resolve ties. This is another modification that we propose. The modified decoding procedure is as follows:

1) Evaluate $J-1$ estimates for each bit-copy (exclude one different parity check from the original set of checks for each bit-copy).
2) Update the value of bit copy based on majority of $J-1$ estimates (and the previous value of bit, if needed).
3) Iterate (1) and (2).

Note on the complexity of the schemes: The complexity of a system is the number of components in the system [17]. The original TK scheme requires Jn memory cells, mK J-input XOR gates, nJ $(J-1)$-input majority logic gates and nJ two-input XOR gates. The modified scheme requires Jn memory cells, mK $(K-1)$-input XOR gates and nJ $(J-1)$-input majority logic gates.

VI. Analytical Characterization of Faulty Decoders

In this section, we set up a theoretical framework for the analysis of faulty message passing decoders. Since a memory failure is said to have occurred if the bit copies in the registers do not correspond to a correctable error pattern, understanding decoding failures of message passing decoders is of prime importance. For an analytical characterization of the performance of faulty message passing decoders, it is important to understand and characterize the decoding failures of message passing decoders built from perfect gates. The main obstacle in the analytical characterization of performance of iterative decoders is the dependence of messages passed in successive iterations. This is why it is easy to analyze an ensemble of code in the limit of the codelength and girth tending to infinity [20], [21], [32]. For analyzing a particular code even for the number of iterations for which the messages are independent, a complete knowledge of the combinatorial properties of the code is essential. Even in this very restrictive case, there is no general method which can predict the performance of a code accurately as the code performance is dependent on the choice of decoding algorithm and implementation nuances such as the precision of the messages passed. However, finite length analysis of codes over the erasure channel [45] and the investigation of error floors of LDPC codes [46]–[55] have thrown some light on the decoding failures for a given code. Roughly, error floor is an abrupt change in the frame error rate (FER) performance of an iterative decoder in the high signal-to-noise ratio (SNR) region. We use this recently gained understanding to characterize the failures of faulty decoders [51].

A. Decoding Failures of Gallager B Decoder

We define a decoding failure as an event that is said to have occurred when the decoded word is different from the transmitted codeword. The concept of error events that lead to a decoding error is well-understood for the conventional bounded distance decoder and the maximum likelihood decoder. A generalization of conventional decoder errors, necessitated by the graph based decoders, is effected by defining a class of error events, namely trapping set events. The study of trapping sets is fundamental to the understanding the dynamics of iterative decoding algorithms. The dynamics of a decoder is dictated by the properties of the Tanner graph and the decoding algorithm. A common approach to improving performance of a code is

to avoid detrimental topological structures (characteristic examples of this approach include: [56]–[65]). Unfortunately, the above papers do not contribute to the clarity of the error mechanisms. Richardson's work [46] paved the way for understanding decoding failures in terms of trapping sets and since then there has been considerable work in this area. In [53], [54], and [51] we have studied trapping sets of Gallager B decoder for column-weight-three codes. Broadly speaking, a trapping set is a small weight error pattern which leads to a decoding failure. While the performance of bounded distance decoders is dependent on the minimum distance, the performance of iterative decoders heavily depends on trapping sets.

B. Decoding Failures of Faulty Gallager B Decoder

In memory systems where the restoration organ is built from perfect gates, characterizing decoding failures is analogous to the case of a BSC. The memory cells can be modeled as a BSC and the restoration organ refreshes the contents of the registers periodically. If the error pattern is a correctable one, the new contents of the registers will be a valid codeword. However, in the case of faulty restoration organs, the contents of the registers need not contain a valid codeword.

The system in this case can be visualized in the following manner. Initially J copies of a codeword are stored in the memory. The content of each memory cell is passed through a BSC. This introduces a few errors in the memory. The restoration organ then tries to correct these errors. Due to the faulty nature of the restoration organ, it corrects some errors and possibly introduces new ones. However, we hope that the contents still belong to the decoding class of the original codeword. The faulty decoder in this case can be modeled as another binary channel which introduces possibly correlated errors. Consider the following memory architecture.

1) Evaluate $J-1$ estimates for each bit-copy (exclude one different parity check from the original set of checks for each bit-copy).
2) Update the value of bit copy based on majority of $J-1$ estimates (and the previous value of bit, if needed).
3) Iterate (1) and (2).
4) Replace each estimate of a bit copy by the majority of the corresponding J copies of the bit estimates.

