Partial-MDS Codes and Their Application to RAID Type of Architectures

Mario Blaum, *Fellow, IEEE*, James Lee Hafner, and Steven Hetzler, *Senior Member, IEEE*

***Abstract*—A family of codes with a natural 2-D structure is presented, inspired by an application of redundant arrays of independent disks (RAID) type of architectures whose units are solid-state drives (SSDs). Arrays of SSDs behave differently from arrays of hard disk drives, since hard errors in sectors are common and traditional RAID approaches (like RAID 5 or RAID 6) may be either insufficient or excessive. An efficient solution to this problem is given by the new codes presented, called partial maximum distance separable (PMDS) codes.**

***Index Terms*—Array codes, error-correcting codes (ECCs), flash storage devices, hard errors, maximum distance separable (MDS) codes, RAID architectures, Reed–Solomon codes, solid-state drives (SSDs).**

I. Introduction

CONSIDER an array of, say, n storage devices. Each storage device contains a (large) number of sectors, each sector protected by an error-correcting code (ECC) dealing with the most common errors in the media. However, it may occur that one or more of the storage devices experiences a catastrophic failure, where by catastrophic we mean that all the information in the device is lost. In that case, data loss will occur if no further protection is implemented. For that reason, the architecture known as redundant arrays of independent disks (RAID) was proposed [11].

The way RAID architectures work is by assigning one or more devices to parity. For instance, take a row of n sectors, we call this set of n sectors, a "stripe": $n-1$ sectors carry data, while the nth is the XOR of the $n-1$ data sectors. We repeat this for each stripe of sectors in the array. Such an architecture is called a RAID 4 or a RAID 5 type of architecture. In what follows, we will call it RAID 5. The difference between RAID 5 and RAID 4 is in the distribution of the parity sectors, but we do not address this issue here. A RAID 5 architecture provides protection against a catastrophic device failure. A RAID 6 architecture gives protection against two catastrophic failures. We will use interchangeably the terms row and stripe.

Manuscript received May 01, 2012; revised November 15, 2012; accepted February 27, 2013. Date of publication June 12, 2013; date of current version June 12, 2013.

The authors are with the IBM Almaden Research Center, San Jose, CA 95120 USA (e-mail: mblaum@us.ibm.com; hafner@us.ibm.com; hetzler@us.ibm.com).

Communicated by N. Kashyap, Associate Editor for Coding Theory.

Digital Object Identifier 10.1109/TIT.2013.2252395

From a coding point of view, the model of failures corresponds to erasures, i.e., errors whose location is known [22]. It is preferable to use maximum distance separable (MDS) codes for RAID 6 types of architectures: in order to correct two erasures, exactly two parities are needed. There are many choices for MDS codes correcting two or more erasures: we can use Reed–Solomon (RS) codes [23], [26], or array codes, like EVENODD [2], RDP [8], X-codes [34], B-codes [33], C-codes [21], Liberation codes [27], and others [3], [31].

Architectures like RAID 5 and RAID 6 are efficient when the storage devices are hard disk drives (HDDs). However, when using solid-state drives (SSDs) like flash, these types of architectures, by themselves, either are not efficient or they are wasteful. Arrays of SSDs pose new challenges for code design, so we will spend the rest of this section addressing some of them. Different ways to adapt RAID architectures to SSDs are being considered in recent literature. For instance, ways to enhance the performance of RAID 5 are described in [1], [19], and [25]. See also [13] and [18], where the internal ECC and an RAID type of architecture interact. In particular, Greenan *et al*. [13] use an adaptive method to increase the redundancy when the bit-error rate increases.

Contrary to HDDs, SSDs degrade significantly in time and as a function of the number of writes [24]. As time goes by and the number of writes increases, the likelihood of a hard error in a sector also increases. A hard error occurs mainly when the correction capability of the internal ECC of a sector is exceeded. In general, BCH codes [22] are used for the internal ECC of a sector, although many other codes (including LDPC codes with soft decoding) are possible. However, we do not address the internal ECC problem in this paper. The point is a hard error corresponds to an uncorrectable error in a sector. Normally, the ECC is coupled with a cyclic redundancy code (CRC), which detects the situation when the ECC miscorrects (the ECC has an inherent detection capability that may not be sufficient; hence, it often needs to be reinforced by the CRC). We will assume that a hard error means that the information in a sector is lost (an erased sector) and that we can detect this situation.

From the aforementioned discussion, we see that, contrary to arrays of HDDs, arrays of SSDs present a mixed failure mode: on one hand, we have catastrophic SSDs failures, as in the case of HDDs. On the other hand, we also have hard errors, which in general are silent: their existence is unknown until the sectors are accessed. This situation complicates the task of an RAID type of architecture. In effect, assume that a catastrophic SSD failure occurs in a RAID 5 architecture. Each sector of the failed device is reconstructed by XORing the corresponding sectors in each stripe of the surviving devices. However, if there is a stripe

that in addition has suffered a hard error, such a stripe has two sectors that have failed. Since we are using RAID 5, we cannot recover from such an event and data loss will occur.

A possible solution to the aforementioned situation is using a RAID 6 type of architecture, in which two SSDs are used for parity. Certainly, this architecture allows for the recovery of two erased sectors in the same stripe. However, such a solution is expensive, since it requires an additional whole device to protect against hard errors. Moreover, two hard errors in a stripe, in addition to the catastrophic device failure, would still cause data loss, and such a scenario may not be unlikely, depending on the statistics of errors. We would like a solution to this problem without dedicating a whole second SSD to parity.

In order to handle this mixed environment of hard errors with catastrophic failures, we need to take into account the way information is written in SSDs, which is quite different to the way it is done in HDDs. In an SSD, a new write consists of erasing first a number of consecutive sectors and then rewriting all of them. Therefore, the short write operation in arrays of SSDs (like one sector at a time) does not occur. In effect, we assume that the array consists of $m \times n$ blocks (i.e., each block consists of m stripes), repeated one after the other. Each $m \times n$ block is an independent unit, and we will show how to compute the parity for each block. Each new write consists of writing a number of $m \times n$ blocks (this number may be one, depending on the application, the particular SSD used, and other factors). Our goal is to present a family of codes, that we call partial-MDS (PMDS) codes, allowing for the simultaneous correction of catastrophic failures and hard errors. We will justify the name PMDS in the next section.

The paper is organized as follows: in Section II, we present the theoretical framework as well as the basic definitions. In Section III, we present our main construction together with an alternative construction. In Section IV, we study the special case in which the general construction of Section III extends RAID 5, and we find the general conditions for such codes to be PMDS. In Section V, we study specific cases with parameters relevant to applications, each case analyzed in a separate section. In Section VI, we present a third construction for cases extending RAID 5. This third construction is not as powerful as the previous ones (it cannot handle three erasures in the same stripe) but uses finite fields of smaller size, simplifying the implementation. For computing the probability of data loss when a catastrophic device failure has occurred under different scenarios, we refer the reader to [4] (for reasons of space, we omit this discussion here).

Although the results can be extended to finite fields of arbitrary characteristic, for simplicity, we consider only fields of characteristic 2.

II. Partial-MDS Codes

Consider an $m \times n$ array, each entry of the array consisting of b symbols (we assume that each of the b symbols is a bit for the sake of the description, but in practice it may be a much larger symbol). Each row in the array represents a stripe and this stripe is protected by r parity entries in such a way that any r erasures in the stripe will be recovered. In other words, each stripe of the array constitutes a codeword of an $[n, n-r, r+1]$ MDS code. In addition, we will add s extra "global" parities. Those s extra parities may be placed in different ways in the array, but in order to simplify the description we will place them in the last stripe. Being global means that these parities affect all mn entries in the array. For instance, Fig. 1 shows a 4×5 array with $r = 1$ and $s = 2$ such that the two extra global parities are placed in the last stripe.

				P
				P
				P
		P	P	P

Fig. 1. 4×5 array with $r = 1$ and $s = 2$.

	H		F	
			F	
			F	H
			F	

			F	
			F	
H			F	H
			F	

Fig. 2. 4×5 arrays with a catastrophic failure and two hard errors.

The idea of a partial-MDS code (to be defined formally) is the following: looking at Fig. 1, assume that a catastrophic failure occurs (that is, a whole column in the array has failed), and in addition, we have up to two hard errors anywhere in the array. Then, we want the code to correct these failures (erasures in coding parlance). The situation is illustrated in Fig. 2, where the hard errors are indicated with the letter "H": the two hard errors may occur either in different stripes or in the same stripe.

A natural way of solving this problem is by using an MDS code. In our 4×5 array example, we have a total of six parity sectors. So, it is feasible to implement an MDS code on 20 symbols with six parity symbols, in other words a $[20, 14, 7]$ MDS code (like an RS code). The problem with this approach is its complexity. The case of a 4×5 array is given for the purpose of illustration, but more typical values of m in applications are $m = 16$ and even $m = 32$. That would give 18 or 34 parity sectors. Implementing such a code, although feasible, is complex. We want the code, in normal operation, to utilize its underlying RAID structure based on stripes, like single parity in the case of RAID 5. The extra parities are invoked in rare occasions. So, given this constraint of a horizontal code, we want to establish an optimality criterion for codes, which we will call PMDS codes. In the case of the example of RAID 5 plus two global parities, we want the code to correct up to one erasure per stripe, and in addition, two extra erasures anywhere. For example, the code of Fig. 1 is PMDS if it can correct any of the situations depicted in Fig. 2. We call the codes PMDS because they are MDS on rows, and in addition, they require global parities satisfying an optimality constraint. Formally, we have the following.

