J Electron Test (2010) 26:559–580
DOI 10.1007/s10836-010-5168-5

Design of Memories with Concurrent Error Detection and Correction by Nonlinear SEC-DED Codes

Zhen Wang · Mark Karpovsky · Konrad J. Kulikowski

Received: 6 December 2009 / Accepted: 26 July 2010 / Published online: 11 August 2010

Abstract In this paper we propose memory protection architectures based on nonlinear single-error-correcting, double-error-detecting (SEC-DED) codes. Linear SEC-DED codes widely used for design of reliable memories cannot detect and can miscorrect lots of errors with large Hamming weights. This may be a serious disadvantage for many modern technologies when error distributions are hard to estimate and multi-bit errors are highly probable. The proposed protection architectures have fewer undetectable errors and fewer errors that are miscorrected by all codewords than architectures based on linear codes with the same dimension at the cost of a small increase in the latency penalty, the area overhead and the power consumption. The nonlinear SEC-DED codes are generalized from the existing perfect nonlinear codes (Vasil'ev codes, Probl Kibern 8:375–378, 1962; Phelps codes, SIAM J Algebr Discrete Methods 4:398–403, 1983; and the codes based on one switching constructions, Etzion and Vardy, IEEE Trans Inf Theory 40:754–763, 1994). We present the error correcting algorithms, investigate and compare the error detection and correction capabilities of the proposed nonlinear SEC-DED codes to linear extended Hamming codes and show that replacing linear extended Hamming codes by the proposed nonlinear SEC-DED codes results in a drastic improvement in the reliability of the memory systems in the case of repeating errors or high multi-bit error rate. The proposed approach can be applied to RAM, ROM, FLASH and disk memories.

Keywords Coding theory · Reliable memories · SEC-DED · Error detection and correction · Soft errors · Nonlinear codes · Robust

Responsible Editor: S. Hellebrand

Z. Wang (✉) · M. Karpovsky · K. J. Kulikowski
Reliable Computing Laboratory, Boston University,
8 Saint Marys Street, Boston, MA, USA
e-mail: lark@bu.edu

M. Karpovsky
e-mail: markkar@bu.edu

K. J. Kulikowski
e-mail: konkul@bu.edu

1 Introduction

The reliability of memory is a crucial consideration for today's digital devices. For some designs as much as 70% of the chip area is taken by the embedded memory and this number is expected to reach 90% by 2011 [13, 30]. This large area of the chip is especially vulnerable to single-event-upsets (SEUs) caused by single, energetic particles like high-energy neutrons and alpha particles. SEU temporarily alters the state of the devices and results in soft errors. These errors are non-destructive and appear as unwanted bit flips in memory cells and registers. Continuing scaling of device features and performance increases the likelihood of errors, which makes the error models more unpredictable. As the speed of the devices becomes higher the relative size of the clock transition timing window increases and this makes devices more sensitive to SEU [16]. Similarly, decreased voltage levels for modern technologies make bit inversions more likely to occur [9].

The dangers of possible errors in memories resulting from SEUs are often mitigated with the use of linear single-error-correcting, double-error-detecting codes (SEC-DED). These codes have minimum Hamming distance four and are able to correct all single bit errors and detect all double bit errors. In the presence of multi-bit errors, however, the reliability of systems utilizing error protection architectures based on these codes may be questionable. For any linear SEC-DED codes with k information bits, the number of undetectable multi-bit errors is 2^k. In addition to this, a huge number of multi-bit errors will be miscorrected. In the case where SEU results in multi-bit distortions with high probability, these codes may not be sufficient to provide a high reliability. Anomalies of systems caused by multi-bit upsets (MBU) have already been reported [36, 37].

The increase of the MBU rate in deep submicron technologies deteriorates the situation even further. In 65nm triple-well SRAMs with a thin cell architecture, the rate of multi-bit errors caused by neutron induced SEU increases by a factor of ten compared to that in 90 nm technologies—nearly 55% of the errors due to neutron radiation were multi-bit errors [12]. Although there are mechanisms like bit interleaving [27] that can be used to minimize the error rate contribution of multi-bit errors, whether it is enough under such high MBU rate is still unknown. Moreover, the advantage of bit interleaving comes at a price of more layout constraints, which may result in larger power consumptions and longer access times. Thereby, memory protection architectures which can provide better protection against multi-bit errors than that based on classical linear codes are in demand.

Errors that are undetected (miscorrected) by some but not all of the codewords are called *conditionally detectable (miscorrected) errors*. Linear codes do not have conditionally detectable (miscorrected) errors. All errors are either 100% detected (corrected) or not detected (corrected) at all, which is bad for the detection (correction) of *repeating errors*, since an error e will always be masked (miscorrected) as long as it is masked (miscorrected) for one single message. Repeating errors can occur in many situations. In [25], it was shown that transient faults lasting for more than one clock cycle are possible for new technologies. If a SEU lasts for several consecutive READ/WRITE cycles, it is possible that different messages written into the same memory cell are affected by the same error pattern. Another example of repeating errors is a hard error caused by a permanent fault in the device that is unrecoverable by re-writing. These errors may repeat themselves until the memory is replaced. For memories with repeating errors, more powerful error correcting codes are required.

In this paper we analyze the limitations of existing error correcting architectures for memories and show how nonlinear robust codes can be applied to make memories more reliable in the presence of unpredictable environments where the error distributions are unknown or not stationary.

For any systematic code C, let f be the encoding function mapping the information bits x in a codeword onto redundant bits. C is nonlinear if there exists nonzero error vector e such that there is at least one x satisfying $f(x \oplus e) \neq f(x) \oplus f(e)$, where $\oplus$ is the binary addition operation. If such x exists for every nonzero error vector e, C is a nonlinear robust code and has no undetectable errors. Otherwise, C is a nonlinear partially robust code, which has much smaller number of undetectable errors but the same length and the number of information bits as linear codes.

We propose several architectures based on nonlinear SEC-DED partially robust codes, i.e. extended Vasil'ev codes and extended Phelps codes, for single bit error correction in memories. These codes have fewer undetectable errors and fewer multi-bit errors which are always miscorrected while requiring a latency penalty, hardware overhead and power consumption comparable to that of the conventional linear SEC-DED codes. Moreover, extended Vasil'ev codes and extended Phelps codes have conditionally detectable and conditionally miscorrected errors. The detection or correction of these errors is *message dependent*. The more messages the error affects, the smaller the error masking probability is. For memories with repeating errors, extended Vasil'ev codes and extended Phelps codes have more advantages over linear codes. We show that linear extended Hamming codes can be replaced by nonlinear extended Vasil'ev codes or nonlinear extended Phelps codes resulting in improved reliability in the presence of multi-bit distortions or repeating errors.

The rest of the paper is organized as follows. In Section 2, previous work on error correcting codes for design of reliable memories is summarized. In Section 3, we clarify the error model used in this paper. In Section 4, we give the definition of robust codes, partially robust codes, minimum distance robust and partially robust codes and several bounds that can be used to evaluate and compare the error detection capabilities of the codes. In Section 5, several new constructions of minimum distance robust and partially robust codes are presented. The error detection kernels of Hamming, Vasil'ev, Phelps and one switching codes are shown and the advantages of Vasil'ev codes and Phelps codes are demonstrated. In Section 6, we compare three memory architectures based on the linear extended Hamming code, the extended Vasil'ev code and the extended Phelps code and show the benefits of utilizing the extended Vasil'ev code or the extended Phelps code to protect memory devices. The error detection and correction algorithms for the extended Vasil'ev code and the extended Phelps code are given and the latency penalty, the hardware overhead and the power consumption of the three architectures are compared. We conclude the paper in Section 7.

Our initial results on applications of SEC-DED nonlinear codes for design of reliable memories can be found in [40].

2 Previous Work

Since the basic construction of SEC-DED codes was presented by Hamming in 1950 [14], a number of modifications have been proposed. In [15], a class of optimal minimum odd-weight-column SEC-DED codes was constructed for better performance, cost and reliability. To further simplify the encoding and decoding complexity, the author in [24] proposed a coding technique requiring less ones in the parity check matrix than the code presented in [15]. In [2], a hardware efficient method was proposed to construct SEC-DED-AUED systematic codes that can also detect all unidirectional errors. For protecting byte oriented memories, SEC-DED-SBD codes were proposed in [5, 35] and [7]. These codes are known as single-error-correcting, double-error detecting, single-byte-error-detecting codes and are able to detect all single byte errors. SEC-DED-SBD codes that are also able to correct any odd number of erroneous bits per byte were proposed in [31]. To enhance the error correction capability of SEC-DED codes, the author in [8] constructed single-error-correcting, double-error-detecting, double-adjacent-error-correcting (SEC-DED-DAEC) code by selectively avoiding certain types of linear dependencies in the parity check matrix. These codes use the same number of check bits and the similar overhead to other known SEC-DED codes and have the advantage that it can correct all adjacent double errors. In [6], the author constructed single-byte-error-correcting, double-byte-error-detecting codes (SBC-DBD), which can provide complete single byte error correction capabilities. In [23], double-error-correcting and triple-error-detecting code was proposed to correct all double bit errors. The well known Reed-Solomon code, as another example, was utilized in Hubble Space Telescope to protect 16 Mbit DRAMs manufactured by IBM [42].

All the codes mentioned above are linear codes. They concentrate their error detection and correction capabilities on a specific type of errors (e.g. errors with small multiplicities or belonging to the same byte). The reliability of the memory systems based on these codes can not be guaranteed when the MBU rate is high.

Some memory protection architectures based on nonlinear codes have also been proposed in the community. In [3], efficient single error correcting and $d(d \geq 2)$-unidirectional error detecting codes were used to protect memories. Another nonlinear error detecting code—Berger code [1], was used to detect unidirectional errors in flash memories. These existing protection architectures based on nonlinear codes, however, were mainly designed for unidirectional error models. In the presence of symmetric errors, the reliability of the protected memory systems can not be guaranteed.

Nonlinear *robust* codes have been proposed as a solution to the limitation of minimum distance linear error detecting codes in the presence of multi-bit errors. The nonlinear robust codes are designed to provide equal protection against all errors thereby eliminating possible weak areas in the protection. Several variants of robust codes have been proposed. These variants allow tradeoffs in terms of robustness and hardware overhead for many architectures. Robust and partially robust codes have been described in [11, 18, 19]. Robust and partially robust codes with minimum distance larger than two presented in this paper are able to correct errors with small multiplicities and are promising alternatives to linear error correcting codes in applications where protection against multi-bit errors is important. We will overview several constructions for these codes in Section 5.