The analysis of decoding failures of such a scheme would be similar to analysis of the Gallager B decoder. In this paper, we just note the analogy with the analysis of Gallager B decoder but the analysis of the scheme is beyond the scope this paper. Instead, we illustrate a few numerical results in subsequent section.

C. Restoration Based on Projective Geometry Codes and One Step Majority Logic Decoding

One class of decoders which are amenable for analysis are the so called majority logic decoders [66]. Though these decoders do not fall into the category of message passing algorithms, they are iterative in nature, and can be used for restoration organs in TK scheme. Codes constructed from projective and affine planes form an important class of majority logic decodable codes whose combinatorial properties are well understood. Projective geometry and affine geometry codes [67] have gained popularity due to their elegant geometric structure and suitability for iterative decoding.

Let C_s represent the LDPC code constructed based on the family of the two-dimensional projective geometries $\mathrm{PG}(2,2^s)$, $s \geq 1$. This code is characterized by $n_v = 2^{2s}+2^s+1$ variable nodes, $n_c = 2^{2s}+2^s+1$ check nodes, minimum distance $d_{\min} = 2^s+2$ and column weight $\gamma = 2^s+1$. It is well known that such a code can correct up to $\lfloor \gamma/2 \rfloor$ errors when decoded using a one step majority logic decoder.

In one step majority logic decoding algorithm, the value of a variable is flipped if majority of the check nodes connected to it are unsatisfied. A single step majority logic decoder is a very low complexity decoder which requires very little circuitry. The bit error performance in this case is well understood. This serves as a motivation to analyze these codes and decoders for the faulty gate case. In [68], we proposed an analytical method to evaluate the performance of one-step majority logic decoders constructed from faulty gates. The decoder construction is chosen in such a way as to extend the analysis for multistep decoding and possible extension to message passing algorithms such as the Gallager B algorithm.

VII. Numerical Results

In this section, we present simulation results for the $\mathrm{PG}(2,4)$ code of length seven and dimension four. We assume that errors in memory cells occur with probability α, and errors in XOR gates and majority logic gates with probability ϵ_{XOR} and ϵ_{maj} respectively. The gate failures are of von Neumann type. The gates used in the circuit are 3-input XOR and 3-input majority logic gates. For a given memory failure probability α, we assume that the components in the correcting circuit fail with a probability 0.01α. It should be noted that these values are used for illustration purpose only and that the scheme can be evaluated for any set of values of failures.

We assume that the memory contents pass through a BSC after a time period T and then updated by the message passing decoder. The message passing can be run for any number of iterations. It is assumed that the time for update is smaller compared to T and that memory contents do not change while the update is in progress. The bit error rate (BER) and FER at any time are determined by using the original definition of memory failure by Taylor [17]. We have run simulations for various values of α and different number of iterations of the message passing decoder.

We compare bit error performance and complexity of the proposed scheme with three different schemes.

i) *Uncoded case:* In this case, the bits are stored in memory cells and no error correction is employed. This scheme uses one memory cell to store one bit and has the least complexity. The memory in this scheme can be modeled as a cascade of BSCs. The BER after p time steps in this case is given by the expression

$$\Pr(\text{error uncoded}) = \sum_{i\text{odd}} \binom{p}{i} \alpha^i (1-\alpha)^{(p-i)} = \alpha_p.$$

The above expression comes from the fact that a bit will be in error after passing through p BSCs if it has been flipped an odd number of times.

ii) *Triple redundancy (TR):* In this case, each bit is stored in three memory cells and no other coding scheme is employed. Hence, to store a single bit we need three memory cells. The bit error after p time steps in this case is given by

$$\Pr(\text{error TR}) = 3\alpha_p^2(1-\alpha_p) + \alpha_p^3 = \alpha_{\text{maj}\,p}.$$

In this case, since we take majority of the three copies, an error occurs if two or more bits get flipped.

iii) *TR with coding (TRC):* In this case each bit is stored in three memory cells in coded form, i.e., three copies of a codeword are stored. In this scheme to store k bits we need $3n$ memory cells assuming that an (n, k) code is used. The probability of error in this case is upper bounded by

$$\Pr(\text{error TRC}) \leq 1 - (1-\alpha_{\text{maj}\,p})^7 - 7(1-\alpha_{\text{maj}\,p})^6\alpha_{\text{maj}\,p}.$$

The above expression calculates the probability that the contents in the register do not belong to the decoding class of the sent codeword. Since we are using a single error correcting code, this probability is the same as making two or more errors. The expression actually calculates the FER and hence is an upper bound on the BER.