Definition 2.1: Let $\mathcal{C}$ be a linear $[mn, m(n-r)-s]$ code over a field such that when codewords are taken row-wise as $m \times n$ arrays, each row belongs in an $[n, n-r, r+1]$ MDS code. Given $t \geq 1$ and $(s_1, s_2, \ldots, s_t)$ such that each $s_j \geq 1$ and $\sum_{j=1}^{t} s_j = s$, we say that $\mathcal{C}$ is $(r; s_1, s_2, \ldots, s_t)$-erasure

correcting if, for any $0 \leq i_1 < i_2 < \cdots < i_t \leq m-1, \mathcal{C}$ can correct up to $s_j + r$ erasures in each row i_j of an array in $\mathcal{C}$. We say that $\mathcal{C}$ is an $(r; s)$ partial-MDS (PMDS) code if, for every $t \geq 1$ and for every $(s_1, s_2, \ldots, s_t)$ such that each $s_j \geq 1$ and $\sum_{j=1}^{t} s_j = s, \mathcal{C}$ is an $(r; s_1, s_2, \ldots, s_t)$-erasure correcting code.

In the next section, we give a general construction of codes by providing their $(mr + s) \times mn$ parity-check matrices. Some of these codes are going to be PMDS. In particular, we will analyze the case $r = 1$ in Section IV due to its important practical value, since it extends RAID 5.

III. Code Construction

As stated in Section II, our entries consist of b bits. We will assume that each entry is in the field $GF(2^b)$[23].

Given an element $\alpha \in GF(2^b)$, we say that the (multiplicative) order of α, denoted $\mathcal{O}(\alpha)$, is the minimum $\ell, 0 < \ell$, such that $\alpha^\ell = 1$. If α is primitive [23], then $\mathcal{O}(\alpha) = 2^b - 1$. Associated with α there is a minimal (irreducible) polynomial that we denote $f_\alpha(x)$[23].

We present next a general construction, and then we illustrate it with some examples.

Construction 3.1: Let $\alpha \in GF(2^b)$ and let $mn \leq \mathcal{O}(\alpha)$. Let $\mathcal{C}(m, n, r, s; f_\alpha(x))$ be the code whose $(mr + s) \times mn$ parity-check matrix is

$$\mathcal{H}(m,n,r,s) = \left(\begin{array}{c|c|c|c} H(n,r,0,0) & \underline{0}(n,r) & \cdots & \underline{0}(n,r) \\ \underline{0}(n,r) & H(n,r,0,r) & \cdots & \underline{0}(n,r) \\ \vdots & \vdots & \ddots & \vdots \\ \underline{0}(n,r) & \underline{0}(n,r) & \cdots & H(n,r,0,(m-1)r) \\ \hline \multicolumn{4}{c}{H(mn,s,r,0)} \end{array}\right) \tag{1}$$

where $\underline{0}(n, r)$ is the $r \times n$ matrix consisting of zeros and $H(n, r, i, j)$ is the $r \times n$ matrix

$$H(n,r,i,j) = \begin{pmatrix} \beta^j & \beta^{j+1} & \beta^{j+2} & \cdots & \beta^{j+n-1} \\ \alpha^{j2^i} & \alpha^{(j+1)2^i} & \alpha^{(j+2)2^i} & \cdots & \alpha^{(j+n-1)2^i} \\ \alpha^{j2^{i+1}} & \alpha^{(j+1)2^{i+1}} & \alpha^{(j+2)2^{i+1}} & \cdots & \alpha^{(j+n-1)2^{i+1}} \\ \alpha^{j2^{i+2}} & \alpha^{(j+1)2^{i+2}} & \alpha^{(j+2)2^{i+2}} & \cdots & \alpha^{(j+n-1)2^{i+2}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha^{j2^{i+r-2}} & \alpha^{(j+1)2^{i+r-2}} & \alpha^{(j+2)2^{i+r-2}} & \cdots & \alpha^{(j+n-1)2^{i+r-2}} \end{pmatrix} \tag{2}$$

where $\beta = 1$ if $i = 0$ and $\beta = \alpha^{2^{i-1}}$ if $i > 0$.

Let us point out that matrices $H(n, r, i, j)$ as given by (2), in which each row is the square of the previous one, were used in [9], [10], and [29] for constructing codes for which the metric is given by the rank, in [5] for constructing codes that can be encoded on columns and decoded on rows, and in [20] for constructing the so-called differential MDS codes.

The idea of Construction 3.1 is providing PMDS codes with simple encoding and decoding algorithms, and that the field $GF(2^b)$ used is as small as possible. Also, we want a relatively fast algorithm to check whether a code is PMDS. The problem is difficult in general, but for special cases of importance in applications, like $r = 1$, we will show the existence of PMDS codes using Construction 3.1.

Let us illustrate Construction 3.1 in the next example.

Example 3.1: Consider an element α in $GF(2^b)$ such that $\mathcal{O}(\alpha) \geq 15$. Let $m = 3$ and $n = 5$; then, the parity-check matrix of $\mathcal{C}(3, 5, 1, 3\,;\, f_\alpha(x))$ is given by

$$\mathcal{H}(3,5,1,3) = \left(\begin{array}{ccccc|ccccc|ccccc} 1&1&1&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&1&1&1&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1 \\ \hline 1&\alpha&\alpha^2&\alpha^3&\alpha^4&\alpha^5&\alpha^6&\alpha^7&\alpha^8&\alpha^9&\alpha^{10}&\alpha^{11}&\alpha^{12}&\alpha^{13}&\alpha^{14} \\ 1&\alpha^2&\alpha^4&\alpha^6&\alpha^8&\alpha^{10}&\alpha^{12}&\alpha^{14}&\alpha^{16}&\alpha^{18}&\alpha^{20}&\alpha^{22}&\alpha^{24}&\alpha^{26}&\alpha^{28} \\ 1&\alpha^4&\alpha^8&\alpha^{12}&\alpha^{16}&\alpha^{20}&\alpha^{24}&\alpha^{28}&\alpha^{32}&\alpha^{36}&\alpha^{40}&\alpha^{44}&\alpha^{48}&\alpha^{52}&\alpha^{56} \end{array}\right)$$

while the parity-check matrix of $\mathcal{C}(3, 5, 2, 2\,;\, f_\alpha(x))$ is

$$\mathcal{H}(3,5,2,2) = \left(\begin{array}{ccccc|ccccc|ccccc} 1&1&1&1&1&0&0&0&0&0&0&0&0&0&0 \\ 1&\alpha&\alpha^2&\alpha^3&\alpha^4&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&1&1&1&1&0&0&0&0&0 \\ 0&0&0&0&0&\alpha^5&\alpha^6&\alpha^7&\alpha^8&\alpha^9&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1 \\ 0&0&0&0&0&0&0&0&0&0&\alpha^{10}&\alpha^{11}&\alpha^{12}&\alpha^{13}&\alpha^{14} \\ \hline 1&\alpha^2&\alpha^4&\alpha^6&\alpha^8&\alpha^{10}&\alpha^{12}&\alpha^{14}&\alpha^{16}&\alpha^{18}&\alpha^{20}&\alpha^{22}&\alpha^{24}&\alpha^{26}&\alpha^{28} \\ 1&\alpha^4&\alpha^8&\alpha^{12}&\alpha^{16}&\alpha^{20}&\alpha^{24}&\alpha^{28}&\alpha^{32}&\alpha^{36}&\alpha^{40}&\alpha^{44}&\alpha^{48}&\alpha^{52}&\alpha^{56} \end{array}\right).$$

So far, we have not proved that Construction 3.1 provides PMDS codes. Actually, this is not true in general. The answer depends on the particular parameters m and n and on the element α in the field. We can also try random constructions and check whether they provide PMDS codes. Such an approach was tried in [28] for a family of codes called sector-disk (SD) codes, of which PMDS codes are a subset. SD codes handle the situation in which whole columns have been erased (representing catastrophic device failures) and in addition s random erasures have occurred; thus, SD codes require less stringent conditions than PMDS codes (but probably sufficient for most applications). It was found in [28] that random codes are unlikely to give SD (and hence PMDS) codes in general. However, after extensive searches, in some cases, random codes gave SD codes with better parameters than codes obtained with Construction 3.1. This raises the hope of finding theoretically families of PMDS and SD codes, at least for some relevant parameters.