3 Error Model

In this paper we assume that faults (e.g. SEU, MBU, stuck-at faults, pattern-sensitive faults, dynamic faults, etc.) manifest themselves as additive errors at the output of the memory. Let x be the non-distorted output, e be the error vector and $\tilde{x}$ be the distorted output. Then $\tilde{x} = x \oplus e$, where $\oplus$ is the XOR operation. If faults exist but do not manifest at the output of the memory, e will be the all-zero vector and $\tilde{x} = x$. No harm will be caused by the presence of these faults. The reliability of the memory protection architecture is estimated by analyzing the capabilities of detecting or correcting the additive errors at the output of the device. This error model is commonly used to analyze the performance of error control codes used for the protection of memories (see references in Section 2).

4 Definitions

4.1 Robust and Partially Robust Codes

Throughout the whole paper we denote by "$\oplus$" the component-wise XOR and "$\cdot$" the component-wise AND operation of binary vectors. We denote by (n, k, d) a code of length n, dimension k and minimum distance d, Most of the results can be easily generalized for q-ary case, where $q = p^s$ and p is a prime.

Definition 1 (Kernels of the code) For any error correcting code $C \subseteq GF(2^n)$, the *detection kernel* K_d is the set of errors that are masked by all codewords.

$$K_d = \left\{ e | e \oplus c \in C, \forall c \in C \right\}. \tag{1}$$

It is easy to show that K_d is a linear subspace of C. If C is linear, $K_d = C$. Denote by A the error correction algorithm for code C. Denote by E the set of errors that A attempts to correct. The *correction kernel* K_c is defined as follows:

$$K_c = \left\{ e | e \notin E, \forall c \in C, \exists e^{'} \in E, A(e \oplus c) = A(e^{'} \oplus c) \right\}. \tag{2}$$

Throughout this paper we denote by ω_d the dimension of K_d.

The detection kernels of different codes will be analyzed and compared in this section. The correction kernels, which are related to the error correction algorithms, will be discussed in Section 6.

Example 1 (Kernels of Binary Linear Hamming Codes) A $(n, n - \lceil \log_2(n+1) \rceil, 3)$ linear Hamming code $C \subseteq GF(2^n)$ has minimum distance 3 and is able to correct all single bit errors. Denote by H the parity check matrix of C. An error e is undetectable if and only if e is a codeword ($He = 0$). Thereby the detection kernel K_d of a Hamming code is C itself. For single error correcting codes $E = \{e| \, ||e|| = 1\}$, where $||e||$ is the multiplicity of the error. A multi-bit error e, $||e|| > 1$ will be miscorrected if and only if it has the same syndrome as some single bit error. So the correction kernel of Hamming code is $\{e | He = He_i^*\}$, where e_i^* is an error vector of Hamming weight one and He is the matrix multiplication in binary field. Obviously, K_d and K_c are disjoint. For perfect linear Hamming code, $K_d \bigcup K_c \bigcup E = GF(2^n)$.

A main characteristic of traditional linear error detecting codes is that they concentrate their error detecting power on a small subset of errors which are assumed to be the most likely to occur. Typically, such codes concentrate their error detection on errors of a small multiplicity. They are designed to guarantee detection of all errors with a multiplicity less than d. Error detection beyond the minimum distance of the code is typically not a part of the design criteria and can be unpredictable and ineffective. While for some classes of errors the codes provide 100% protection, for a very large class of errors linear codes offer no protection for all messages. For any linear systematic error detecting code of length n and dimension k there are 2^k undetectable errors. Linear codes have the largest detection kernel K_d (the set of undetectable errors) of any class of systematic codes with the same n and k.

Robust codes are designed to provide guaranteed level of detection against all errors. These codes are characterized by their error masking probability $Q(e)$, which is the fraction of codewords that mask a given error e.

$$Q(e) = \frac{\left| \left\{ c | c \in C, c \oplus e \in C \right\} \right|}{|C|}. \tag{3}$$

Definition 2 The code C is *robust* iff $\max_{e \neq 0} Q(e) < 1$, or equivalently the detection kernel of the code contains only the zero vector $K_d = \{0\}$.

Robust codes are optimal when the maximum $Q(e)$ for all errors is minimized [20]. For a robust code the error masking probability is bounded for nonzero errors. Most robust codes do not have a minimum distance larger than one and do not guarantee 100% detection probability for any subset of errors. A possible variant of the robust codes is to include a minimum distance into the design criteria.

Definition 3 Let $||e||$ denote the multiplicity of an error e. A robust code where $Q(e) = 0$ for all $||e|| < d, e \neq 0$ is a *d-minimum distance robust code.*

Example 2 Consider a 4-bit one hot code $C = \{0001, 0010, 0100, 1000\}$. It is easy to verify that for every nonzero error $e \in GF(2^4)$, there are at most two $c \in C$ satisfying $c \oplus e \in C$. Thereby, $Q(e) = \frac{|\{c|c \in C, c \oplus e \in C\}|}{|C|} \leq 0.5$, $|K_d| = \{0\}$ and C is robust. Moreover, for any single bit error e, there is no $c \in C$ satisfying $c \oplus e \in C$. The code C is a 2-minimum distance robust code.

Minimum distance robust codes are robust codes with a minimum distance larger than one. Since these codes are robust they have no undetectable errors and the worst case error masking probability is bounded by $\max_{e \neq 0} Q(e) < 1$. However, unlike traditional robust codes they also provide a guaranteed 100% probability of detection of errors of small multiplicities ($||e|| < d$). These codes can be useful for providing the highest protection against the most likely or most dangerous threat while maintaining a detection guarantee in case of an unexpected behavior.

For some applications the error characteristics of robust codes can be considered too pessimistic. *Partially robust* codes and *minimum distance partially robust* codes (see Definition 4) allow for a tradeoff among robustness, decoding complexity and overheard, which fill the gap between the optimistic linear codes and pessimistic robust codes. Several constructions of optimal or nearly optimal minimum distance partially robust codes are given in Section 5.2.

Definition 4 A (n, k, d) code with a detection kernel smaller than 2^k is a *partially robust code*. If the code also has a minimum distance greater than one it is referred to as a *minimum distance partially robust code*.

Example 3 The code $C = \{(x, (Px)^3)\}$ where $x \in GF(2^{32})$, P is a 32 by 6 encoding matrix of a shortened (38, 32) Hamming code, and the cubing operation is over $GF(2^6)$, is a binary partially robust code with $\omega_d = 26$ and $\max_{e \notin K_d} Q(e) = 2^{-5}$. C has the same number of redundant bits as the (38, 32) shortened Hamming codes but much less undetectable errors. (For the (38, 32) shortened Hamming code, $\omega_d = 32$.)

Partially robust codes reduce the number of undetectable errors while preserving some structures of linear codes which can be exploited to build efficient prediction hardware that generates redundant bits of a message. Like linear codes, partially robust codes still have undetectable errors (hence they are not completely robust). The number of undetectable errors is reduced by many orders of magnitude compared to that of the linear codes. For practical partially robust constructions, the number of undetectable errors can be reduced from 2^k to 2^{k-r} compared to a linear (n, k, d) code [17]. The probability of masking for the errors that are detectable is bounded by

$$Q_{mc} = \max_{\{e|e \notin K_d\}} Q(e). \tag{4}$$

For memory applications, we are mostly interested in minimum distance robust or partially robust codes that can be used for error corrections. Compared to traditional linear error correcting codes, the advantage of these codes is that they can provide better protection against multi-bit errors and repeating errors due to the fact that they have less undetectable and miscorrected errors.

4.2 Optimality of Systematic Minimum Distance Robust and Partially Robust Codes

In this section, we present an exact upper bound for the size of the systematic minimum distance robust and partially robust codes in terms of Q_{mc}, n, k, d and the dimension of the detection kernel ω_d.

The redundant bits of a systematic code are generated by an encoding function $f : GF(2^k) \rightarrow GF(2^r)$, $r = n - k$. In order to simplify the analysis, we distinguish between the encoding functions for linear and nonlinear redundant bits. Denote by r_L and r_N the number of linear and nonlinear redundant bits respectively. $r = r_L + r_N$. We represent a codeword c of a systematic code in the following format.

$$c = \big(x, f_L(x), f_N(x)\big), \tag{5}$$

where $f_L : GF(2^k) \rightarrow GF(2^{r_L})$ is the encoding function for linear redundant bits which can be implemented using only *XOR* gates, $f_N : GF(2^k) \rightarrow GF(2^{r_N})$ is the encoding function for nonlinear redundant bits which cannot be implemented using only *XOR* gates.

Theorem 1 *Denote by $r(d, n)$ the smallest possible number of redundant bits for a systematic code with minimum Hamming distance d and length n. For any (n, k, d) code C,*

$$2^k \leq Q_{mc}\big(2^n - 2^k 2^{r(d,n)} + \big(2^k - 2^{\omega_d}\big)2^{r_N}\big) + 2^{\omega_d}, \tag{6}$$

where ω_d is the dimension of the detection kernel of C and $Q_{mc} = \max_{\{e|e \notin K_d\}} Q(e)$.

Proof Let $e = (e_1, e_2, e_3)$ be the error vector, where $e_1 \in GF(2^k), e_2 \in GF(2^{r_L}), e_3 \in GF(2^{r_N})$. We divide the errors into two classes as stated below.

1. $e_2 \neq f_L(e_1)$. These errors will be detected by the linear redundant bits of the code and are never masked. The number of errors in this class is $2^k(2^{r_L - 1})2^{r_N}$.

2. $e_2 = f_L(e_1)$. In this case an error $e = (e_1, e_2, e_3)$ is masked by a codeword $c = (x_1, x_2, x_3)$ iff there exists another codeword $c' = (x'_1, x'_2, x'_3)$ such that

$$x_1 \oplus x'_2 = e_1;$$
$$f_N(x_1) \oplus f_N\left(x'_2\right) = e_3.$$

Equivalently, $f_N(x_1 \oplus e_1) \oplus f_N(x_1) = e_3$. Errors in this class can be further divided into two classes.

(a) If $e \in K_d$, it will be masked by all codewords of the code. The number of errors in this class is 2^{ω_d}.
(b) If we can find an error $e' = (e'_1, e'_2, e'_3)$ in K_d such that $e_1 = e'_1, e_2 = e'_2, e_3 \neq e'_3$, e will always be detected. The number of errors in this class is $2^{\omega_d}(2^{r_N} - 1)$.
(c) All the other errors will be masked by no more than $2^k Q_{mc}$ codewords.