Fig. 6(a) shows the BER curves for $\alpha = 10^{-3}$ for the above schemes as well as the proposed scheme in which the message passing decoder is run for $i = 4, 6, 8$ iterations. From the plot it is clear that in the uncoded case, the bit error probability increases very quickly. The same holds true for the TR and TRC schemes. In the proposed method, the BER does not increase very rapidly with time and even after 50 time steps, it is of order 10^{-6}. It is also worth noting that running the message passing decoder for more number of iterations does not result in improved performance. However, the proposed scheme has higher complexity.

Fig. 6(b) shows the FER curves for different values of α. The message passing decoder is run for five iterations in each case. As expected the performance is heavily dependent on the value of α.

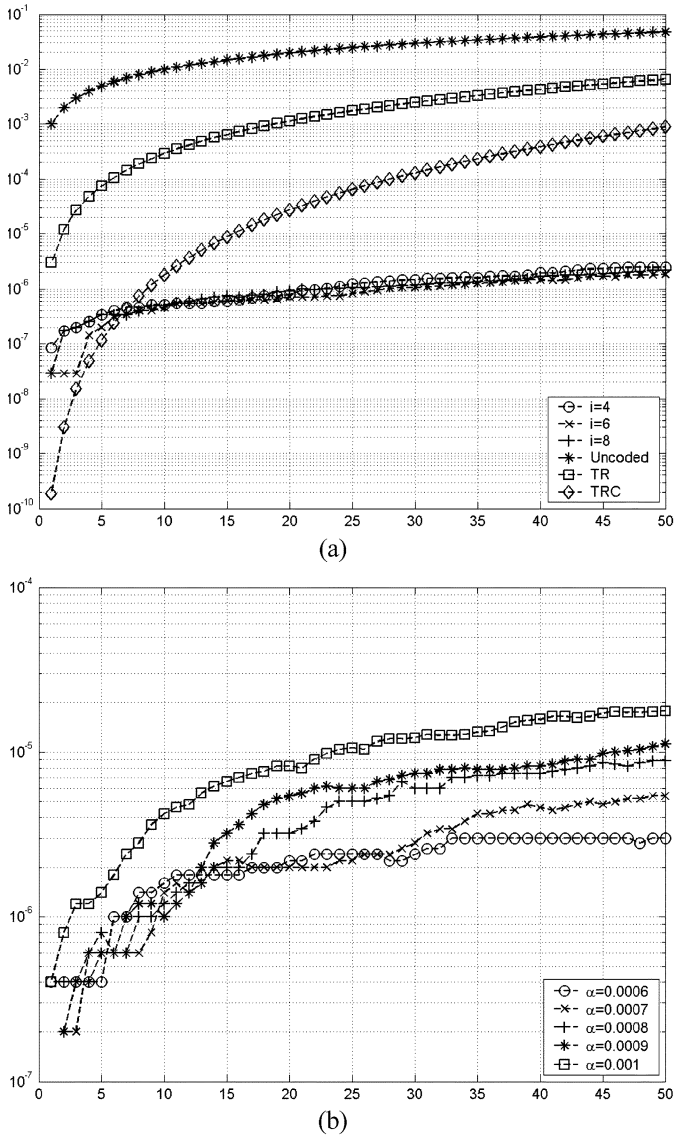


Fig. 6. Performance of faulty decoder of projective geometry code $\text{PG}(2,4)$ (a) comparison of different schemes for $\alpha = 10^{-3}$ (b) performance of proposed scheme for different memory failures.

VIII. Open Questions and Future Work

A. Other Failure Models

In addition to von Neumann type of error, this framework enables us to study other error mechanisms from the Pippenger's ϵ-admissible failure model class [9]. Dobrushin and Ortyukov [43] model of failures of connections between gates will be used in which the connections are modeled as perfect while the effect of connection failures is incorporated into the error of the gate at the end of a connection.