We denote by $(a_{i,j})_{\substack{0 \leq i \leq m-1 \\ 0 \leq j \leq n-1}}$ the received entries from a stored array in $\mathcal{C}(m, n, r, s; f_\alpha(x))$, assuming that the erased ones are equal to 0. The first step to retrieve the erased entries consists of computing the $rm + s$ syndromes. Using the

parity-check matrix $\mathcal{H}(m,n,r,s)$ given by (1), the syndromes are

$$S_{ir} = \bigoplus_{j=0}^{n-1} a_{i,j} \quad \text{for} \quad 0 \le i \le m-1 \tag{3}$$

$$S_{ir+l+1} = \bigoplus_{j=0}^{n-1} \alpha^{(ni+j)2^l} a_{i,j} \text{ for } 0 \le i \le m-1, 0 \le l \le r-2 \tag{4}$$

$$S_{mr+u} = \bigoplus_{i=0}^{m-1} \bigoplus_{j=0}^{n-1} \alpha^{(ni+j)(2^{r+u-1})} a_{i,j} \quad \text{for} \quad 0 \le u \le s-1. \tag{5}$$

After computing the syndromes, the erasures are recovered by solving a linear system based on the parity-check matrix, provided that such a solution exists.

Before leaving this section, we present an alternative to Construction 3.1.

Construction 3.2: Let $\alpha \in GF(2^b)$ and let $mn \le \mathcal{O}(\alpha)$. Let $\mathcal{C}^{(1)}(m,n,r,s;f_\alpha(x))$ be the code whose $(mr+s)\times mn$ parity-check matrix is

$$\mathcal{H}^{(1)}(m,n,r,s) = \left(\begin{array}{c|c|c|c} H^{(1)}(n,r,0,0) & \underline{0}(n,r) & \dots & \underline{0}(n,r) \\ \underline{0}(n,r) & H^{(1)}(n,r,0,r) & \dots & \underline{0}(n,r) \\ \vdots & \vdots & \ddots & \vdots \\ \underline{0}(n,r) & \underline{0}(n,r) & \dots & H^{(1)}(n,r,0,(m-1)r) \\ \hline \multicolumn{4}{c}{H^{(1)}(mn,s,r,0)} \end{array}\right) \tag{6}$$

where $H^{(1)}(n,r,i,j)$ is the $r\times n$ matrix

$$H^{(1)}(n,r,i,j) = \left(\begin{array}{c|c|c|c} \alpha^{ij} & \alpha^{i(j+1)} & \alpha^{i(j+2)} & \alpha^{i(j+n-1)} \\ \alpha^{(i+1)j} & \alpha^{(i+1)(j+1)} & \alpha^{(i+1)(j+2)} & \alpha^{(i+1)(j+n-1)} \\ \alpha^{(i+2)j} & \alpha^{(i+2)(j+1)} & \alpha^{(i+2)(j+2)} & \alpha^{(i+2)(j+n-1)} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha^{(i+r-1)j} & \alpha^{(i+r-1)(j+1)} & \alpha^{(i+r-1)(j+2)} & \alpha^{(i+r-1)(j+n-1)} \end{array}\right) \tag{7}$$

and $\underline{0}(n,r)$ is an $r\times n$ zero matrix.

Next, we illustrate Construction 3.2 with some examples.

Example 3.2: As in Example 3.1, consider an element α in $GF(2^b)$ such that $\mathcal{O}(\alpha) \ge 15$. Let $m=3$ and $n=5$; then, the parity-check matrix of $\mathcal{C}(3,5,1,3;f_\alpha(x))$ is given by

$$\mathcal{H}^{(1)}(3,5,1,3) = \left(\begin{array}{ccccc|ccccc|ccccc} 1&1&1&1&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&1&1&1&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1\\ \hline 1&\alpha&\alpha^2&\alpha^3&\alpha^4&\alpha^5&\alpha^6&\alpha^7&\alpha^8&\alpha^9&\alpha^{10}&\alpha^{11}&\alpha^{12}&\alpha^{13}&\alpha^{14}\\ 1&\alpha^2&\alpha^4&\alpha^6&\alpha^8&\alpha^{10}&\alpha^{12}&\alpha^{14}&\alpha^{16}&\alpha^{18}&\alpha^{20}&\alpha^{22}&\alpha^{24}&\alpha^{26}&\alpha^{28}\\ 1&\alpha^3&\alpha^6&\alpha^9&\alpha^{12}&\alpha^{15}&\alpha^{18}&\alpha^{21}&\alpha^{24}&\alpha^{27}&\alpha^{30}&\alpha^{33}&\alpha^{36}&\alpha^{39}&\alpha^{42} \end{array}\right)$$

$$\mathcal{H}^{(1)}(3,5,3,1) = \left(\begin{array}{ccccc|ccccc|ccccc} 1&1&1&1&1&0&0&0&0&0&0&0&0&0&0\\ 1&\alpha&\alpha^2&\alpha^3&\alpha^4&0&0&0&0&0&0&0&0&0&0\\ 1&\alpha^2&\alpha^4&\alpha^6&\alpha^8&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&1&1&1&1&0&0&0&0&0\\ 0&0&0&0&0&\alpha^5&\alpha^6&\alpha^7&\alpha^8&\alpha^9&0&0&0&0&0\\ 0&0&0&0&0&\alpha^{10}&\alpha^{12}&\alpha^{14}&\alpha^{16}&\alpha^{18}&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1\\ 0&0&0&0&0&0&0&0&0&0&\alpha^{10}&\alpha^{11}&\alpha^{12}&\alpha^{13}&\alpha^{14}\\ 0&0&0&0&0&0&0&0&0&0&\alpha^{20}&\alpha^{22}&\alpha^{24}&\alpha^{26}&\alpha^{28}\\ \hline 1&\alpha^3&\alpha^6&\alpha^9&\alpha^{12}&\alpha^{15}&\alpha^{18}&\alpha^{21}&\alpha^{24}&\alpha^{27}&\alpha^{30}&\alpha^{33}&\alpha^{36}&\alpha^{39}&\alpha^{42} \end{array}\right)$$

and

$$\mathcal{H}^{(1)}(3,5,2,2) = \left(\begin{array}{ccccc|ccccc|ccccc} 1&1&1&1&1&0&0&0&0&0&0&0&0&0&0\\ 1&\alpha&\alpha^2&\alpha^3&\alpha^4&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&1&1&1&1&0&0&0&0&0\\ 0&0&0&0&0&\alpha^5&\alpha^6&\alpha^7&\alpha^8&\alpha^9&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1\\ 0&0&0&0&0&0&0&0&0&0&\alpha^{10}&\alpha^{11}&\alpha^{12}&\alpha^{13}&\alpha^{14}\\ \hline 1&\alpha^2&\alpha^4&\alpha^6&\alpha^8&\alpha^{10}&\alpha^{12}&\alpha^{14}&\alpha^{16}&\alpha^{18}&\alpha^{20}&\alpha^{22}&\alpha^{24}&\alpha^{26}&\alpha^{28}\\ 1&\alpha^3&\alpha^6&\alpha^9&\alpha^{12}&\alpha^{15}&\alpha^{18}&\alpha^{21}&\alpha^{24}&\alpha^{27}&\alpha^{30}&\alpha^{33}&\alpha^{36}&\alpha^{39}&\alpha^{42} \end{array}\right).$$

$\mathcal{C}(m,n,1,2;f_\alpha(x))$ and $\mathcal{C}^{(1)}(m,n,1,2;f_\alpha(x))$ coincide, as we can see. The results in [4] show that codes $\mathcal{C}(m,n,1,3;f_\alpha(x))$ are PMDS more often than codes $\mathcal{C}^{(1)}(m,n,1,3;f_\alpha(x))$. Codes $\mathcal{C}^{(1)}(m,n,r,s;f_\alpha(x))$ were used in [28] to produce SD codes.

In the next section, we study the case $r=1$ and give necessary and sufficient conditions that determine whether a $\mathcal{C}(m,n,1,s;f_\alpha(x))$ code is PMDS.

IV. Case $r = 1$

In this section, we assume that $r=1$; thus, the parity-check matrix $\mathcal{H}(m,n,1,s)$ given by (1) can be written as

$$\mathcal{H}(m,n,1,s) = (\mathcal{H}_0(m,n,1,s), \mathcal{H}_1(m,n,1,s), \dots, \mathcal{H}_{m-1}(m,n,1,s))$$

where for $0 \le j \le m-1$

$$\mathcal{H}_j(m,n,1,s) = \begin{array}{c} \\ j \rightarrow \end{array} \left(\begin{array}{cccc} 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \\ 1 & 1 & \dots & 1 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \\ \alpha^{jn} & \alpha^{jn+1} & \dots & \alpha^{(j+1)n-1} \\ \alpha^{2jn} & \alpha^{2(jn+1)} & \dots & \alpha^{2((j+1)n-1)} \\ \alpha^{4jn} & \alpha^{4(jn+1)} & \dots & \alpha^{4((j+1)n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{2^{s-1}jn} & \alpha^{2^{s-1}(jn+1)} & \dots & \alpha^{2^{s-1}((j+1)n-1)} \end{array}\right) \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \left.\begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array}\right\} m .$$

Assume that $\sum_{j=1}^{t} s_j = s$ for integers $s_j \geq 1$. According to Definition 2.1, we will characterize when $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is $(r;s_1,s_2,\ldots,s_t)$-erasure correcting. We need a series of lemmas first.