According to the above analysis, we have

$$\begin{aligned}2^k &\leq Q_{mc}\left(2^n - 2^k\left(2^{r_L} - 1\right)2^{r_N} - 2^{\omega_d}2^{r_N}\right) + 2^{\omega_d}\\ &\leq Q_{mc}\left(2^n - 2^k 2^{r(d,n)} + \left(2^k - 2^{\omega_d}\right)2^{r_N}\right) + 2^{\omega_d}.\end{aligned}$$

□

The function $r(d, n)$ can be estimated using existing bounds for error control codes that are extensively studied in the community. For example, the Hamming bound and the Singleton bound [26]. When $r(d, n)$ is derived from the Hamming bound, Eq. 6 is equivalent to

$$2^k \leq Q_{mc}\left(2^n - 2^k \sum_{i=0}^{\lfloor \frac{d-1}{2} \rfloor} \binom{n}{i} + \left(2^k - 2^{\omega_d}\right)2^{r_N}\right) + 2^{\omega_d}. \quad (7)$$

Definition 5 A systematic minimum distance robust or partially robust code (n, k, d) satisfying the equality in Eq. 6 is *perfect.*

For the design of systematic minimum distance robust and partially robust codes, the best codes should have the maximum k given all the other parameters. When perfect codes are not available, the codes with maximum possible k when other parameters are fixed are called *optimum systematic minimum distance robust (partially robust) codes*.

Definition 6 A (n, k, d) code which has the maximum possible k for given n, Q_{mc}, ω_d and d with respect to Eq. 6 is called *optimum*. For optimum codes increasing k will violate bound Eq. 6.

Remark 1 The optimality of minimum distance robust and partially robust codes that achieve the equality in bound Eq. 6 are twofold. First, it is perfect in a sense that it has the minimum number of redundant bits among all codes with distance d and length n. Second, it is perfect in a sense that it achieves the highest possible robustness with a given number of nonlinear redundant bits r_N. To be perfect in terms of bound Eq. 6, the following two conditions must be satisfied.

1. The total number of redundant bits $r = r_L + r_N = r(d, n)$;
2. $f_N(x)$ is a *perfect nonlinear function*.

The nonlinearity of a function $f_N : GF(2^k) \rightarrow GF(2^{r_N})$ can be measured using derivatives $D_a f_N(x) = f_N(x \oplus a) \oplus f_N(x)$. The nonlinearity can be defined by [4]

$$P_{f_N} = \max_{0 \neq a \in GF(2^k)} \max_{b \in GF(2^{r_N})} Pr\left(D_a f_N(x) = b\right), \quad (8)$$

where $Pr(E)$ denotes the probability of occurrence of event E. The smaller the value of P_{f_N}, the higher the corresponding nonlinearity of f_N. f_N is a perfect nonlinear function when $P_{f_N} = 2^{-r_N}$.

Example 4

1. The nonlinear function $f_N : GF(2^{2st}) \rightarrow GF(2^t)$ defined by $f_N(x) = x_1 \bullet x_2 \oplus x_3 \bullet x_4 \oplus \cdots \oplus x_{2s-1} \bullet x_{2s}$, where $x_i \in GF(2^t)$, $1 \leq i \leq 2s$ and $\bullet$ is the multiplication in $GF(2^t)$ is a perfect nonlinear function with $P_{f_N} = 2^{-t}$ [4].
2. Let $f_N(x) : GF(2^n) \rightarrow GF(2^n)$ be a nonlinear function defined by $f_N(x) = x^3$, where x^3 is the cubing operation in $GF(2^n)$. For every $a, b \in GF(2^n)$, $a \neq 0$, there are at most two x satisfying $D_a f_N(x) = b$. Thereby $P_{f_N} = 2^{-n+1}$. f_N is not a perfect nonlinear function because $P_{f_N} > 2^{-r_N}$. However, it has the smallest possible P_{f_N} among functions mapping $GF(2^n)$ to itself and is called *almost perfect nonlinear* (APN) [4].
3. For and (n, k, d) linear systematic code C, let x be the information bits of the codeword and $H = [P|I]$ be the $r \times n$ parity check matrix in standard form, where I is the $r \times r$ identity matrix. The encoding function $f_L(x) = Px$ is linear and has $P_{f_L} = 1$.

More constructions of perfect and almost perfect nonlinear functions can be found in [4].

The characteristics and error detection capabilities of systematic robust and partially robust codes are strongly related to the corresponding nonlinear encoding functions. For a nonlinear robust code C composed of all vectors $\{(x, f(x))\}$, where $x \in GF(2^k)$ is the information part and $f : GF(2^k) \to GF(2^r)$ is the nonlinear encoding function with nonlinearity P_f, an error $e = (e_x, e_y)$, $e_x \in GF(2^k)$, $e_y \in GF(2^r)$ is masked iff $f(x \oplus e_x) = f(x) \oplus e_y$. If $e_x = 0$ and $e_y \neq 0$, $f(x) \neq f(x) \oplus e_y$. If $e_x \neq 0$, according to the definition of P_f, $P_r(f(x \oplus e_x) = f(x) \oplus e_y) \leq P_f$. Thereby, every nonzero error is masked by C with a probability of at most P_f.

In the next section we will present several constructions of optimum or perfect minimum distance robust and partially robust codes. The optimality of these codes are summarized in Table 2.

5 Constructions of Codes

5.1 Minimum Distance Robust Codes

The simplest way to construct minimum distance robust codes is to append extra nonlinear redundant bits to codewords of an existing code with given distance d.

Theorem 2 *Let C be a (n, k, d) linear systematic code and let $f : GF(2^k) \to GF(2^r)$ be a nonlinear function with nonlinearity P_f. The code*

$$C' = \{(x, \phi(x), f(x)) | (x, \phi(x)) \in C\}, \tag{9}$$

where ϕ is the encoding function for code C, is a $(n + r, k, d)$ minimum distance robust code with $Q_{mc} = P_f$ and $\omega_d = 0$.

Proof Appending extra nonlinear bits does not change the minimum distance of the code. Any error which will affect only the redundant bits of C will clearly be immediately detected. Any other error will be detected by the robust code. Q_{mc} of the code follows from the definition of P_f. □

Example 5 (Shortened Robust Hamming) Let $C = \{(x, Px)\}$ be a $(38, 32, 3)$ shortened Hamming code, where $x \in GF(2^{32})$, P is a 6×32 encoding matrix and $Px \in GF(2^6)$. Let $f : GF(2^{32}) \to GF(2)$ be a perfect nonlinear function defined by $f(x = (x_1, x_2, ..., x_{32})) = x_1 \cdot x_2 \oplus x_3 \cdot x_4 \oplus ... \oplus x_{31} \cdot x_{32}$ [21]. Then the code $C' = \{(x, Px, f(x))\}$ is a robust code with minimum distance three. For this code, $\omega_d = 0$, $Q(e) = 0$ when $||e|| < 3$ and $Q(e) \leq 0.5$ when $||e|| \geq 3$. It is easy to verify that C' is not perfect because the equality in Eq. 6 is not satisfied. However, C' is optimal since increasing k will violate Eq. 6, assuming that other parameters are unchanged.

5.2 Minimum Distance Partially Robust Codes

Many known constructions of nonlinear codes are minimum distance partially robust codes. They have a minimum distance larger than one and have much fewer undetectable errors than linear codes. Such codes can even be perfect with respect to the classical Hamming bound [26].

5.2.1 Vasil'ev Codes and Their Generalizations

The first perfect nonlinear Hamming code was constructed by Vasil'ev in [39] and was generalized by Mollard in [29]. We first review the basic construction of the Vasil'ev code.

Theorem 3 (Vasil'ev Code [39]) *For $u \in GF(2^m)$, let $p(u)$ be the linear parity check of u. Let V be a perfect not necessarily linear Hamming code of length $m = 2^r - 1$ with $k_V = m - r$ information bits. Let $f : V \to \{0, 1\}$ be an arbitrary nonlinear mapping such that $f(\mathbf{0}) = 0$, $\mathbf{0} \in GF(2^m)$ and $f(v) \oplus f(v') \neq f(v \oplus v')$ for some $v, v' \in V$. The code C defined by*

$$C = \{(u, u \oplus v, p(u) \oplus f(v)) | u \in GF(2^m), v \in V\} \tag{10}$$

is a perfect nonlinear Hamming code.

Corollary 1 *Vasil'ev code is a $(2m + 1, 2m - r, 3)$ partially robust code with $\omega_d = m$ and $Q_{mc} = P_f$ where P_f is the nonlinearity of f. The code is perfect with respect to bound Eq. 6.*

Proof Let H be the check matrix of V. An error $e = (e_1, e_2, e_3)$ where $e_1, e_2 \in GF(2^m)$ and $e_3 \in GF(2)$ is masked if and only if $H(e_1 \oplus e_2) = \mathbf{0}$ and $f(v \oplus e_1 \oplus e_2) \oplus f(v) \oplus p(e_1) \oplus e_3 = 0$. The errors can be divided into four classes as follows.

1. $e_1 = e_2$ and $p(e_1) = e_3$, the error will always be masked. The number of errors in this class is 2^m;
2. $e_1 = e_2$ but $p(e_1) \neq e_3$, the error will always be detected. There are 2^m errors belonging to this class;

3. $H(e_1 \oplus e_2) = \mathbf{0}$ but $e_1 \neq e_2$, the error masking probability depends on the nonlinear function f; In the worst case, a specific error will be masked by $P_f \times |C|$ codewords. The number of errors in this class is $2^{m+1}(2^{m-r} - 1)$;
4. $H(e_1 \oplus e_2) \neq \mathbf{0}$. The error in this class will always be detected. The number of errors is $2^{m+1}(2^m - 2^{(m-r)})$. □

Vasil'ev codes are perfect single error correcting codes and have the same parameters as linear Hamming codes. The basic construction of Vasil'ev code can be further generalized as follows. The following theorem can be proved in a similar way to the proof for Theorem 3 and Corollary 1.

Theorem 4 (Generalized Vasil'ev Code) *For $u \in GF(2^a)$, let $p(u)$ be the linear parity check of u. Let V be a $(m, k_V, 3)$ not necessarily linear Hamming code with $r = m - k_V$ redundant bits. without loss of generality, assume that the first k_V bits in any codeword of V are information bits. Denote by $v = (y, z)$, $y \in GF(2^{k_V})$, $z \in GF(2^r)$ the codewords of V. Select $f : GF(2^{k_V}) \rightarrow GF(2)$ to be an arbitrary mapping such that $f(\mathbf{0}) = 0$ and $f(y) \oplus f(y') \neq f(y \oplus y')$ for some $y, y' \in GF(2^{k_V})$. The code C defined by*

$$C = \big\{(u, (u, \mathbf{0}) \oplus v, p(u) \oplus f(y))\big\}, \tag{11}$$

*where $u \in GF(2^a)$, $\mathbf{0} \in GF(2^{m-a})$, $0 < a \leq m$, $v \in V$ is a $(a + m + 1, a + k_V, 3)$ code with $\omega_d = a$ and $Q_{mc} = P_f$. C is optimum with respect to bound Eq. 6. Adding one more overall linear parity bit to C will result in a nonlinear SEC-DED code with the same ω_d and Q_{mc} as C and minimum distance four, which we call the **extended Vasil'ev code**.*

The significance of Theorem 4 is twofold. First, it can generate robust SEC-DED codes of arbitrary lengths. These codes have the same number of redundant bits as the best linear SEC-DED codes in terms of the number of redundant bits but much smaller number of undetectable multi-bit errors and are more suitable for applications where the MBU rate is high. Second, it allows a tradeoff between robustness and hardware overhead. Generally speaking, smaller a results in increased robustness of the code but requires more hardware for the encoder. By carefully selecting a and m, we can construct codes for situations that have different requirements for robustness and the hardware overhead.