B. Restoration for Permanent Failures

Permanent failures are another large source of gate unreliability and are caused by variation and failures in the transistor fabrication process and transistor aging. Error correction codes have been investigated as a means for mitigating permanent failures [69], [70], but von Neumann redundant hardware schemes are more wide spread for dealing with such failures. In computing systems, the combined use of redundancy and coding has been shown to be more practically effective for permanent failures than the fault tolerance provided by either one of the two techniques independently [71], [72]. All these techniques are ad hoc and no theoretical justification for such architectures is given. The proposed architecture can be used for permanent failures as well. In addition, if the decoding algorithm detects "unusual" messages from certain memory locations, it may indicate that there is a permanent failure, and new memory locations may be allocated for such suspected permanent memory failures.

IX. Conclusion

We believe that successful demonstration of applying concepts of modern coding theory to development of novel fault-tolerant memory systems will provide a new measure of perfor-

mance in memory and communications systems. In addition to the technical merit of improved system performance it will also serve as a framework for research in these fields. New coding techniques can improve the performance of memory systems without requiring major improvements in materials and devices. This is especially important at this point in time when the physical properties of the nano devices are being pushed to their limits.

ACKNOWLEDGMENT

The authors wish to thank Dr. D. Ma and Dr. J. Wang for helpful discussions.

REFERENCES

[1] H. Goronkin and Y. Yang, "High-performance emerging solid-state memory technologies," *MRS Bulletin*, vol. 29, no. 11, pp. 805–814, Nov. 2004.

[2] W. G. Kuhr, A. R. Gallo, R. W. Manning, and C. W. Rhodine, "Molecular based on a CMOS platform," *MRS Bulletin*, vol. 29, no. 11, pp. 838–842, Nov. 2004.

[3] E. S. Hasaneen, A. Rodriguez, B. Yarlagadda, F. Jain, E. Heller, W. Huang, J. Lee, and F. Papadimitrakopoulos, "Nonvolatile quantum dot memory (NVQDM) in floating gate configuration: Device and circuit modeling," in *Proc.3rd IEEE Conf. Nanotechnology*, Aug. 12–14, 2003, vol. 2, pp. 741–744.

[4] D. D. Awschalom, M. E. Flatte, and N. Samarth, "Spintronics," *Sci. Amer.*, vol. 286, p. 66, Jun. 2002.

[5] P. Larsson and C. Svensson, "Noise in digital CMOS circuits," *IEEE J. Solid-State Circuits*, vol. 29, no. 6, pp. 655–661, Jun. 1994.

[6] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley-Interscience, 1991.

[7] N. Pippenger, "Developments in 'the synthesis of reliable organisms from unreliable gates'," in *Proc. Symp. Pure Mathematics*, 1990, pp. 311–324.

[8] J. V. Neumann, *Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components*. Princeton, NJ: Princeton University Press, 1956, pp. 43–98.

[9] N. Pippenger, "Invariance of complexity measures for networks with unreliable gates," *J. Assoc. Comput. Mach.*, vol. 36, pp. 531–539, 1989.

[10] T. M. Austin, "DIVA: A reliable substrate for deep submicron microarchitecture design," in *Proc. 32nd Annu. ACM/IEEE Int. Symp. Microarch.*, 1999, pp. 196–207.

[11] S. Buchner, M. Baze, D. Brown, D. McMorrow, and J. Melinger, "Comparison of error rates in combinatorial and sequential logic," *IEEE Trans. Nucl. Sci.*, vol. 44, no. 6, pp. 2209–2216, Dec. 1997.

[12] P. Shivakumar, M. Kistlerand, S. W. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," in *Proc. Int. Conf. Dependable Systems and Networks (DSN'02)*, 2002, pp. 389–398.

[13] M. Mezard, G. Parisi, and M. A. Virasoro, *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications*. Singapore: World Scientific, 1987, vol. 9.

[14] J. Han and P. Jonker, "A system architecture solution for unreliable nanoelectronic devices," *IEEE Trans. Nanotechnol.*, vol. 1, pp. 201–208, Dec. 2002.

[15] S. Roy and V. Beiu, "Majority multiplexing-economical redundant fault-tolerant designs for nanoarchitectures," *IEEE Trans. Nanotechnol.*, vol. 4, no. 4, pp. 441–451, Jul. 2005.

[16] K. Nikolic, A. Sadek, and M. Forshaw, "Architectures for reliable computing with unreliable nanodevices," in *Proc. IEEE NANO*, 2001, pp. 254–259.