Lemma 4.1: Let $\underline{s} = (s_1,s_2,\ldots,s_t)$ be an all-positive integer t-tuple whose entries sum to s. Then, $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is $(1;\underline{s})$-erasure correcting if and only if

1) code $\mathcal{C}(m,n,1,s-1;f_\alpha(x))$ is $(1;\underline{s}')$-erasure correcting for every $\underline{s}'$ obtained by subtracting 1 from an entry of $\underline{s}$; and
2) for any $0 = i_1 < i_2 < i_3 < \cdots < i_t \leq m-1$, and for any $1 \leq j \leq t$ and $0 \leq l_{j,0} < l_{j,1} < \cdots < l_{j,s_j} \leq n-1$

$$\bigoplus_{j=1}^{t} \alpha^{i_j n + l_{j,0} - l_{1,0}} \bigoplus_{u=1}^{s_j} \left(1 \oplus \alpha^{l_{j,u}-l_{j,0}}\right) \neq 0. \tag{8}$$

Proof: Take rows $i_1,i_2,\ldots,i_t$, where $0 \leq i_1 < i_2 < \cdots < i_t \leq m-1$, such that row i_j has exactly $s_j + r = s_j + 1$ erasures in locations $(i_j,l_{j,0}),(i_j,l_{j,1}),\ldots,(i_j,l_{j,s_j})$, for $0 \leq l_{j,0} < l_{j,1} < \cdots < l_{j,s_j} \leq n-1$ and $\sum_{j=1}^{t} s_j = s$. Assuming the erased entries to be equal to zero and computing the syndromes according to (3) and (5), we obtain

$$\bigoplus_{v=0}^{s_j} a_{i_j,l_{j,v}} = S_{i_j} \quad \text{for} \quad 1 \leq j \leq t$$

$$\bigoplus_{j=1}^{t} \bigoplus_{v=0}^{s_j} \alpha^{2^u (i_j n + l_{j,v})} a_{i_j,l_{j,v}} = S_{m+u} \quad \text{for} \quad 0 \leq u \leq s-1.$$

This system has a unique solution if and only if, by Cramer's rule, the matrix corresponding to the coefficients of the variables is invertible. By some rather tedious but straightforward row operations on this matrix (see [4] for details), it is invertible if and only if the $s \times s$ matrix $\hat{\underline{c}}$ is invertible, where $\hat{\underline{c}}$ consists of a first row $\underline{w}_0$ followed by successive rows $\underline{w}_u$ such that each component in a row is the square of the corresponding component in the previous row. The first row $\underline{w}_0$ is given by the vector of length s

$$\underline{w}_0 = \left(\underline{w}_0^{(1)}, \underline{w}_0^{(2)}, \ldots, \underline{w}_0^{(t)}\right)$$

where for $1 \leq j \leq t$, making the change of variable $i_j \leftarrow i_j - i_1$

$$\underline{w}_0^{(j)} = \left(x_1^{(j)}, x_2^{(j)}, \ldots, x_{s_j}^{(j)}\right)$$

and for $1 \leq v \leq s_j$

$$x_v^{(j)} = \alpha^{i_j n + l_{j,0} - l_{1,0}} \left(1 \oplus \alpha^{l_{j,v}-l_{j,0}}\right).$$

Matrix $\hat{\underline{c}}$ is invertible if and only if its determinant is invertible. The determinant of a matrix of this type is known [5] for fields: it is the product of the XOR of all possible subsets of elements of the first row. For example, if we have a matrix

$$\begin{pmatrix} \gamma_1 & \gamma_2 & \gamma_3 \\ \gamma_1^2 & \gamma_2^2 & \gamma_3^2 \\ \gamma_1^4 & \gamma_2^4 & \gamma_3^4 \end{pmatrix}$$

then its determinant is

$$\gamma_1\gamma_2\gamma_3 \left(\gamma_1 \oplus \gamma_2\right)\left(\gamma_1 \oplus \gamma_3\right)\left(\gamma_2 \oplus \gamma_3\right)\left(\gamma_1 \oplus \gamma_2 \oplus \gamma_2\right).$$

Then, code $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is $(1;\underline{s})$-erasure correcting if and only if the determinant $\det(\hat{\underline{c}})$ is nonzero, if and only if the XOR of any subset of the elements of $\underline{w}_0$ is nonzero. Since $\underline{w}_0$ has s elements, we may assume that if we XOR a number elements smaller than s, the result is true by induction, so assume that we take the XOR of all the s elements in $\underline{w}_0$. This XOR corresponds to the left-hand side of (8). Then, code $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is $(1;\underline{s})$-erasure correcting if and only if code $\mathcal{C}(m,n,1,s-1;f_\alpha(x))$ is $(1;\underline{s}')$-erasure correcting for every $\underline{s}'$ obtained by subtracting 1 from an entry of $\underline{s}$ and (8) holds. ■

Lemma 4.2: Consider a code $\mathcal{C}(m,n,1,s;f_\alpha(x))$ and let $s_j \geq 1$ for $1 \leq j \leq t$ such that $\sum_{j=1}^{t} s_j = s$. For each s_j, if s_j is odd, let $s_j' = s_j$, while if s_j is even, $s_j' = s_j - 1$. Let $s' = \sum_{j=1}^{t} s_j'$. Then, $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is $(1;s_1,s_2,\ldots,s_t)$-erasure correcting if and only if $\mathcal{C}(m,n,1,s';f_\alpha(x))$ is $(1;s_1',s_2',\ldots,s_t')$-erasure correcting.

Proof: We will prove that if s_v is even, then $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is $(1;s_1,s_2,\ldots,s_t)$-erasure correcting if and only if $\mathcal{C}(m,n,1,s-1;f_\alpha(x))$ is $(1;s_1,s_2,\ldots,s_{v-1},s_v',s_{v+1},\ldots,s_t)$-erasure correcting. The result then follows by induction on the number of even coordinates s_v.

Consider (8). If s_j is odd

$$\bigoplus_{u=1}^{s_j} \left(1 \oplus \alpha^{l_{j,u}-l_{j,0}}\right) = 1 \oplus \bigoplus_{u=1}^{s_j} \alpha^{l_{j,u}-l_{j,0}} \tag{9}$$

while if s_j is even

$$\begin{aligned} \bigoplus_{u=1}^{s_j} \left(1 \oplus \alpha^{l_{j,u}-l_{j,0}}\right) &= \bigoplus_{u=1}^{s_j} \alpha^{l_{j,u}-l_{j,0}} \\ &= \alpha^{l_{j,1}-l_{j,0}} \left(1 \oplus \bigoplus_{u=2}^{s_j} \alpha^{l_{j,u}-l_{j,1}}\right). \end{aligned} \tag{10}$$

Assume that s_1 is even; then, $s_1' = s_1 - 1$. According to (10) and (9), the left-hand side of (8) becomes

$$\begin{aligned} &= \alpha^{l_{1,1}-l_{1,0}} \left\{ \bigoplus_{u=2}^{s_1} \left(1 \oplus \alpha^{l_{1,u}-l_{1,0}}\right) \oplus \right. \\ &\qquad \left. \left[\bigoplus_{j=2}^{t} \alpha^{i_j n + l_{j,0} - l_{1,1}} \bigoplus_{u=1}^{s_j} \left(1 \oplus \alpha^{l_{j,u}-l_{j,0}}\right) \right] \right\}. \end{aligned} \tag{11}$$

Since $\alpha^{l_{1,1}-l_{1,0}}$ is always nonzero, by (8) and (11), the second condition in Lemma 4.1 is equivalent for both $\mathcal{C}(m,n,1,s-1;f_\alpha(x))$ and $\mathcal{C}(m,n,1,s;f_\alpha(x))$. So, $\mathcal{C}(m,n,1,s-1;f_\alpha(x))$ is $(1;s_1',s_2,\ldots,s_t)$-erasure correcting

if and only if $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is $(1;s_1,s_2,\ldots,s_t)$-erasure correcting by induction on s.