Example 6

1. Let $a = 16$ and V be a $(21, 16, 3)$ shortened Hamming code. Select f to be the same nonrepetitive quadratic function as in Example 5. The extended Vasil'ev code constructed by adding one more overall parity bit to the generalized Vasil'ev construction described in Theorem 4 is a $(39, 32, 4)$ partially robust code with $Q_{mc} = 0.5$ and $\omega_d = 16$.
2. Alternatively let $a = 6$ and V be a $(31, 26, 3)$ perfect Hamming code. In this case we can construct a $(39, 32, 4)$ partially robust code with $Q_{mc} = 0.5$ and $\omega_d = 6$ at the price of larger hardware overhead for the encoder.
3. For applications where hardware overhead is more critical, we can select $a = 18$ and V to be a $(19, 14, 3)$ shortened Hamming code. The resulting partially robust code will have $\omega_d = 18$, which is the biggest of the 3 discussed variants. However, the hardware overhead for the encoder of this implementation will be the smallest.

5.2.2 Phelps Codes and Their Generalizations

We next review the basic construction of *Phelps Codes* that was proposed in [32], analyze its detection kernel and conditional error masking properties and generalize the construction.

Theorem 5 (Phelps Code [32]) *Let C, B be two perfect linear Hamming codes of length $n = 2^m - 1$. Let $\{C = C_0, C_1, C_2, \cdots, C_n\}$, $\{B = B_0, B_1, B_2, \cdots, B_n\}$ be two partitions of $GF(2^n)$ with $|C| = |C_i| = |B| = |B_i| = 2^{n-m}$ such that the minimum distance between any two vectors in the same coset C_i (B_i) is three. Let α be any permutation of $GF(2^m)$ such that $\alpha(\mathbf{0}) = \mathbf{0}$. Represent each coset C_i (B_i) by a m-bit vector which is called the **coset vector** of C_i (B_i). Denote by $[x] \in GF(2^m)$ the coset vector of the coset that x belongs to. The coset vector can always be selected in such a way that $[x] \oplus [y] = [x \oplus y]$, $x, y \in GF(2^n)$. Define the **extended Phelps code** C' as follows:*

$$\big(c, p(c), b, p(b)\big) \in C' \text{ if and only if } \alpha([c]) = [b],$$

*where $p : GF(2^n) \rightarrow GF(2)$ is the linear parity function. Deleting any coordinate of C' will result in a perfect nonlinear Hamming code with $d = 3$ and length $2^{m+1} - 1$ which is called **Phelps code**.*

Corollary 2 *Let P_α be the nonlinearity of α. Phelps code is a*

$$\big(2^{m+1} - 1, 2^{m+1} - m - 2, 3\big)$$

perfect nonlinear Hamming code with $Q_{mc} = P_\alpha$ and $\omega_d = 2^{m+1} - 2m - 2$, which is optimum with respect to bound Eq. 6. The extended Phelps code has $d = 4$ and the same Q_{mc} and ω_d as the Phelps code.

Proof We first analyze the error masking properties for the extended Phelps code C' constructed in Theorem 5. The codewords of C' can be written in the following format:

$$c = \big(x_1, x_2, x_3, x_4\big),$$

where $x_1, x_3 \in GF(2^n)$, $x_2 = p(x_1) \in GF(2)$, $x_4 = p(x_3) \in GF(2)$ and $[x_3] = \alpha([x_1])$. Denote by $e = (e_1, e_2, e_3, e_4)$ the error vector, $e_1 \in GF(2^n), e_2 \in GF(2), e_3 \in GF(2^n)$, $e_4 \in GF(2)$. If $e_2 \neq p(e_1)$ or $e_4 \neq p(e_3)$, the error will never be masked. If $e_2 = p(e_1)$ and $e_4 = p(e_3)$, the error e will be masked by a codeword $c = (x_1, x_2, x_3, x_4)$ if and only if

$$[x_3] = \alpha([x_1]);$$
$$[x_3 \oplus e_3] = \alpha\big([x_1 \oplus e_1]\big).$$

The errors can be further divided into the following classes.

1. $e_1 \in C, e_3 \in B$. Then x_1 and $x_1 \oplus e_1$ belong to the same coset. x_3 and $x_3 \oplus e_3$ belong to the same coset. In this case the above two equations are always satisfied. Errors in this class form the detection kernel of C' and will be masked by all codewords. The number of these errors is $2^{2^{m+1}-2m-2}$.
2. $e_1 \in C, e_3 \notin B$. In this case x_1 and $x_1 \oplus e_1$ belong to the same coset. x_3 and $x_3 \oplus e_3$ belong to different cosets. Errors in this class will never be masked.
3. $e_1 \notin C, e_3 \in B$. Similar to the last case, errors in this class will never be masked.
4. $e_1 \notin C$, $e_3 \notin B$. The error masking equations are equivalent to:

$$\alpha\big([x_1] \oplus [e_1]\big) \oplus \alpha[x_1] = [e_3]. \quad (12)$$

   α is a nonlinear function from $GF(2^m)$ to $GF(2^m)$ with nonlinearity P_α. The number of $[x_1]$ satisfying the above equation is no more than $P_\alpha \times 2^m$. There are at most $P_\alpha \times 2^{2n-m}$ codewords that mask the error $e = (e_1, e_2, e_3, e_4)$. Thereby the error will be masked by a probability no larger than P_α.

Without loss of generality, we assume that the parity check bit x_4 is deleted for all codewords. The resulting code is a perfect nonlinear Hamming code with minimum distance 3. It is easy to show that this code has the same Q_{mc} and ω_d as C' and is optimum with respect to bound Eq. 6. $\square$

Remark 2

1. Codes C and B do not have to be linear Hamming codes. Generally speaking, ω_d of the Phelps code is equal to the sum of the dimension of K_d for C and B.
2. To optimize Q_{mc}, P_α should be as small as possible. The best nonlinear function with the smallest P_α from $GF(2^m)$ to $GF(2^m)$ has $P_\alpha = 2^{-m+1}$ and is called almost perfect nonlinear function [4]. One example is x^3 [28], which is based on the cube operation in $GF(2^m)$.

Example 7 Let $C = B$ be a $(7, 4, 3)$ perfect linear Hamming code defined by the parity check matrix

$$H = \begin{pmatrix} 1001011 \\ 0101110 \\ 0010111 \end{pmatrix}.$$

$GF(2^7)$ can be partitioned into eight cosets of the perfect Hamming code as it is shown in Table 1. The coset vectors $[x] \in GF(2^3)$ are assigned in such a way that $[x] \oplus [y] = [x \oplus y], x, y \in GF(2^7)$. For example, suppose $x = 0000001$ and $y = 0010000$. We have $[x] = 001, [y] = 111$ and $[x \oplus y] = [0010001]$. Since $H(x \oplus y) = 100 = h_1$, where h_1 is the first column of H, $x \oplus y \in C_7$ and $[x \oplus y] = 110 = [x] \oplus [y]$.

Let $\alpha([x]) = [x]^3$ with $P_\alpha = 2^{-r+1} = \frac{1}{4}$ (Example 4). According to Corollary 2, the resulting Phelps code has $\omega_d = 2^{3+1} - 2 \times 3 - 2 = 8$ and $Q_{mc} = P_\alpha = \frac{1}{4}$.

Theorem 5 can be further generalized to generate partially robust codes with Hamming distance three and four for any given length.

Table 1 Cosets of the (7, 4, 3) linear Hamming code

	Coset leader x	Coset vector $[x]$
$C_0 = C$ $(B_0 = B)$	0000000	000
C_1 (B_1)	0000001	001
C_2 (B_2)	0000010	010
C_3 (B_3)	0000100	100
C_4 (B_4)	0001000	101
C_5 (B_5)	0010000	111
C_6 (B_6)	0100000	011
C_7 (B_7)	1000000	110

Theorem 6 (Generalized Phelps Code) *Let C be a $(n_1, k_1, 3)$ linear code and B be a $(n_2, k_2, 3)$ linear code. Without loss of generality, assume that $r_2 = n_2 - k_2 \geq r_1 = n_1 - k_1$. Let $\{C_0 = C, C_1, \cdots, C_{2^{r_1}-1}\}$ be a partition of $GF(2^{n_1})$ and $\{B_0 = B, B_1, \cdots, B_{2^{r_2}-1}\}$ be a partition of $GF(2^{n_2})$, where $|C_i| = 2^{k_1}, 0 \leq i \leq 2^{r_1} - 1$ and $|B_i| = 2^{k_2}, 0 \leq i \leq 2^{r_2} - 1$. Represent each coset C_i by a r_1-bit coset vector and each coset B_i by a r_2-bit coset vector. The coset vector can always be selected in such a way that $[x] \oplus [y] = [x \oplus y]$, where x, y are vectors in the same field and $[x]$ is the coset vector of the coset that x belongs to. Let α be any permutation of $GF(2^{r_1})$ such that $\alpha(\mathbf{0}) = \mathbf{0}$. Define C' as*

$$(c, p(c), b, p(b)) \in C' \text{ if and only if } (\mathbf{0}, \alpha([c])) = [b],$$

where $c \in GF(2^{n_1})$, $b \in GF(2^{n_2})$ and p is the linear parity function. C' is a $(n_1 + n_2 + 2, n_1 + k_2, 4)$ partially robust code with $Q_{mc} = P_\alpha$ and $\omega_d = k_1 + k_2$. Deleting any coordinate from every codeword of C' will result in a partially robust code with minimum distance three and the same value of Q_{mc} and ω_d as C'.

Theorem 6 can be proved in a similar way to Theorem 5 and Corollary 2.

Example 8 Let C be a $(15, 10, 3)$ linear code whose parity check matrix is:

$$H = \begin{pmatrix} 100001001011001 \\ 010000100101100 \\ 001001011001111 \\ 000100101100111 \\ 000010010110011 \end{pmatrix}.$$

Let B be a $(22, 17, 3)$ shortened Hamming code whose parity check matrix is:

$$H = \begin{pmatrix} 1000010010110011111000 \\ 0100001001011001111100 \\ 0010010110011111000110 \\ 0001001011001111100011 \\ 0000100101100111110001 \end{pmatrix}.$$

Let $\alpha([x]) = [x]^3$ with $P_\alpha = \frac{1}{16}$, where $[x] \in GF(2^5)$. Construct C' as described in Theorem 6. Then C' is a $(39, 32, 4)$ partially robust code with $\omega_d = 27$ and $Q_{mc} = P_\alpha = \frac{1}{16}$.