[17] M. Taylor, "Reliable information storage in memories designed from unreliable components," *Bell Syst. Tech. J.*, vol. 47, pp. 2299–2337, 1968.

[18] M. Taylor, "Reliable computation in computing systems designed from unreliable components," *Bell Syst. Tech. J.*, vol. 47, pp. 2339–2266, Dec. 1968.

[19] A. Kuznetsov, "Information storage in a memory assembled from unreliable components," *Prob. Inf. Transmission*, vol. 9, pp. 254–264, 1973.

[20] R. G. Gallager, *Low Density Parity Check Codes*. Cambridge, MA: M.I.T. Press, 1963.

[21] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[22] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[23] J. Pearl, *Probablisitic Reasoning in Intelligent Systems*. San Francisco, CA: Kaufmann, 1988.

[24] S. L. Lauritzen and D. J. Spiegelhalter, "Local computations with probabilities on graphical structures and their application to expert systems," *J. Roy. Statist. Soc., Ser. B*, vol. 50, pp. 157–224, 1988.

[25] D. J. C. MacKay and R. M. Neal, Good error correcting codes based on very sparse matrices, Cambridge Univ., Cambridge, U.K. [Online]. Available: http://wol.ra.phy.cam.ac.uk/mackay

[26] R. J. McEliece, D. J. C. MacKay, and J. F. Cheng, "Turbo decoding as an instance of Pearl's 'belief propagation' algorithm," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 2, pp. 140–152, Feb. 1998.

[27] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannonlimit error-correcting coding and decoding: Turbo codes," in *Proc. IEEE Int. Conf. Commun.*, May 23–26, 1993, vol. 2, pp. 1064–1070.

[28] N. Wiberg, Codes and decoding on general graphs Linkoping University, Linkoping, Sweden, Apr. 1996.

[29] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. IT-27, no. 5, pp. 533–547, Sep. 1981.

[30] G. D. Forney, Jr., "On iterative decoding and the two-way algorithm," in *Proc. Int. Symp. Turbo Codes and Related Topics*, Brest, France, Sep. 1997.

[31] T. J. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[32] T. J. Richardson, M. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 638–656, Feb. 2001.

[33] M. A. Margulis, "Explicit group-theoretic constructions for combinatorial designs with applications to expanders and concentrators," *Probl. Pered. Inform.*, vol. 24, no. 1, pp. 51–60, 1988.

[34] J. Rosenthal and P. O. Vontobel, "Constructions of LDPC codes using Ramanujan graphs and ideas from Margulis," in *Proc. 38th Annu. Allerton Conf. Communication, Control, and Computing*, 2000, pp. 248–257.

[35] D. Spielman, "Highly fault-tolerant parallel computation," in *Proc. IEEE Conf. Foundations of Computer Science*, 1996, pp. 154–163.

[36] C. N. Hadjicostis and G. C. Verghese, "Coding approaches to fault tolerance in linear dynamic systems," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 210–228, Jan. 2005.

[37] H. Kimura, T. Hanyu, and M. Michitaka, "Multiple-valued logic-in-memory VLSI using MFSFETs and its applications," *J. Multiple-Valued Logic Soft Computing* vol. 9, no. 1, pp. 23–42, 2003 [Online]. Available: http://www.oldcitypublishing.com/MVLSC/MVLSC.htm

[38] D. Olson, "Multiple-valued logic circuit architecture; supplementary symmetrical logic circuit structure (SUS-LOC)," Oct. 17, 2000 [Online]. Available: http://www.uspto.gov/patft/

[39] P. Cappelletti, C. Golla, P. Olivo, and E. Zano, *Flash Memories*. Boston, MA: Kluwer, 1999.

[40] B. Eitan, R. Kazerounian, A. Roy, G. Crisenza, P. Cappelletti, and A. Modelli, "Multilevel flash cells and their trade-offs," in *Tech. Dig. Int. Electron Device Meeting*, Dec. 8–11, 1996, pp. 169–172.

[41] B. Ricco, G. Torelli, M. Lanzoni, A. Manstretta, H. E. Maes, D. Montanari, and A. Modelli, "Nonvolatile multilevel memories for digital applications," *Proc. IEEE*, vol. 86, no. 12, pp. 2399–2423, Dec. 1998.