Similarly, if $2 \leq v \leq t$ and s_v even, then $s'_v = s_v - 1$. According to (10) and (9), (8) becomes

$$= \left[\bigoplus_{u=1}^{s_1}\left(1 \oplus \alpha^{l_{1,u}-l_{1,0}}\right)\right] \oplus \left[\alpha^{i_v n+l_{v,1}-l_{1,0}}\bigoplus_{u=2}^{s_v}\left(1 \oplus \alpha^{l_{v,u}-l_{v,1}}\right)\right] \oplus \left[\bigoplus_{\substack{j=2\\ j\neq v}}^{t}\alpha^{i_j n+l_{j,0}-l_{1,0}}\bigoplus_{u=1}^{s_j}\left(1 \oplus \alpha^{l_{j,u}-l_{j,0}}\right)\right]. \quad (12)$$

By (8) and (12), again the second condition in Lemma 4.1 is equivalent for both $\mathcal{C}(m,n,1,s-1;f_\alpha(x))$ and $\mathcal{C}(m,n,1,s;f_\alpha(x))$; thus, $\mathcal{C}(m,n,1,s-1;f_\alpha(x))$ is $(1;s_1,s_2,\ldots,s_{v-1},s'_v,s_{v+1}\ldots,s_t)$-erasure correcting if and only if $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is $(1;s_1,s_2,\ldots,s_v,\ldots,s_t)$-erasure correcting again by induction on s. ■

Lemma 4.3: Consider a code $\mathcal{C}(m,n,1,s;f_\alpha(x))$ and let $s_j \geq 1$, each s_j an odd number for $1 \leq j \leq t$ such that $\sum_{j=1}^{t} s_j = s$. Then, $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is $(1;s_1,s_2,\ldots,s_t)$-erasure correcting if and only if

1) code $\mathcal{C}(m,n,1,s-1;f_\alpha(x))$ is $(1;\underline{s}')$-erasure correcting for every $\underline{s}'$ obtained by subtracting 1 from an entry of $\underline{s}$; and
2) for any $0 = i_1 < i_2 < i_3 < \cdots < i_t \leq m-1$, for each $1 \leq j \leq t$ and for any $0 \leq l_{j,0} < l_{j,1} < \cdots < l_{j,s_j} \leq n-1$

$$\bigoplus_{j=1}^{t}\alpha^{i_j n+l_{j,0}-l_{1,0}}\left(1 \oplus \bigoplus_{u=1}^{s_j}\alpha^{l_{j,u}-l_{j,0}}\right) \neq 0. \quad (13)$$

Proof: Notice that in this case, (8) becomes (13). ■

The combination of Lemmas 4.1, 4.2, and 4.3 gives the following theorem.

Theorem 4.1: For $s \geq 1$, code $\mathcal{C}(m,n,1,s;f_\alpha(x))$ as given by Construction 3.1 is PMDS if and only if:

1) code $\mathcal{C}(m,n,1,s-1;f_\alpha(x))$ is PMDS; and
2) for every $(s_1,s_2,\ldots,s_t)$ such that $\sum_{j=1}^{t} s_j = s$ and each s_j is odd, for any $0 = i_1 < i_2 < i_3 < \cdots < i_t \leq m-1$, and for any $1 \leq j \leq t$ and $0 \leq l_{j,0} < l_{j,1} < \cdots < l_{j,s_j} \leq n-1$, condition (13) holds.

Theorem 4.1 gives us conditions to check in order to determine if a code $\mathcal{C}(m,n,1,s;f_\alpha(x))$ as given by Construction 3.1 is PMDS, but by itself it does not provide us with any family of PMDS codes. Next we give such a family.

Assume that $f_\alpha(x) = M_p(x)$, where p is a prime number and $M_p(x) = 1 + x + \cdots + x^{p-1}$. In particular, $\mathcal{O}(\alpha) = p$. When 2 is primitive in $GF(p)$, the polynomial $M_p(x)$ is irreducible [23]. For example, $M_5(x)$ is irreducible but $M_7(x)$ is not. Notice that the powers of α obtained by expanding the left-hand side of (13) are at most $mn-1 < p-1 = \deg(M_p(x))$. Then, this expression must be nonzero; otherwise, we would contradict the fact that $M_p(x)$ is the minimal polynomial of α. Therefore, by Theorem 4.1, the code is PMDS. Let us state this fact as a theorem, which provides a family of PMDS codes (it is not known whether the number of irreducible polynomials $M_p(x)$ is infinite).

Theorem 4.2: Consider the code $\mathcal{C}(m,n,1,s;f_\alpha(x))$ given by Construction 3.1 such that $f_\alpha(x) = M_p(x)$ (hence, $M_p(x)$ is irreducible, or equivalently, 2 is primitive in $GF(p)$). Then, $\mathcal{C}(m,n,1,s;f_\alpha(x))$ is PMDS.

Although our results have been given in the finite field $GF(2^b)$, Theorem 4.1 is also valid in the ring of polynomials modulo $M_p(x)$, even if $M_p(x)$ is not irreducible. Such a ring was used to construct the Blaum–Roth (BR) codes [6] to deal with symbols of large size by replacing lookup tables by XOR operations. We omit the details and refer the reader to [4].

So far, we have dealt with general values of s. In the next section, we examine special cases that are important in applications.

V. Special Cases

In this section, we examine several special cases separately.

A. Case $\mathcal{C}(m,n,1,1;f_\alpha(x))$

Notice that $\mathcal{C}(m,n,1,1;f_\alpha(x))$ is always PMDS, since the elements of type $1 \oplus \alpha^j$ for $1 \leq j \leq n-1$ are always nonzero and the result follows from Theorem 4.1. Let us state it as a lemma.

Lemma 5.1: Code $\mathcal{C}(m,n,1,1;f_\alpha(x))$ is always PMDS.

B. Case $\mathcal{C}(m,n,1,2;f_\alpha(x))$

This case is important in applications, in particular, for arrays of SSDs. Since $\mathcal{C}(m,n,1,1;f_\alpha(x))$ is PMDS, Theorem 4.1 gives the following theorem for the case $s = 2$.

Theorem 5.1: Code $\mathcal{C}(m,n,1,2;f_\alpha(x))$ is PMDS if and only if, for any $1 < i \leq m-1$, and for any $0 \leq l_{1,0} < l_{1,1} \leq n-1, 0 \leq l_{2,0} < l_{2,1} \leq n-1$

$$1 \oplus \alpha^{l_{1,1}-l_{1,0}} \oplus \alpha^{in+l_{2,0}-l_{1,0}}\left(1 \oplus \alpha^{l_{2,1}-l_{2,0}}\right) \neq 0. \quad (14)$$

Let us point out that this case was recently studied in [17], where the codes were used in a cloud storage system, the Windows Azure Storage. The authors call their codes local reconstruction codes (LRC) as opposed to PMDS, and they evolved from previous codes, the Pyramid Codes [16]. So, we can see that applications of PMDS codes are not limited to RAID applications for SSDs, as our original motivation was. See also the recent paper [12] (and the references therein), where bounds on codes with local properties (very related to our PMDS codes) are given.

Given the practical importance of this case, let us examine the decoding (of which the encoding is a special case) in some detail.

Consider a PMDS code $\mathcal{C}(m,n,1,2;f_\alpha(x))$, i.e., it satisfies the conditions of Theorem 5.1. Without loss of generality, assume that we either have three erasures in the same row i_0, or two pairs of erasures in different rows i_0 and i_1, where $0 \leq i_0 < i_1 \leq m-1$. Consider first the case in which the three erasures occur in the same row i_0 and in entries j_0, j_1, and j_2 of row $i_0, 0 \leq j_0 < j_1 < j_2$. Assuming initially that $a_{i_0,j_0} = a_{i_0,j_1} = a_{i_0,j_2} = 0$, using (3) and (5) ((4) is used only for $r > 1$), we compute the syndromes S_{i_0}, S_m, and S_{m+1}.

Using the parity-check matrix $\mathcal{H}(m,n,1,2)$ as given by (1), we have to solve the linear system

$$V\begin{pmatrix} a_{i_0,j_0} \\ a_{i_0,j_1} \\ a_{i_0,j_2} \end{pmatrix} = \begin{pmatrix} S_{i_0} \\ S_m \\ S_{m+1} \end{pmatrix}$$

where

$$V = \begin{pmatrix} 1 & 1 & 1 \\ \alpha^{i_0 n+j_0} & \alpha^{i_0 n+j_1} & \alpha^{i_0 n+j_2} \\ \alpha^{2(i_0 n+j_0)} & \alpha^{2(i_0 n+j_1)} & \alpha^{2(i_0 n+j_2)} \end{pmatrix}.$$

Since matrix V is a Vandermonde matrix

$$\det V = \alpha^{3i_0 n+2j_0+j_1}\left(1 \oplus \alpha^{j_1-j_0}\right)\left(1 \oplus \alpha^{j_2-j_0}\right)\left(1 \oplus \alpha^{j_2-j_1}\right),$$

which is easily inverted in a field.

The encoding is a special case of the decoding. For instance, assume that we place the two global parities in locations $(m-1,n-3)$ and $(m-1,n-2)$, as depicted in Fig. 1. After computing the parities $a_{i,n-1}$ for $0 \le i \le m-2$ using single parity, we have to compute the parities $a_{m-1,n-3}, a_{m-1,n-2}$ and $a_{m-1,n-1}$ using the aforementioned method. In particular, the Vandermonde determinant becomes (making $i_0 = m-1, j_0 = n-3, j_1 = n-2$ and $j_2 = n-1$) $\alpha^{3mn-8}(1 \oplus \alpha)\left(1 \oplus \alpha^2\right)(1 \oplus \alpha) = \alpha^{3mn-8}\left(1 \oplus \alpha^4\right)$. So, we have to invert only $\alpha^{3mn-8}\left(1 \oplus \alpha^4\right)$ and some operations may be precalculated, making the encoding very efficient. We omit the details.

We analyze next the case of two pairs of erasures in rows i_0 and $i_1, 0 \le i_0 < i_1 \le m-1$, and assume that the erased entries are a_{i_0,j_0} and a_{i_0,j_1} in row $i_0, 0 \le j_0 < j_1 \le n-1$, and a_{i_1,ℓ_0} and a_{i_1,ℓ_1} in row $i_1, 0 \le \ell_0 < \ell_1 \le n-1$.