5.2.3 One Switching Constructions and Their Generalizations

Vasil'ev codes and Phelps codes usually have $\omega_d > 1$. Another important construction is *the switching construction*, which can generate nonlinear codes with $\omega_d = 1$. We first review the basic switching constructions presented by T. Etzion and A. Vardy in [10].

Theorem 7 (The Switching Code [10]) *Let C be a perfect linear Hamming code of length $n = 2^r - 1$. Refer to the codewords of Hamming weight three as* triples. *Denote by T_i the subspace spanned by the triples that have one at the i_{th} bit. Let $\{C \oplus e_i^*\}$ be a translate of C, where $e_i^* \in GF(2^n)$ and has one at position i and zero elsewhere. Assume $\{T_i \oplus z\} \in C$ for some $z \in C$. A* ***switch*** *is a process of replacing the coset $\{T_i \oplus z\}$ with the coset $\{T_i \oplus z \oplus e_i^*\}$. The resulting one switching code C' defined by*

$$C' = \{C \backslash (T_i \oplus z)\} \bigcup \{T_i \oplus z \oplus e_i^*\} \tag{13}$$

for some $i \in \{1, 2, \cdots, n\}$ is a perfect nonlinear Hamming code of length $n = 2^r - 1$.

Corollary 3 *The one switching code C' is a partially robust code with $\omega_d = 2^{r-1} - 1$. In addition, $2^{2^r-1-r} - 2^{2^{r-1}-1}$ errors are masked with probability $1 - 2^{-2^{r-1}+1+r}$ and $2^{2^r-1-r} - 2^{2^{r-1}-1}$ errors are masked with probability $2^{-2^{r-1}+1+r}$. The code is optimum with respect to bound Eq. 6.*

Proof Without loss of generality, we assume that $z = 0$. C' is constructed by replacing T_i with the coset $\{T_i \oplus e_i^*\}$. The errors can be divided into four classes as stated below.

1. $e \in T_i$. Since T_i is a linear subspace, we have

   $$c \oplus e \in \{T_i \oplus e_i^*\} \text{ iff } c \in \{T_i \oplus e_i^*\};$$
   $$c \oplus e \in \{C \backslash T_i\} \text{ iff } c \in \{C \backslash T_i\}.$$

   Hence $c \oplus e \in C'$ for every $c \in C'$. Errors in this class form the detection kernel of the code and will always be masked. The dimension of T_i is $\frac{n-1}{2}$ [33]. Thereby the number of errors in this class is $2^{\frac{n-1}{2}} = 2^{2^{r-1}-1}$.
2. $e \in \{C \backslash T_i\}$. If $c \in \{T_i \oplus e\}$, $c \oplus e \in T_i$ and the error will be detected. If $c \in \{T_i \oplus e_i^*\}$, $c \oplus e \in \{T_i \oplus e_i^* \oplus e\}$, again the error will be detected. All the other codewords $c \in C', c \notin \{T_i \oplus e\}, c \notin \{T_i \oplus e_i^*\}$ will mask the error. Thereby errors in this class will be masked by a probability $\frac{|C|-2|T_i|}{|C|} = 1 - 2^{-2^{r-1}+1+r}$. The number of errors in this class is $|C| - |T_i|$.

3. $e \in \{\{C \backslash T_i\} \oplus e_i^*\}$. If $c \in \{T_i \oplus e_i^*\}$, $c \oplus e \in \{C \backslash T_i\}$ and the error will be masked. If $c \in \{T_i \oplus e \oplus e_i^*\}$, $c \oplus e \in \{T_i \oplus e_i^*\}$. The error will be masked. For all the other codewords the error will be detected. Thereby errors in this class will be masked by a probability $\frac{2|T_i|}{|C|} = 2^{-2^{r-1}+1+r}$. The number of errors in this class is $|C| - |T_i|$.
4. Errors that do not belong to the above three cases will always be detected. □

Example 9 Let C be a $(15, 11, 3)$ perfect linear Hamming code with the following parity check matrix

$$H = \begin{pmatrix} 100010011010111 \\ 010011010111100 \\ 001001101011110 \\ 000100110101111 \end{pmatrix}.$$

There are $\frac{n-1}{2} = 7$ codewords of Hamming weight three with the first bit equal to one, which are

100100000000001,
100000000001100,
101000001000000,
100000010100000, .
100000100000010,
100001000010000,
110010000000000.

T_1 is the subspace spanned by the above seven codewords. The dimension of T_1 is seven. Construct C' by replacing T_1 with $\{T_1 \oplus e_1^*\}$, where e_1^* has one at the first bit and zero elsewhere. C' is a prefect nonlinear Hamming code with $\omega_d = 7$. There are $2^{11} - 2^7$ errors that are masked by probability $\frac{7}{8}$ and $2^{11} - 2^7$ errors that are masked by probability $\frac{1}{8}$.

Another generalization of Theorem 7 was shown in [33], which indicated that perfect nonlinear Hamming codes with $\omega_d = 1$ could be constructed by switching linear Hamming codes for multiple times.

Theorem 8 [33] *Let C be a linear Hamming code of length $n = 2^r - 1, r \geq 4$, there exists $z_i, 1 \leq i \leq r$ such that the code*

$$C' = \left\{ C \backslash \left(\bigcup_{i=1}^{k} T_i \oplus z_i \right) \right\} \bigcup \left\{ \bigcup_{i=1}^{k} T_i \oplus z_i \oplus e_i^* \right\} \quad (14)$$

is a perfect nonlinear code with $\omega_d = 1$.

To end the section we summarize the optimality of different robust and partially robust codes with respect to bound Eq. 6 in Table 2 for the case when $r(d, n)$ is derived from the Hamming bound. The best candidates for protecting memories in channels with high MBU rate or laziness are generalized Vasil'ev codes and generalized Phelps codes. One switching code is better than linear Hamming code but worse than the above two candidates due to its larger Q_{mc}. We note that the multiple switching construction (Theorem 8)

Table 2 Optimality of robust and partially robust codes with respect to bound Eq. 6

	n	k	ω_d	Q_{mc}	d	r_N	Perfect	Optimum
Vasil'ev codes (Theorem 3)	$2^r - 1$	$2^r - 1 - r$	$2^{r-1} - 1$	0.5	3	1	√	–
Generalized Vasil'ev code (Theorem 4)	$a + m + 1$	$a + k_V$	a	0.5	3	1	–	√
Phelps code (Theorem 5)	$2^r - 1$	$2^r - 1 - r$	$2^r - 2r$	P_α	3	$r - 1$	–	√
Generalized extended Phelps code (Example 8)	39	32	28	$\frac{1}{4}$	4	6	–	–
One switching code (Theorem 7)	$2^r - 1$	$2^r - 1 - r$	$2^{r-1} - 1$	$1 - 2^{-2^{r-1}+1+r}$	3	1	–	√
$(x, (Px)^3)$ [17]	$k + r$	k	$k - r$	2^{-r+1}	1, 2[a]	r	–	–
Quadratic systematic code[b] [20]	$(2s + 1)r$	$2sr$	0	2^{-r}	1	r	–	√
Robust Hamming code[c] [22]	2^r	$2^r - 1 - r$	0	0.5	3	1	–	√

$r(d, n)$ is derived from the Hamming bound
[a]The distance of the code depends on r
[b]The codeword of the quadratic systematic code is in the format of $(x_1, x_2, \cdots x_{2s}, x_{2s+1})$, where $x_i \in GF(2^r)$, $1 \leq i \leq 2s + 1$ and $x_{2s+1} = x_1 \bullet x_2 \oplus x_3 \bullet x_4 \oplus \cdots \oplus x_{2s-1} \bullet x_{2s}$. $\bullet$ is the multiplication in $GF(2^r)$ [21]
[c]The codeword of the robust Hamming code is in the format of $(x, Px, f(x))$, where (x, Px) is the codeword of a $(2^r - 1, 2^r - 1 - r, 3$ perfect linear Hamming code and $f : GF(2^k) \rightarrow GF(2)$ is a nonlinear function. When f is a perfect nonlinear function, $Q_{mc} = 0.5$

can generate robust codes with $\omega_d = 1$. However, its high encoding and decoding complexities and high Q_{mc} make the code not a viable alternative for memory protection.

The constructions of minimum distance robust and partially robust codes presented in this section can be easily generalized for nonbinary case.

6 Architectures of Reliable Memory Systems Based on Nonlinear SEC-DED Codes

To demonstrate the advantage of utilizing nonlinear minimum distance partially robust codes to protect memories, we compare the error correction properties, the hardware overhead and the power consumption for the $(39, 32, 4)$ extended Vasil'ev code with $Q_{mc} = 0.5$ and $\omega_d = 6$ (Example 6), the $(39, 32, 4)$ extended Phelps code with $Q_{mc} = \frac{1}{16}$ and $\omega_d = 27$ (Example 8) and the linear $(39, 32, 4)$ extended Hamming code used in [38] to protect double data rate DIMM memory in a Virtex-II Pro device .

Figure 1 shows the general memory architecture with error correction function based on systematic error correcting codes. During a WRITE operation, the redundant bits of the code are generated by the encoder and saved in the redundant memory block. During a READ operation, the ECC block computes the syndrome of the retrieved data and executes the error correction

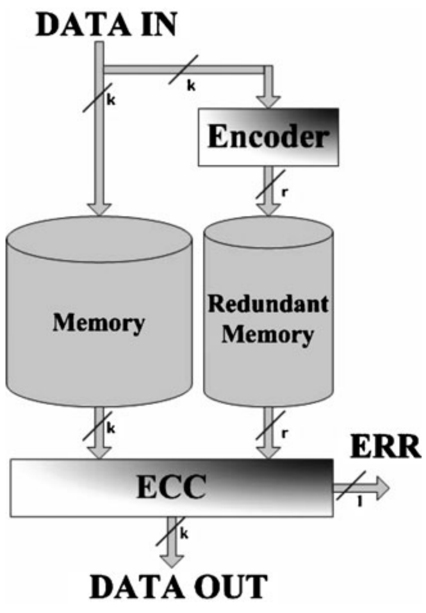

Fig. 1 General memory architecture with ECC

algorithm. If uncorrectable errors occur, ERR will be asserted and no correction will be attempted.