[42] R. Bahar, J. Mundy, and J. Chen, "A probability-based design methodology for nanoscale computation," in *Proc. Int. Conf. Computer Aided Design (ICCAD'03)*, Nov. 9–13, 2003, pp. 480–486.

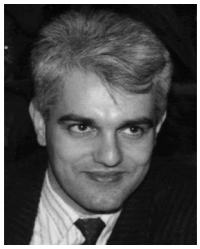
[43] R. L. Dobrushin and S. I. Ortyukov, "Lower bound for the redundancy of self-correcting arrangements of unreliable functional elements," *Probl. Inform. Transm.*, vol. 13, pp. 59–65, 1977.

[44] M. Shokrollahi, *An Introduction to Low-Density Parity-Check Codes*. New York: Springer-Verlag, 2002.

[45] C. Di, D. Proietti, T. Richardson, E. Telatar, and R. Urbanke, "Finite length analysis of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 48, pp. 1570–1579, Jun. 2002.

[46] T. J. Richardson, "Error floors of LDPC codes," in *Proc. 41st Annu. Allerton Conf. Communications, Control and Computing*, 2003, pp. 1426–1435.

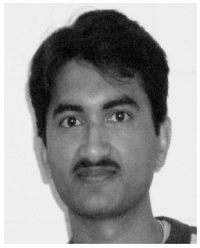
[47] M. Stepanov, V. Chernyak, M. Chertkov, and B. Vasic, "Instanton approach for codes without/with loops," in *Proc. Los Alamos Workshop Appl. Stat. Phys. Coding Theory*, Jan. 10–12, 2005.
[48] M. G. Stepanov, V. Chernyak, M. Chertkov, and B. Vasic, "Diagnosis of weaknesses in modern error correction codes: A physics approach," *Phys. Rev. Lett.*, vol. 95, p. 228701, Nov. 2005.
[49] V. Chernyak, M. Chertkov, M. Stepanov, and B. Vasic, "Local theory of BER for LDPC codes: Instantons on a tree," in *Proc. Los Alamos Workshop Appl. Stat. Phys. Coding Theory*, Jan. 10–12, 2005.
[50] V. Chernyak, M. Chertkov, M. Stepanov, and B. Vasic, "Instanton method of post-error-correction analytical evaluation," in *Proc. IEEE Inf. Theory Workshop*, Oct. 2004, pp. 220–224.
[51] S. K. Chilappagari, S. Sankaranarayanan, and B. Vasic, "Error floors of LDPC codes on the binary symmetric channel," in *Proc. Int. Conf. Communications*, Jun. 11–15, 2006, vol. 3, pp. 1089–1094.
[52] M. Chertkov, "Instanton for random avection," *Phys. Rev. E*, vol. 3, pp. 2722–2735, 1997.
[53] B. Vasic, S. K. Chilappagari, S. Sankaranarayanan, and R. Radhakrishnan, "Failures of the Gallager B decoder: Analysis and applications," in *Proc. UCSD Center for Info. Theory and Its Applications Inaugural Workshop*, Feb. 6–9, 2006 [Online]. Available: htpp//ita.5i.net/papers/160.pdf
[54] B. Vasic, S. K. Chilappagari, and S. Sankaranarayanan, "Error floor characterization of low-density parity check codes on binary symmetric channel," in *Proc. IEEE Commun. Theory Workshop*, Jun. 12–15, 2005.
[55] C. Kelley, D. Sridhara, J. Xu, and J. Rosenthal, "Pseudocodeword weights and stopping sets," in *Proc. Int. Symp. Inf. Theory*, Chicago, IL, Jun. 27 –Jul. 2, 2004, p. 150.
[56] R. Koetter and P. O. Vontobel, "Graph covers and iterative decoding of finite-length codes," in *Proc. 3rd Int. Conf. Turbo Codes and Related Topics*, Sep. 1–5, 2003, pp. 75–82.
[57] D. J. C. MacKay, Encyclopedia of sparse graph codes, Cambridge Univ., Cambridge, U.K. [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html
[58] D. J. C. MacKay and M. C. Davey, Evaluation of Gallager codes for short block length and high rate applications, Univ. Toronto, Toronto, ON, Canada [Online]. Available: http://www.cs.toronto.edu/mackay/CodesRegular.html
[59] N. Varnica and A. Kavcic, "Optimized LDPC codes for partial response channels," in *Proc. IEEE Int. Symp. Inf. Theory*, 2002, p. 197.
[60] H. Tang, J. Xu, S. Lin, and K. Abdel-Ghaffar, "Codes on finite geometries," *IEEE Trans. Inf. Theory*, vol. 51, no. 2, pp. 572–596, Feb. 2005.
[61] R. M. Tanner, D. Sridhara, and T. Fuja, "A class of group-structured LDPC codes," in *Proc. ISCTA*, Ambleside, U.K., 2001 [Online]. Available: www.cse.ucsc.edu/tanner/isctaGrpStrLDPC.pdf
[62] J. Xu and S. Lin, "A combinatoric superposition method for constructing low-density parity check codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun./Jul. 2003, p. 30.
[63] B. Vasic, E. Kurtas, and A. Kuznetsov, "LDPC codes based on mutually orthogonal latin rectangles and their application in perpendicular magnetic recording," *IEEE Trans. Magn.*, vol. 38, pp. 2346–2348, Sep. 2002.
[64] B. Vasic and K. Pedagani, "Run-length-limited low-density parity-check codes based on deliberate error insertion," *IEEE Trans. Magn.*, vol. 40, pp. 1738–1743, May 2004.
[65] B. Vasic, E. Kurtas, and A. Kuznetsov, "Lattice low-density parity-check codes and their application in partial response channels," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun./Jul. 2002, p. 453.
[66] S. Lin and J. C. D. J. *, Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1983.
[67] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: A rediscovery and new results," *IEEE Trans. Inf. Theory*, vol. 47, no. 7, pp. 2711–2736, Nov. 2001.
[68] S. K. Chilappagari, M. Ivkovic, and B. Vasic, "Analysis of one step majority logic decoders constructed from faulty gates," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 9–14, 2006, pp. 469–473.
[69] C. H. Stapper and H. S. Lee, "Synergistic fault tolerance for memory chips," *IEEE Trans. Comput.*, vol. 41, no. 9, pp. 1078–1087, Sep. 1992.
[70] G. C. Zhu, W. Yu, and F. R. Kschischang, "Finite-geometry low-density parity-check codes for channels with stuck-at defects," in *Proc. Canad. Workshop Inf. Theory*, May 2005 [Online]. Available: http://www.comm.toronto.edu/weiyu/publications.html
[71] F. I. Osman, "Error-correction technique for random-access memories," *IEEE J. Solid-State Circuits*, vol. SC-17, no. 5, pp. 877–882, Oct. 1982.
[72] T. P. Haraszti*, CMOS Memory Circuits*. Boston, MA: Kluwer, 2000.