Again, using the parity-check matrix $\mathcal{H}(m,n,1,2)$, we have to solve the linear system of four equations with four unknowns

$$W\begin{pmatrix} a_{i_0,j_0} \\ a_{i_0,j_1} \\ a_{i_1,\ell_0} \\ a_{i_1,\ell_1} \end{pmatrix} = \begin{pmatrix} S_{i_0} \\ S_{i_1} \\ S_m \\ S_{m+1} \end{pmatrix}$$

where

$$W = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ \alpha^{i_0 n+j_0} & \alpha^{i_0 n+j_1} & \alpha^{i_1 n+\ell_0} & \alpha^{i_1 n+\ell_1} \\ \alpha^{2(i_0 n+j_0)} & \alpha^{2(i_0 n+j_1)} & \alpha^{2(i_1 n+\ell_0)} & \alpha^{2(i_1 n+\ell_1)} \end{pmatrix}.$$

S_{i_0} and S_{i_1} are given by (3) and S_m and S_{m+1} are given by (5). In order to solve this linear system, we need to invert $\det W$. The code was chosen to be PMDS, so this determinant is invertible. We omit the details.

Table I gives some concrete PMDS codes $\mathcal{C}(m,n,1,2;f_\alpha(x))$. There, we give the value $b, f_\alpha(x)$ (in octal notation), $\mathcal{O}(\alpha)$, and values m and n for which the code $\mathcal{C}(m,n,1,2;f_\alpha(x))$ is PMDS according to Theorem 5.1. For extensive tables of binary irreducible polynomials, see [32].

TABLE I
SOME VALUES OF $b, f_\alpha(x), \mathcal{O}(\alpha), m$ AND n FOR WHICH CODES $\mathcal{C}(m,n,1,2;f_\alpha(x))$ ARE PMDS

b	$f_\alpha(x)$	$\mathcal{O}(\alpha)$	m	n
8	4 3 5	255	5	5
	5 6 7	85	7	5
	4 3 3	51	10	5
9	1 0 2 1	511	20	6
	1 2 3 1	73	10	7
10	3 0 2 5	1023	21	6
			15	7
11	6 0 1 5	2047	29	6
			25	7
			22	8
	5 3 6 1	2047	13	10
12	1 5 6 4 7	4095	67	6
			58	7
			50	8
			24	9
			22	10
16	2 2 7 2 1 5	13107	404	6
			346	7
			303	8
			269	9
			242	10
			164	11
			160	12
			59	16
			45	17
			53	18
			24	20
			19	22
			21	23
			18	24
			17	25
			16	26

We also applied the method to search for PMDS codes over the ring of polynomials modulo $M_p(x) = 1 + x + x^2 + \cdots + x^{p-1}$, p prime, when $M_p(x)$ is not irreducible, and tables of such PMDS codes are given in [4].

C. The Case $\mathcal{C}(m,n,1,3;f_\alpha(x))$

There are two ways to obtain $s = 3$ as a sum of odd numbers: one is 3 itself, the other is $1+1+1$. Then, by Theorem 4.1, we have the following.

Theorem 5.2: Code $\mathcal{C}(m,n,1,3;f_\alpha(x))$ is PMDS if and only if code $\mathcal{C}(m,n,1,2;f_\alpha(x))$ is PMDS, and for $1 \le l_1 < l_2 < l_3 \le n-1$

$$1 \oplus \alpha^{l_1} \oplus \alpha^{l_2} \oplus \alpha^{l_3} \neq 0 \tag{15}$$

and, for any $1 \le i_2 < i_3 \le m-1, 0 \le l_{1,0} < l_{1,1} \le n-1, 0 \le l_{2,0} < l_{2,1} \le n-1$ and $0 \le l_{3,0} < l_{3,1} \le n-1$

$$1 \oplus \alpha^{l_{1,1}-l_{1,0}} \oplus \alpha^{i_2 n+l_{2,0}-l_{1,0}}\left(1 \oplus \alpha^{l_{2,1}-l_{2,0}}\right) \oplus \alpha^{i_3 n+l_{3,0}-l_{1,0}}\left(1 \oplus \alpha^{l_{3,1}-l_{3,0}}\right) \neq 0. \tag{16}$$

The case $\mathcal{C}(m,n,1,4;f_\alpha(x))$ is given in [4].

D. Case $\mathcal{C}(m,n,r,1;f_\alpha(x))$

So far, in this section, we have considered cases in which $r = 1$. If $r = s = 1$, we have seen in Section V-A that the code is PMDS, so we examine here the case $r > 1$. Thus, assume that row $i, 0 \le i \le m-1$, has $r+1$ erasures in locations $0 \le j_0 < j_1 < \cdots < j_r \le n-1$. The following theorem is given without proof (it is proven similarly to the previous cases by examining determinants):

Theorem 5.3: Consider code $\mathcal{C}(m,n,r,1;f_\alpha(x))$. If r is even, then $\mathcal{C}(m,n,r,1;f_\alpha(x))$ is PMDS if and only if $\mathcal{C}(m,n,r-1,1;f_\alpha(x))$ is PMDS, while if r is odd, $\mathcal{C}(m,n,r,1;f_\alpha(x))$ is PMDS if and only if $\mathcal{C}(m,n,r-1,1;f_\alpha(x))$ is PMDS and for any $1 \le l_1 < l_2 < \cdots < l_r \le n-1$

$$1 \oplus \bigoplus_{u=1}^{r} \alpha^{l_u} \neq 0. \tag{17}$$

Since $\mathcal{C}(m,n,1,1;f_\alpha(x))$ is PMDS, by Theorem 5.3, also $\mathcal{C}(m,n,2,1;f_\alpha(x))$ is PMDS. According to (17), $\mathcal{C}(m,n,3,1;f_\alpha(x))$ and $\mathcal{C}(m,n,4,1;f_\alpha(x))$ are PMDS if and only if, for any $1 \le l_1 < l_2 < l_3 \le n-1$, (15) holds.

E. Case $\mathcal{C}^{(1)}(m,n,r,1;f_\alpha(x))$

Consider code $\mathcal{C}^{(1)}(m,n,r,1;f_\alpha(x))$ as given by Construction 3.2. Using the parity-check matrix $\mathcal{H}^{(1)}(m,n,r,1)$ as defined by (6), $\mathcal{C}^{(1)}(m,n,r,1;f_\alpha(x))$ is $(r;1)$-erasure correcting if and only if, for any $0 \le i \le m-1$ and for any $1 \le j_0 < j_1 < \ldots < j_r$, the Vandermonde determinant

$$\det\begin{pmatrix} 1 & 1 & \ldots & 1 \\ \alpha^{in+j_0} & \alpha^{in+j_1} & \ldots & \alpha^{in+j_r} \\ \alpha^{2(in+j_0)} & \alpha^{2(in+j_1)} & \ldots & \alpha^{2(in+j_r)} \\ \alpha^{3(in+j_0)} & \alpha^{3(in+j_1)} & \ldots & \alpha^{3(in+j_r)} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{r(in+j_0)} & \alpha^{r(in+j_1)} & \ldots & \alpha^{r(in+j_r)} \end{pmatrix} = \prod_{0\le t<l\le r} \left(\alpha^{in+j_t} \oplus \alpha^{in+j_l}\right)$$

is invertible. Since this is always the case, we have the following theorem.

Theorem 5.4: Code $\mathcal{C}^{(1)}(m,n,r,1;f_\alpha(x))$ is PMDS.

Comparing Theorems 5.3 and 5.4, we conclude that codes $\mathcal{C}^{(1)}(m,n,r,1;f_\alpha(x))$ are preferable to codes $\mathcal{C}(m,n,r,1;f_\alpha(x))$ for $r \ge 2$, since the former are PMDS without restrictions.

The case $\mathcal{C}(m,n,2,2;f_\alpha(x))$ gives a quite complicated expression [4] and is omitted here.

VI. Simplified Construction

In this section, we present a construction that is an alternative to codes $\mathcal{C}(m,n,1,s;f_\alpha(x))$ for $1 \le s \le 2$. In the case of $s=2$, the new construction can correct the situation depicted at the left of Fig. 2, that is, two pairs of erasures in two different rows. It cannot correct the situation at the right of Fig. 2, i.e., three erasures in the same row. This is a tradeoff, since the new construction, as we will see, uses a smaller finite field. Explicitly, we have the following.