6.1 Memory Protection Architecture Based on the Extended Hamming Code

For linear SEC-DED codes, the encoder performs matrix multiplication over $GF(2)$ between the k-bit data and the encoding matrix P of the selected code. The parity check matrix used to generate the $(39, 32, 4)$ extended Hamming code C in [38] is in standard form $H = [P|I]$, where I is the 7×7 identity matrix and

$$P = \begin{pmatrix} 01010110101010101010110101011011 \\ 10011011001100110011011001101101 \\ 11100011110000111100011110001110 \\ 00000011111111000000011111110000 \\ 00000011111111001111110000000000 \\ 11111100000000000000000000000000 \\ 11111111111111111111111111111111 \end{pmatrix}.$$

The last redundant bit of the design in [38] is equal to the parity of the information bits. C is only able to detect double bit errors occurring in the information part of the code. If at least one bit of the double bit error is in the redundant portion of C, the code may miscorrect it as a single bit error. To make C a SEC-DED code, we compute the last parity bit based on all bits of the codeword.

The redundant bits are generated and written in the memory along with the associated 32-bit data. During the READ stage, the data and the redundant bits are read simultaneously. Syndromes $S = H\tilde{x}$, where $\tilde{x} \in GF(2^{39})$ is a possibly distorted output of the memory, are calculated and used to identify the error type and locate the error. A 32-bit correction mask is created to correct single bit errors occurring to the information part of the code. When a single bit error is detected, the original data is XORed with the mask and the distorted bit is reversed. When there are no errors or multi-bit errors, all the mask bits are zeros and the data go through the ECC block without any changes.

The disadvantage of memory protection architecture based on linear SEC-DED codes is the large number of undetectable and miscorrected multi-bit errors. For any linear systematic code, $K_d = C$ and $\omega_d = k$. Thereby the number of undetectable errors for a $(39, 32, 4)$ extended Hamming code is 2^{32}. All undetectable errors correspond to distortions of more than three bits.

It is easy to prove that any (n, k, d) linear systematic error correcting code C is able to correct up to $2^{n-k} - 1$

errors. If N errors are corrected, $0 \leq N \leq 2^{n-k} - 1$, the number of miscorrected errors is $N(2^k - 1)$.

For example, for the approach described in [38], only single errors occurring to the information part of the code will be corrected. Thereby $N = 32$. The number of miscorrected multi-bit errors for this code is $32(2^{32} - 1)$.

6.2 Memory Protection Architecture Based on the Extended Phelps Code

6.2.1 Error Correction Algorithm

Let C be a $(n_1, k_1, 3)$ linear code and B be a $(n_2, k_2, 3)$ linear code. Without loss of generality, assume that $r_1 = n_1 - k_1 < r_2 = n_2 - k_2$. Denote by H_C and H_B the parity check matrix for C and B respectively. Denote by $c' = (x_1, x_2, x_3, x_4)$ a codeword of C', where $x_1 \in GF(2^{n_1})$, $x_2 = p(x_1) \in GF(2)$, $x_3 \in GF(2^{n_2})$, $x_4 = p(x_3) \in GF(2)$. Let α be any permutation of $GF(2^{r_1})$ such that $\alpha(\mathbf{0}) = \mathbf{0}$. Denote by $[x_1]([x_3])$ the coset vector of the coset which $x_1(x_3)$ belongs to, $[x_1] \in GF(2^{r_1})$, $[x_3] \in GF(2^{r_2})$. The codewords of the extended Phelps code C' constructed as in Theorem 6 satisfy $[x_3] = (\mathbf{0}, \alpha([x_1]))$, where $\mathbf{0} \in GF(2^{r_2 - r_1})$. C' is a $(n_1 + n_2 + 2, n_1 + k_2, 4)$ SEC-DED code and is able to correct all single bit errors and simultaneously detect all double bit errors.

Denote by $e = (e_1, e_2, e_3, e_4)$ the error vector and $\tilde{c}' = (\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4)$ the distorted codeword, in which $\tilde{x}_i = x_i \oplus e_i$, $1 \leq i \leq 4$. The syndrome of the code that can be used to detect and locate errors are defined as $S = (S_1, S_2, S_3, S_4)$, where

$$S_1 = \tilde{x}_1, \tag{15}$$

$$S_2 = p(\tilde{x}_1) \oplus p(\tilde{x}_2) = p(e_1) \oplus e_2, \tag{16}$$

$$S_3 = \tilde{x}_3, \tag{17}$$

$$S_4 = p(\tilde{x}_3) \oplus p(\tilde{x}_4) = p(e_3) \oplus e_4. \tag{18}$$

The correction algorithm is as described below. For the purpose of comparing the error correction abilities of extended Phelps codes and linear extended Hamming codes presented before, in this algorithm only single bit errors occurring to the information part will be corrected.

1. Compute by Eq. 15 to Eq. 18 the syndrome of the code $S = (S_1, S_2, S_3, S_4)$, where $S_1 \in GF(2^{n_1})$, $S_3 \in GF(2^{n_2})$ and $S_2, S_4 \in GF(2)$.
2. If $S_2 = S_4 = 0$ and $[S_3] = (\mathbf{0}, \alpha([S_1]))$, no errors are detected.
3. If $S_2 = S_4 = 0$ and $[S_3] \neq (\mathbf{0}, \alpha([S_1]))$, multi-bit errors are detected and ERR will be asserted.
4. If $S_2 = 1$, $S_4 = 0$, multi-bit errors occur or a single bit error occurs to the first or the second part of the codeword.
   (a) If $[S_3] \geq 2^{r_1}$, multi-bit errors are detected and ERR will be asserted.
   (b) If $[S_3] < 2^{r_1}$, then
      i. If $[S_3] = (\mathbf{0}, \alpha([S_1]))$, the single bit error is in x_2.
      ii. If $[S_3] \neq (\mathbf{0}, \alpha([S_1]))$, multi-bit errors occur or a single bit error occurs to x_1. Let $\alpha^{-1} : GF(2^{r_1}) \to GF(2^{r_1})$ be the inverse function of α. Denote by $e_{[S_1]}$ and $e_{[\hat{S}_1]}$ the coset leaders of the cosets whose coset vectors are $[S_1]$ and $\alpha^{-1}([S_3]_{(r_1-1:0)})$, where $[S_3]_{(r_1-1:0)} \in GF(2^{r_1})$ is the rightmost r_1 bits of $[S_3]$. e_1 is the coset leader of the coset whose coset vector is $[e_{[S_1]} \oplus e_{[\hat{S}_1]}]$. If $||e_1|| > 1$, multi-bit errors are detected. If $||e_1|| = 1$, correct the single bit error by adding e_1 to x_1.
5. If $S_2 = 0$, $S_4 = 1$, multi-bit errors occur or a single bit error occurs to the third or the forth part of the codeword.
   (a) If $[S_3] = (\mathbf{0}, \alpha([S_1]))$, the single bit error is in x_4.
   (b) If $[S_3] \neq (\mathbf{0}, \alpha([S_1]))$, multi-bit errors occur or the single bit error is in x_3. Denote by $e_{[S_3]}$ and $e_{[\hat{S}_3]}$ the coset leaders of the cosets whose coset vectors are $[S_3]$ and $(\mathbf{0}, \alpha([S_1]))$. e_3 is the coset leader of the coset whose coset vector is $[e_{[S_3]} \oplus e_{[\hat{S}_3]}]$. Without loss of generality, assume the first k_2 bits of any codewords in B are the information bits. If $e_3 = e_i^*$, $1 \leq i \leq k_2$, where e_i^* has one at position i and zero elsewhere, correct the single bit error by adding e_3 to x_3. If $e_3 = e_i^*$, $k_2 < i \leq n_2$, single bit errors occur to the redundant bits. No corrections will be attempted. If $||e_3|| > 1$, multi-bit errors are detected.
6. If $S_2 = S_4 = 1$, multi-bit errors occur. ERR will be asserted and the data will go through ECC without any correction.

Example 10 In this example we show the error correction procedure for a $(11, 6, 4)$ extended Phelps code. Let C be a $(4, 1, 3)$ linear code whose parity check matrix is

$$H_C = \begin{pmatrix} 1001 \\ 1010 \\ 0100 \end{pmatrix}.$$

Table 3 Selected coset leaders and coset vectors

	Coset leader	Coset vector
$C_0 = C$	0000	000
C_1	0001	001
C_2	0010	010
C_3	1000	011
C_4	0100	100
C_5	1110	101
C_6	0110	110
C_7	1100	111
$B_0 = B$	00000	000
B_1	00001	001
B_2	00010	010
B_3	01000	011
B_4	00100	100
B_5	11000	101
B_6	10000	110
B_7	01100	111

Let B be a $(5, 2, 3)$ linear shortened Hamming code with

$$H_B = \begin{pmatrix} 01001 \\ 11010 \\ 10100 \end{pmatrix}.$$

Let $\{C_0 = C, C_1, \cdots, C_7\}, |C_i| = 2, 0 \leq i \leq 7$ be a partition of $GF(2^4)$ and $\{B_0 = B, B_1, \cdots, B_7\}, |B_i| = 4, 0 \leq i \leq 7$ be a partition of $GF(2^5)$. The coset leaders and coset vectors for C_i and B_i are selected as stated in Table 3. It is easy to verify that $[x \oplus y] = [x] \oplus [y]$ is satisfied.

Let $011001 \in GF(2^6)$ be the message that needs to be encoded. x_1 is the first four bits of the message: $x_1 = 0110 \in GF(2^4)$, which belongs to C_6 whose coset vector is 110. Let $\alpha([x_1]) = [x_1]^3$ where $x_1 \in GF(2^3)$. Select the primitive polynomial to be $[x_1]^3 + [x_1] + 1$ for $GF(2^3)$. Then $\alpha(110) = 111$. So $x_3 \in B_7$. x_3 is equal to the vector in B_7 with 01 for the information bits, which is 01100. $x_2 = p(x_1) = 0$, $x_4 = p(x_3) = 0$. So the entire codeword is $c = 01100011000 \in GF(2^{11})$. Suppose a single bit error occurs to the 6_{th} bit of the codeword, $\tilde{c} = 01100111000$. Then $S_1 = 0110$, $[S_1] = 110$, $S_2 = 0$, $S_3 = 11100$ and $S_4 = 1$. $H_B S_3 = H_B \times 00001 = 100$, so $S_3 = \tilde{x}_3$ belongs to B_1 and $[S_3] = 001$. Thereby $\alpha([S_1]) \neq [S_3]$ and the error is in x_3. $e_{[S_3]} \oplus e_{[\tilde{S}_3]} = 00001 \oplus 01100 = 01101$. So the error is the coset leader of the coset that 01101 belongs to. $H_B \times 01101 = H_B \times 10000$, hence $01101 \in B_6$ and $e_3 = 10000$.