Bane Vasic (S'92–M'93–SM'02) received the B.Sc., M. Sc., and Ph.D. degrees in electrical engineering from the University of Nis, Nis, Yugoslavia (now Serbia), in 1989, 1991, and 1994, respectively.

From 1996 to 1997, he worked as a Visiting Scientist at the Rochester Institute of Technology and Kodak Research, Rochester, NY, where he was involved in research in opitcal storage channels. From 1998 to 2000, he was with Lucent Technologies, Bell Laboratories (Bell-Labs). He was involved in research in iterative decoding and low-density parity check codes, as well as development of codes and detectors implemented in Bell-Labs chips. Presently, he is a Professor in the Electrical and Computer Engineering Department, University of Arizona, Tucson. His research interests include coding theory, information theory, communication theory, and digital communications and recording.

Dr. Vasic is a Member of the Editorial Board for the IEEE TRANSACTIONS ON MAGNETICS. He served as Technical Program Chair of the IEEE Communication Theory Workshop in 2003 and as Co-organizer of the Center for Disccrete Mathematics and Theoretical Computer Science (DIMACS) Workshops on Optical/Magnetic Recording and Optical Transmission, and Theoretical Advances in Information Recording in 2004. He was Co-organizer of the Los Alamos Workshop on Applications of Statistical Physics or Coding Theory in 2004, the Communication Theory Symposium within the IEEE International Conference on Communications (ICC 2006), and IEEE Communication Workshop (CTW 2007).



Shashi Kiran Chilappagari (S'05) received the B.Tech. and M.Tech. degrees in electrical engineering from the Indian Institute of Technology—Madras, Chennai, India, in 2004. He is currently working toward the Ph.D. degree in electrical engineering at the University of Arizona, Tucson.

His research interests include error control coding, data compression and information theory.