Construction 6.1: Let $\alpha \in GF(2^b)$ and let $\max\{m,n\} \le \mathcal{O}(\alpha)$. Let $\mathcal{C}^{(2)}(m,n,1,2;f_\alpha(x))$ be the code whose $(m+2)\times mn$ parity-check matrix is

$$\mathcal{H}^{(2)}(m,n,1,2) = \left(\begin{array}{ccc|ccc|c|ccc} 1 & 1 & \ldots\ 1 & 0 & 0 & \ldots\ 0 & \ldots & 0 & 0 & \ldots\ 0 \\ 0 & 0 & \ldots\ 0 & 1 & 1 & \ldots\ 1 & \ldots & 0 & 0 & \ldots\ 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \ldots\ 0 & 0 & 0 & \ldots\ 0 & \ldots & 1 & 1 & \ldots\ 1 \\ \hline 1 & \alpha & \ldots\ \alpha^{n-1} & 1 & \alpha & \ldots\ \alpha^{n-1} & \ldots & 1 & \alpha & \ldots\ \alpha^{n-1} \\ 1 & \alpha & \ldots\ \alpha^{n-1} & \alpha & \alpha^2 & \ldots\ \alpha^n & \ldots & \alpha^{m-1} & \alpha^{m-2} & \ldots\ \alpha^{m+n-2} \end{array}\right).$$

$\mathcal{C}^{(2)}(m,n,1,1;f_\alpha(x))$ is the code whose $(m+1)\times mn$ parity-check matrix is given by the first $m+1$ rows of $\mathcal{H}^{(2)}(m,n,1,2)$.

The following example illustrates Construction 6.1.

Example 6.1: Consider codes $\mathcal{C}^{(2)}(3,5,1,2;f_\alpha(x))$ and $\mathcal{C}^{(2)}(5,3,1,2;f_\alpha(x))$, where $f_\alpha(x) = M_5(x)$. Then, since $\alpha^5 = 1$, their respective parity-check matrices are

$$\mathcal{H}^{(2)}(3,5,1,2) = \left(\begin{array}{ccccc|ccccc|ccccc} 1&1&1&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&1&1&1&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1 \\ \hline 1&\alpha&\alpha^2&\alpha^3&\alpha^4&1&\alpha&\alpha^2&\alpha^3&\alpha^4&1&\alpha&\alpha^2&\alpha^3&\alpha^4 \\ 1&\alpha&\alpha^2&\alpha^3&\alpha^4&\alpha&\alpha^2&\alpha^3&\alpha^4&1&\alpha^2&\alpha^3&\alpha^4&1&\alpha \end{array}\right)$$

and

$$\mathcal{H}^{(2)}(5,3,1,2) = \left(\begin{array}{ccc|ccc|ccc|ccc|ccc} 1&1&1&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&1&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&1&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&1&1&1 \\ \hline 1&\alpha&\alpha^2&1&\alpha&\alpha^2&1&\alpha&\alpha^2&1&\alpha&\alpha^2&1&\alpha&\alpha^2 \\ 1&\alpha&\alpha^2&\alpha&\alpha^2&\alpha^3&\alpha^2&\alpha^3&\alpha^4&\alpha^3&\alpha^4&1&\alpha^4&1&\alpha \end{array}\right).$$

The following lemma is immediate.

Lemma 6.1: The code $\mathcal{C}^{(2)}(m,n,1,1;f_\alpha(x))$ given by Construction 6.1 is PMDS.

Comparing Lemmas 5.1 and 6.1, both $\mathcal{C}(m,n,1,1;f_\alpha(x))$ and $\mathcal{C}^{(2)}(m,n,1,1;f_\beta(x))$ are PMDS. However, the conditions on $\mathcal{C}^{(2)}(m,n,1,1;f_\beta(x))$ are less stringent. For example, assume that we consider $\mathcal{C}(7,7,1,1;f_\alpha(x))$ and $\mathcal{C}^{(2)}(7,7,1,1;f_\beta(x))$. For $\mathcal{C}(7,7,1,1;f_\alpha(x))$, this means $49 \le \mathcal{O}(\alpha)$, and if we take a finite field $GF(2^b)$, the smallest field we could use is $GF(64)$. For $\mathcal{C}^{(2)}(7,7,1,1;f_\beta(x))$, on the other hand, we require $7 \le \mathcal{O}(\beta)$; thus, we can use the finite field $GF(8)$ with β primitive in $GF(8)$. Although we are using a smaller field, the PMDS property is not lost in $\mathcal{C}^{(2)}(m,n,1,1;f_\beta(x))$. This is not the case for codes $\mathcal{C}^{(2)}(m,n,1,2;f_\alpha(x))$: since the last two rows of the 3×3 matrix necessary to recover from three erasures in the same row are linearly dependent, the codes are not (1;2)-erasure correcting (and hence are not PMDS). However, they are (1;1,1)-erasure correcting, as stated in the following lemma.

Lemma 6.2: The code $\mathcal{C}^{(2)}(m,n,1,2;f_\alpha(x))$ given by Construction 6.1 is (1;1,1)-erasure correcting.

Proof: Assume that we have two erasures in locations j_0 and j_1 of row i_0 and two erasures in locations ℓ_0 and ℓ_1 of row i_1, where $0 \le i_0 < i_1 \le m-1, 0 \le j_0 < j_1 \le n-1$ and $0 \le \ell_0 < \ell_1 \le n-1$. Using the parity-check matrix $\mathcal{H}^{(2)}(m,n,1,2)$ as given in Construction 6.1, these four erasures can be recovered if and only if

$$\det\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ \alpha^{j_0} & \alpha^{j_1} & \alpha^{\ell_0} & \alpha^{\ell_1} \\ \alpha^{i_0+j_0} & \alpha^{i_0+j_1} & \alpha^{i_1+\ell_0} & \alpha^{i_1+\ell_1} \end{pmatrix} \ne 0.$$

By row operations, we find out that this determinant is nonzero if and only if $1 \oplus \alpha^{j_1-j_0}, 1 \oplus \alpha^{\ell_1-\ell_0}$, and $1 \oplus \alpha^{i_1-i_0}$ are nonzero, which is certainly the case since $j_1-j_0, \ell_1-\ell_0$ and i_1-i_0 are smaller than $\mathcal{O}(\alpha)$. ■

Lemma 6.2 is important in applications. Let us compare it with $\mathcal{C}(m,n,1,2;f_\alpha(x))$ codes that are PMDS as given in Section V-B. For the sake of discussion, let us assume that $\max\{m,n\} \leq 15$, a situation that covers some practical applications. In the case of $\mathcal{C}(m,n,1,2;f_\alpha(x))$, if $m \geq 9$ and $n = 15$, we would need to operate on the field $GF(2^8)$. If we use a code $\mathcal{C}^{(2)}(m,n,1,2;f_\beta(x))$, we can take the finite field $GF(2^4)$. If we use $GF(2^5)$, we can increase m to 16, a value convenient in applications. So, there is a tradeoff here: the code is not PMDS, but it requires a much smaller field, simplifying implementation.

Let us point out that Construction 6.1 is closely related to generalized concatenated (GC) codes [7]. Implementations of GC codes are given by the two-level [14], [15] and the multilevel [30] integrated interleaving schemes.

Using ideas similar to the ones of GC codes we can extend Construction 6.1 to codes that are $(1;\overbrace{1,1,\ldots,1}^{s})$-erasure correcting (that we denote $\mathcal{C}^{(2)}(m,n,1,s;f_\alpha(x))$) as well as other combinations by using horizontal and vertical codes, but for reasons of space we omit them here. Moreover, as shown in [4], there is not much gain for codes $\mathcal{C}^{(2)}(m,n,1,s;f_\alpha(x))$ and $s \geq 3$ with respect to codes $\mathcal{C}^{(2)}(m,n,1,2;f_\alpha(x))$ in a mixed environment of catastrophic failures and hard errors.

Finally, consider the code $\mathcal{C}^{(2)}(m,n,r,1;f_\alpha(x)), \mathcal{O}(\alpha) \geq \max\{m,n\}$, whose parity-check matrix is the $(mr+1)\times mn$ matrix

$$\left(\begin{array}{c|c|c|c} H^{(1)}(n,r,0,0) & \underline{0}(n,r) & \ldots & \underline{0}(n,r) \\ \underline{0}(n,r) & H^{(1)}(n,r,0,0) & \ldots & \underline{0}(n,r) \\ \vdots & \vdots & \ddots & \vdots \\ \underline{0}(n,r) & \underline{0}(n,r) & \ldots & H^{(1)}(n,r,0,0 \\ \hline H^{(1)}(n,1,r,0) & H^{(1)}(n,1,r,0) & \ldots & H^{(1)}(n,1,r,0) \end{array}\right)$$

where $H^{(1)}(n,r,i,j)$ is given by (7) and $\underline{0}(n,r)$ is an $r\times n$ zero matrix. Correcting $r+1$ erasures in a row of an $m\times n$ array implies inverting a Vandermonde matrix, as done in the case of $\mathcal{C}^{(1)}(m,n,r,1,f_\alpha(x))$, so we have the following theorem.

Theorem 6.1: Code $\mathcal{C}^{(2)}(m,n,r,1;f_\alpha(x))$ is PMDS.

Comparing Theorems 5.4 and 6.1, we conclude that codes $\mathcal{C}^{(2)}(m,n,r,1;f_\alpha(x))$ are preferable to codes $\mathcal{C}^{(1)}(m,n,r,1;f_\alpha(x))$, since they require a smaller field.

VII. Conclusion

We have presented constructions of codes that are suitable for a flash array type of architecture, in which hard errors coexist with catastrophic device failures. We have presented specific codes that are useful in applications. Necessary and sufficient conditions for codes satisfying an optimality criterion were given.

Acknowledgment

The authors would like to thank the three anonymous reviewers for their numerous suggestions.