Theorem 9 *Let C be a $(n_1, k_1, 3)$ linear code and B be a $(n_2, k_2, 3)$ linear code with $r_1 = r_2$, $2^{r_1-1} > max\{n_1, k_2\}$, where $r_1 = n_1 - k_1$ and $r_2 = n_2 - k_2$. Assume that α is an almost perfect nonlinear function with $P_\alpha = 2^{-r_1+1}$. The $(n_1 + n_2 + 2, n_1 + k_2, 4)$ extended generalized Phelps code C' constructed as in Theorem 6 has $\omega_d = k_1 + k_2$. The size of the correction kernel of the code is $(n_1 + k_2)(2^{k_1+k_2} - 1)$.*

Proof We divide the errors into four classes as stated below.

1. A nonzero error $e = (e_1, e_2, e_3, e_4)$ is masked if and only if it satisfies $S_2 = 0, S_4 = 0$ and $[S_3] = \alpha([S_1])$. $S_2 = 0$ and $S_4 = 0$ are satisfied if and only if $e_2 = p(e_1), e_4 = p(e_3)$. If $e_1 \in C$ and $e_3 \in B$, $[S_3] = \alpha([S_1])$ is always satisfied. These errors are undetectable and form the detection kernel of the code. The number of errors in this class is $2^{k_1+k_2}$. If $e_1 \notin C$ and $e_3 \notin B$, errors will be conditionally detectable. The number of these errors is $(2^{n_1} - 2^{k_1})(2^{n_2} - 2^{k_2})$. If $e_1 \in C, e_3 \notin B$ or $e_1 \notin C, e_3 \in B$, errors will always be detected.
2. If $S_2 = 0, S_4 = 0$ and $[S_3] \neq \alpha([S_1])$, multi-bit errors are detected.
3. $S_2 = 1, S_4 = 0, [S_3] = \alpha([S_1])$. We assume that a single bit error occurs to x_2. The error will be detected but not corrected.
4. $S_2 = 1, S_4 = 0, [S_3] \neq \alpha([S_1])$. We assume that a single bit error occurs to x_1. $S_2 = 1$ and $S_4 = 0$ are satisfied if and only if $e_2 = p(e_1) \oplus 1$ and $e_4 = p(e_3)$.
   (a) If $e_1 \in C, e_3 \notin B, [S_3] \neq \alpha([S_1])$ is always satisfied. In this case $[x_1 \oplus e_1] = [x_1]$. A multi-bit error e is miscorrected as a single bit error $e_i^*, 1 \leq i \leq n_1$ if and only if $\alpha([x_1] \oplus [e_i^*]) = \alpha([x_1]) \oplus [e_3]$. For every e_i^*, there are at most $2^{r_1} P_\alpha$ solutions for $[x_1]$. Since $P_\alpha = 2^{-r_1+1}$ and $2^{r_1-1} > n_1$, for all n_1 possible e_i^*, the total number of solutions for $[x_1]$ satisfying $\alpha([x_1] \oplus [e_i^*]) = \alpha([x_1]) \oplus [e_3]$ is less than 2^{r_1}. Thereby errors in this class are conditionally miscorrected.
   (b) If $e_1 \notin C, e_3 \in B, [S_3] \neq \alpha([S_1])$ is always satisfied. In this case $[x_3 \oplus e_3] = [x_3]$. A multi-bit error e is miscorrected as a single bit error $e_i^*, 1 \leq i \leq n_1$ if and only if $\alpha([x_1] \oplus [e_1] \oplus [e_i^*]) = \alpha([x_1])$. If $[e_1] = [e_i^*]$, errors will be corrected as e_i^* for all codewords. The number of errors in this class is $n_1 2^{k_1+k_2}$. n_1 of them are successfully corrected. The other $n_1(2^{k_1+k_2} - 1)$ errors belong to K_C. If $[e_1] \neq [e_i^*]$, the error will be conditionally miscorrected.
   (c) If $e_1 \notin C, e_3 \notin B$. A multi-bit error e is miscorrected as a single bit error $e_i^*, 1 \leq i \leq n_1$ if and only if $\alpha([x_1] \oplus [e_1] \oplus [e_i^*]) = \alpha([x_1]) \oplus [e_3]$. If $[e_1] = [e_i^*]$, errors will be always detected. If $[e_1] \neq [e_i^*]$, errors will be conditionally miscorrected.

5. $S_2 = 0, S_4 = 1, [S_3] = \alpha([S_1])$. We assume that a single bit error occurs to x_4. The error will be detected but not corrected.
6. $S_2 = 0, S_4 = 1, [S_3] \neq \alpha([S_1])$. We assume that a single bit error occurs to x_3.
   (a) If $e_1 \in C, e_3 \notin B$, $[S_3] \neq \alpha([S_1])$ is always satisfied. In this case a multi-bit error will be miscorrected as $e_i^*, 1 \le i \le k_2$ occurring to x_3 if and only if $\alpha([x_1]) = \alpha([x_1]) \oplus [e_3] \oplus [e_i^*]$. Errors will be corrected as $[e_i^*]$ by all codewords if $[e_3] = [e_i^*]$. The number of these errors is $k_2 2^{k_1+k_2}$. k_2 of them are successfully corrected. The other $k_2(2^{k_1+k_2} - 1)$ belong to K_C. If $[e_3] \neq [e_i^*]$, errors are always detected.
   (b) If $e_1 \notin C, e_3 \in B$, $[S_3] \neq \alpha([S_1])$ is always satisfied. In this case a multi-bit error will be miscorrected as $e_i^*, 1 \le i \le k_2$ occurring to x_3 if and only if $\alpha([x_1] \oplus [e_1]) = \alpha([x_1]) \oplus [e_i^*]$. Following the same analysis as for 4.(a), errors in this class will be conditionally miscorrected.
   (c) If $e_1 \notin C$, $e_3 \notin B$, a multi-bit error will be miscorrected as $e_i^*, 1 \le i \le k_2$ occurring to x_3 if and only if $\alpha([x_1] \oplus [e_1]) = \alpha([x_1]) \oplus [e_3] \oplus [e_i^*]$. These errors will be conditionally miscorrected.
7. $S_2 = 1, S_4 = 1$. In this case multi-bit errors occur and no error correction will be attempted. The number of errors in this class is $2^{n_1+n_2}$. □

From Theorem 9, it is easy to show that the (39, 32, 4) extended Phelps code constructed in Example 8 has $\omega_d = k_1 + k_2 = 5 + 22 = 27$. The size of the correction kernel is $(n_1 + k_2)(2^{k_1+k_2} - 1) = 32(2^{27} - 1)$.

6.2.2 *Hardware Implementation of the Encoder and the Decoder for the Extended Phelps Code*

The encoder for the extended Phelps code is mainly composed of the following parts:

1. Syndrome computation unit for C;
2. Circuits to realize the nonlinear permutation α. In our case α is the cube operation in $GF(2^{r_1})$.
3. Encoder for the linear Hamming code B;
4. Exclusive OR network to convert codewords of B to vectors in other cosets;
5. Parity check generation unit.

The input to the encoder can be any $n_1 + k_2$ bits binary vectors. The first n_1-bit is x_1 and the left k_2-bit is the information part of x_3. The syndrome computation unit computes $H_C x_1$, which is used to determine the coset that x_1 belongs to. $[x_3] = [x_1]^3$. x_3 is computed by first derive the codeword in B and then mask it with the coset leader of the coset which x_3 belongs to. The parity check generation unit computes the parity bits x_2 and x_4.

The architecture of the decoder for the extended Phelps code is shown in Fig. 2. After receiving a possibly distorted codeword $\tilde{c}' = (\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4)$, the

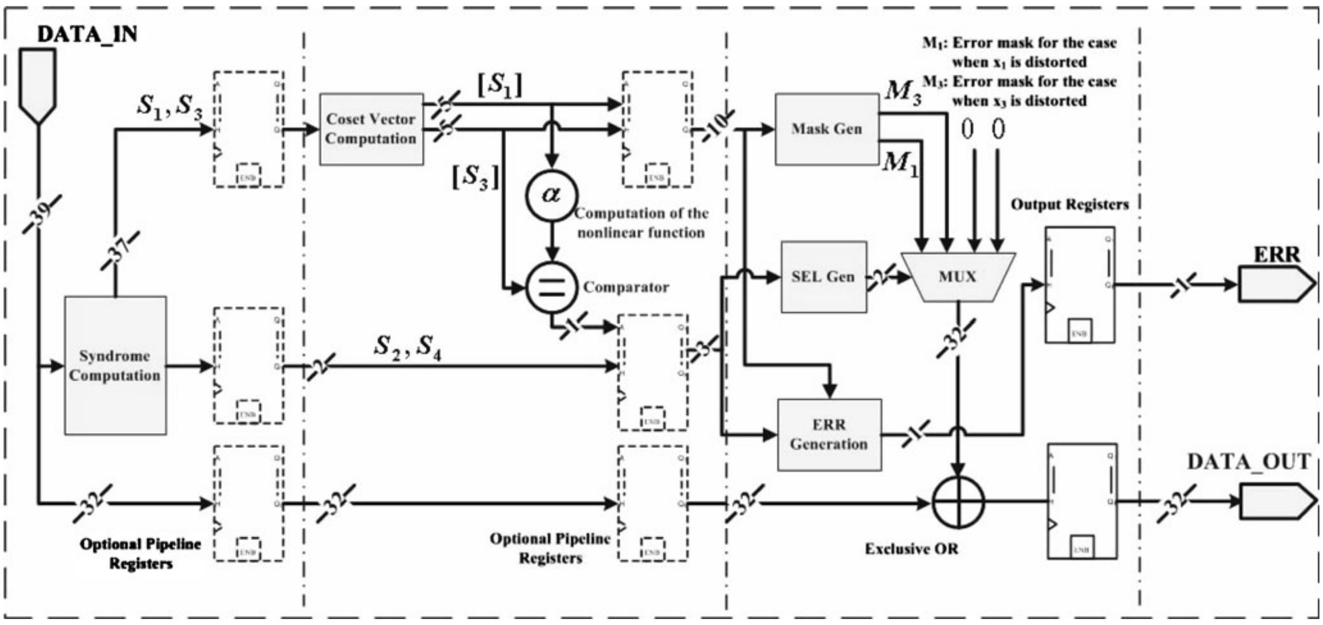

Fig. 2 The decoder architecture for the (39, 32, 4) extended Phelps code

syndrome $S = (S_1, S_2, S_3, S_4)$ will be computed. S_1 and S_3 are used to determine the coset vectors of the coset which $\tilde{x}_1$ and $\tilde{x}_3$ belong to. Then whether $[S_3] = [S_1]^3$ will be tested. To speed up the design, the possible error vector e_1 and e_3 will be pre-computed and sent to a MUX before the type of the error is known. If a single bit error in x_1 or the first k_2 bits of x_3 is detected, e_1 or e_3 will be XORed with the corresponding part of the received data to recover the original message. If multi-bit errors are detected, the ERR generation unit will pull up the ERR signal. The received data will go through the ECC module without any error correction.