References

[1] M. Balakrishnan, A. Kadav, V. Prabhakaran, and D. Malkhi, "Differential RAID: Rethinking RAID for SSD reliability," *ACM Trans. Storage*, vol. 6, no. 2, Jul. 2010.

[2] M. Blaum, J. Brady, J. Bruck, and J. Menon, "EVENODD: An efficient scheme for tolerating double disk failures in RAID architectures," *IEEE Trans. Comput.*, vol. C-44, no. 2, pp. 192–202, Feb. 1995.

[3] M. Blaum, P. G. Farrell, and H. C. A. van Tilborg, "Array codes," in *Handbook of Coding Theory*, V. S. Pless and W. C. Huffman, Eds. New York, NY, USA: Elsevier, 1998, ch. 22.

[4] M. Blaum, J. L. Hafner, and S. Hetzler, "Partial-MDS codes and their application to RAID type of architectures," IBM, New York, NY, USA, 2012, Res. Rep. RJ10498.

[5] M. Blaum and R. J. McEliece, "Coding protection for magnetic tapes: A generalization of the Patel-Hong code," *IEEE Trans. Inf. Theory*, vol. IT-31, no. 5, pp. 690–693, Sep. 1985.

[6] M. Blaum and R. M. Roth, "New array codes for multiple phased burst correction," *IEEE Trans. Inf. Theory*, vol. IT-39, no. 1, pp. 66–77, Jan. 1993.

[7] E. L. Blokh and V. V. Zyablov, "Coding of generalized concatenated codes," *Problemy Peredachii Inf.*, vol. 10, no. 3, pp. 218–222, 1974.

[8] P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S. Sankar, "Row-diagonal parity for double disk failure correction," in *Proc. 3rd Conf. File Storage Technol.*, San Francisco, CA, USA, Mar.–Apr. 2004.

[9] P. Delsarte, "Bilinear forms over a finite field, with applications to coding theory," *J. Combin. Theory Ser. A*, vol. 25, pp. 226–241, 1978.

[10] E. M. Gabidulin, "Theory of codes with maximum rank distance," *Prob. Inf. Transmiss.*, vol. 21, no. 1, pp. 3–16, 1985.

[11] G. A. Gibson, *Redundant Disk Arrays*. New York, NY, USA: MIT Press, 1992.

[12] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the locality of codeword symbols," *IEEE Trans. Inf. Theory*, vol. IT-58, no. 11, pp. 6925–6934, Nov. 2012.

[13] K. M. Greenan, D. D. Long, E. L. Miller, T. J. E. Schwarz, and A. Wildani, "Building flexible, fault-tolerant flash-based storage systems," presented at the 5th Workshop Hot Topics Dependability, Lisbon, Portugal, Jun. 2009.

[14] J. Han and L. A. Lastras-Montaño, "Reliable memories with subline accesses," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2007, pp. 2531–2535.

[15] M. Hassner, K. Abdel-Ghaffar, A. Patel, R. Koetter, and B. Trager, "Integrated interleaving—A novel ECC architecture," *IEEE Trans. Magn.*, vol. 37, no. 2, pp. 773–775, Mar. 2001.

[16] C. Huang, M. Chen, and J. Li, "Pyramid codes: flexible schemes to trade space for access efficiency in reliable data storage systems," in *Proc. IEEE Netw. Comput. Appl.*, Cambridge, MA, USA, Jul. 2007, pp. 79–86.

[17] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, and S. Yekhanin, "Erasure coding in Windows Azure storage," presented at the USENIX Annu. Tech. Conf., Boston, MA, USA, Jun. 2012.

[18] W. Hutsell, An in-depth look at the RamSan-500 cached flash solid state disk [Online]. Available: http://www.texmemsys.com/files/f000233.pdf

[19] S. Im and D. Shin, "Flash-aware RAID techniques for dependable and high-performance flash memory SSD," *IEEE Trans. Comput.*, vol. C-60, no. 1, pp. 80–92, Jan. 2011.

[20] L. A. Lastras-Montaño, P. J. Meaney, E. Stephens, B. M. Trager, J. O'Connor, and L. C. Alves, "A new class of array codes for memory storage," presented at the Inf. Theory Appl. Workshop, La Jolla, CA, USA, Feb. 2011.

[21] M. Li and J. Shu, "C-Codes: Cyclic lowest-density MDS array codes constructed using starters for RAID 6," IBM, New York, NY, USA, 2011, Res. Rep. RC25218.

[22] S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2004.

[23] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam, The Netherlands: North Holland, 1977.

[24] N-29–17: NAND flash design and use considerations introduction Micron [Online]. Available: http://download.micron.com/pdf/technotes/nand/tn2917.pdf

[25] A. Park, D. Lee, Y. Woo, G. Lee, J. Lee, and D. Kim, "Reliability and performance enhancement technique for SSD array storage system using RAID mechanism," in *Proc. 9th Int. Conf. Commun. Inf. Technol.*, 2009, pp. 140–145.

[26] J. S. Plank, "A tutorial on Reed-Solomon coding for fault-tolerance in RAID-like systems," *Softw.—Practice Exp.*, vol. 27, no. 9, pp. 995–1012, Sep. 1997.

[27] J. S. Plank, "The RAID-6 liberation codes," in *Proc. 6th USENIX Conf. File Storage Technol.*, San Francisco, CA, USA, 2008, pp. 97–110.

[28] J. S. Plank, M. Blaum, and J. L. Hafner, "SD codes: Erasure codes designed for how storage systems really fail," presented at the 13th Conf. File Storage Technol., San Jose, CA, USA, Feb. 2013.

[29] R. M. Roth, "Maximum rank array codes and their application to criss-cross error correction," *IEEE Trans. Inf. Theory*, vol. IT-37, no. 2, pp. 328–336, Mar. 1991.
[30] X. Tang and R. Koetter, "A novel method for combining algebraic decoding and iterative processing," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2006, pp. 474–478.
[31] A. Thomasian and M. Blaum, "Higher reliability redundant disk arrays: Organization, operation, and coding," *ACM Trans. Storage*, vol. 5, no. 3, Nov. 2009.
[32] W. Wesley Peterson and E. J. Weldon Jr., *Error-Correcting Codes*, 2nd ed. New York, NY, USA: MIT Press, 1984.
[33] L. Xu, V. Bohossian, J. Bruck, and D. G. Wagner, "Low-density MDS codes and factors of complete graphs," *IEEE Trans. Inf. Theory*, vol. IT-45, no. 6, pp. 1817–1826, Sep. 1999.
[34] L. Xu and J. Bruck, "X-code: MDS array codes with optimal encoding," *IEEE Trans. Inf. Theory*, vol. IT-45, no. 1, pp. 272–276, Jan. 1999.

Mario Blaum (S'84–M'85–SM'92–F'00) received the degree of Licenciado from the University of Buenos Aires in 1978, the M. Sc. degree from the Israel Institute of Technology (Technion) in 1981 and the Ph. D. degree from the California Institute of Technology (Caltech) in 1984, all these degrees in Mathematics. From January to June, 1985, he was a Research Fellow at the Department of Electrical Engineering at Caltech. In August, 1985, he joined the IBM Research Division at the Almaden Research Center. In January, 2003, his division was transferred to Hitachi Global Storage Technologies, where he was a Research Staff Member until 2009. In 2009 he rejoined the IBM Almaden Research Center. From September 1990 to September 1991 he was a Consulting Professor at Stanford University. In 2008 and in 2009 he was a Lecturer at Santa Clara University. At present he is a Consulting Professor at the University Complutense of Madrid, Spain. He was an Associate Editor of the IEEE TRANSACTIONS ON INFORMATION THEORY from 2009 to 2012.

Dr. Blaum's research interests include Storage Technology, comprising all aspects of coding and synchronization. He is a Fellow of the IEEE since 2000.

James Lee Hafner received a Ph.D. degree in mathematics from the University of Illinois, and his undergraduate degree in mathematics at Santa Clara University. He spent time in academia at the Institute for Advanced Study in Princeton, at the California Institute of Technology and as a National Science Foundation Postdoctoral Fellow at UCSD, after which he joined the IBM Research division. At IBM, he has worked in diverse areas, including number theory, complexity theory, image databases, and storage system and storage protocols. Dr. Hafner currently works in the Advanced Storage Technologies department at the IBM Almaden Research Center.

Steven Hetzler (SM'98) is an IBM Fellow and manages the Storage Architecture Research group. He has spent over 25 years in data storage research and development. Most recently, he is developing highly reliable storage system architectures using consumer-grade solid state storage, and novel storage systems for tackling the big data explosion. He developed Chasm Analysis, a methodology for analyzing market potential for storage technologies using economic data. Previously, he initiated work on the IP storage protocol that is now known as iSCSI, which he later named. Steve has a blog on data storage at http://drhetzler.com/smorgastor.

Steve holds a Ph.D. in Applied Physics from the California Institute of Technology. He joined IBM Research in November 1985 and was named an IBM Fellow in 1998. He has been issued 51 patents for inventions covering a wide range of topics –including data storage systems and architectures, optics, error correction coding and power management