The latency penalty, the hardware overhead and the power consumption of the encoder and the non-pipelined decoder for the extended Phelps code will be shown in Section 6.4.2. We also note that the design of the decoder for the extended Phelps code can be pipelined to increase the throughput of the system. The possible locations of the pipeline registers are shown in Fig. 2.

6.3 Memory Protection Architecture Based on the Extended Vasil'ev Code

6.3.1 Error Correction Algorithm

The codewords of a $(a + m + 2, a + k_V, 4)$ extended Vasil'ev code constructed as in Theorem 4 are in the format of

$$\big(u, (u, \mathbf{0}) \oplus v, p(u) \oplus f(y), p(u) \oplus p(v) \oplus f(y)\big),$$

where $u \in GF(2^a)$, $\mathbf{0} \in GF(2^{m-a})$, $0 < a \leq m$, $v \in V$ is the codeword of a $(m, k_V, 3)$ Hamming code V, $y \in GF(2^{k_V})$ are the information bits of v, $f : GF(2^{k_V}) \rightarrow \{0, 1\}$ is a nonlinear mapping satisfying $f(\mathbf{0}) = 0$ and p is the linear parity function. In order to simplify the encoding and decoding complexities, we select V to be a linear Hamming code.

The redundant portion of the extended Vasil'ev code contains three parts. The first part is the redundant bits of V which can be generated by a linear XOR network performing matrix multiplication over $GF(2)$. The second and the third part are nonlinear. The encoder for these two parts needs to perform the linear parity predictions $p(u)$, $p(v)$ as well as the nonlinear mapping $f : GF(2^{k_V}) \rightarrow \{0, 1\}$. When k_V is even, we can select f to be the non-repetitive quadratic function (Example 5) for the purpose of minimizing Q_{mc}.

$$f(v) = v_1 \cdot v_2 \oplus v_3 \cdot v_4 \oplus v_5 \cdot v_6 \oplus \cdots \oplus v_{k_V-3} \cdot v_{k_V-2} \oplus v_{k_V-1} \cdot v_{k_V}.$$

Before we describe the error correction algorithm for the extended Vasil'ev code, the syndrome S for locating and correcting errors need to be defined. Denote by $c = (x_1, x_2, x_3, x_4)$ the codeword of the extended Vasil'ev code, $e = (e_1, e_2, e_3, e_4)$ the error vectors and $\tilde{c} = (\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4)$ the distorted codeword.

$$x_1 = u,$$

$$x_2 = (u, \mathbf{0}) \oplus v,$$

$$x_3 = p(u) \oplus f(y),$$

$$x_4 = p(u) \oplus p(v) \oplus f(y).$$

Let H be the parity check matrix of the linear code V and $\tilde{y}$ be the distorted information bits of V. The syndrome can be defined as $S = (S_1, S_2, S_3)$, where

$$S_1 = H\big((\tilde{x}_1, \mathbf{0}) \oplus \tilde{x}_2\big), \quad (19)$$

$$S_2 = p(\tilde{x}_1) \oplus f(\tilde{y}) \oplus \tilde{x}_3, \quad (20)$$

$$S_3 = p(\tilde{x}_1) \oplus p(\tilde{x}_2) \oplus p(\tilde{x}_3) \oplus p(\tilde{x}_4). \quad (21)$$

The error correction algorithm is as stated below. Similar to the design described in [38], only single errors in the information part of the code will be corrected. If single errors in the redundant portion or multi-bit errors are detected, ERR will be asserted but no correction will be attempted.

1. Compute by Eqs. 19, 20, 21 the syndrome of the code $S = (S_1, S_2, S_3)$, where $S_1 \in GF(2^{\lceil \log_2(m+1) \rceil})$ and $S_2, S_3 \in GF(2)$.
2. If S is the all zero vector, then no error is detected. Otherwise one or more errors are detected.
3. If $S_3 = 0$ and at least one of S_1, S_2 is nonzero, errors with even multiplicities are detected and ERR will be raised. Errors in this class are uncorrectable because all of them are multi-bit errors.
4. If $S_3 = 1$ and $S_1 = \mathbf{0}$, a single bit error occurs to one of the last two redundant bits of the code. ERR will be asserted and the data will go through ECC without any correction because only single bit errors in the information part need to be corrected.
5. If $S_3 = 1, S_1 \neq \mathbf{0}$ and S_1 does not match any columns of the parity check matrix H, an uncorrectable multi-bit error of an odd multiplicity is detected and ERR will be raised.
6. If $S_3 = 1$ and $S_1 = h_i$, where h_i is the i_{th} column of H, a single bit error in the first two parts of the code or multi-bit errors are detected. Without

loss of generality, we assume that the first $k_V = m - \lceil \log_2(m+1) \rceil$ bits of V are information bits.

(a) If $a \leq k_V$ and $1 \leq i \leq a$, flip the i_{th} bit of x_1, recalculate S_2. If $S_2 = 0$, the single error is at the i_{th} bit of x_1 and is successfully corrected. Otherwise the single error is at the i_{th} bit of x_2.
(b) If $a \leq k_V$ and $a < i \leq k_V$ flip the i_{th} bit of x_2, recalculate S_2. If $S_2 = 0$, the single error is in the i_{th} bit of x_2 and is successfully corrected. Otherwise multi-bit errors with odd multiplicities are detected.
(c) If $a \leq k_V$ and $i > k_V$, the error occurs to the redundant bits of V and does not need to be corrected. ERR will be asserted and no correction will be attempted.
(d) Similar procedures can be applied to the case when $a > k_V$.

Example 11 In this example we show the encoding and decoding procedure for a (39, 32, 4) extended Vasil'ev code C with $a = 6$. When $a = 6$, $u \in GF(2^6)$, $v = (y, z) \in GF(2^{31})$, $k_V = 26$, where $y \in GF(2^{26})$ are the information bits and $z \in GF(2^5)$ are the redundant bits. Let

$$11111001011011000110010111001111 \in GF(2^{32})$$

be the message that needs to be encoded. Then $u = 111110$ and $p(u) = 1$. y can be computed by XOR $(u, \mathbf{0})$, $\mathbf{0} \in GF(2^{|a-k_V|})$ with the other $k_V = 26$ bits of the message. Thus

$$y = 10100011000110010111001111 \in GF(2^{26}).$$

Let

$$H = \begin{pmatrix} 1111101110110100111100000010000 \\ 1111011101101010100011100001000 \\ 1110111011011001010010011000100 \\ 1101110111000111001001010100010 \\ 10111110000111111000100101100001 \end{pmatrix}$$

be the parity check matrix of V. Then the redundant bits z of $v \in V$ are 00101 and $p(v) = 0$. Let f be the non-repetitive quadratic function as described in Example 5, then $f(y) = 0$. The last two nonlinear redundant bits are 11. The entire codeword is

$$111110010110110001100101110011110010111 \in GF(2^{39}).$$

Suppose a single bit error occurs to the 9_{th} bit of the codeword. After receiving the distorted codeword, S_1, S_2 and S_3 can be computed according to Eqs. 19, 20, 21. We have $S_1 = h_3 = 11101$, $S_2 = 0$, $S_3 = 1$. The 3_{rd} bit of x_1 is flipped and S_2 is recomputed. The new value of S_2 is one. So the error is at the 3_{rd} bit of the x_2, i.e. 9_{th} bit of the entire codeword.

The sizes of K_d and K_c for the extended Vasil'ev code can be computed according to the next theorem.

Theorem 10 *For $(a+m+2, a+k_V, 4)$ extended Vasil'ev codes, where $k_V = m - \lceil \log_2(m+1) \rceil$, let $t = min\{a, k_V\}$, there are 2^a undetectable errors and $2^{a+1}(2^{k_V} - 1)$ conditionally detectable errors. If only errors occurring to the information part of the code are corrected, the number of miscorrected errors is $2t(2^{a+k_V} - 1) + (2^t - 1)|a - k_V|$. The number of conditionally miscorrected errors is $2|a - k_V|(2^{a+k_V} - 2^t)$. The probability of error masking for conditionally detectable errors and the probability of miscorrection for conditionally miscorrected errors are bounded by P_f, which is the nonlinearity of f defined by Eq. 4.*

Proof The syndrome of the code can be re-written as follows.

$$\begin{aligned} S_1 &= H\big((\tilde{x}_1, \mathbf{0}) \oplus \tilde{x}_2\big) = H\big((e_1, \mathbf{0}) \oplus e_2\big), \\ S_2 &= p(\tilde{x}_1) \oplus f(\tilde{y}) \oplus \tilde{x}_3 \\ &= f(\tilde{y}) \oplus f(y) \oplus p(e_1) \oplus e_3, \\ S_3 &= p(\tilde{x}_1) \oplus p(\tilde{x}_2) \oplus p(\tilde{x}_3) \oplus p(\tilde{x}_4) \\ &= p(e_1) \oplus p(e_2) \oplus p(e_3) \oplus p(e_4). \end{aligned}$$

1. $K_d = \{e | S_1 = \mathbf{0} \in GF(2^{\lceil \log_2(m+1) \rceil}), S_2 = S_3 = 0 \in GF(2), \forall c \in C\}$. Since $S_1 = H((e_1, \mathbf{0}) \oplus e_2) = \mathbf{0}$, $(e_1, \mathbf{0}) \oplus e_2$ is a codeword of the linear code V. Because f is a nonlinear function, the only possibility to guarantee $S_2 = 0, \forall c \in C$ is that $(e_1, \mathbf{0}) = e_2$, $p(e_1) = e_3$. $S_3 = 0$, thus $e_4 = e_3 = p(e_1)$. So the detection kernel of the code contains all error vectors $e = (e_1, e_2, e_3, e_4)$ such that $(e_1, \mathbf{0}) = e_2$, $e_3 = e_4 = p(e_1)$. The number of errors in this class is 2^a;
2. If $(e_1, \mathbf{0}) \oplus e_2$ is a nonzero codeword of V and $e_4 = p(e_1) \oplus p(e_2) \oplus p(e_3)$, then $S_1 = \mathbf{0}$, $S_3 = 0, \forall c \in C$. S_2 can be either one or zero depending on the information part of the code. These errors will be conditionally detected. The error masking probability is bounded by P_f. If f is a perfect nonlinear function, these errors will be detected with probability 0.5. The number of errors in this class is $2^{a+1}(2^{k_V} - 1)$.

3. Multi-bit error e will be miscorrected as single bit errors occurring to the information part of the code if and only if $S_3 = 1$, $S_1 = h_i$, $1 \leq i \leq max\{a, k_V\}$. Let $t = min\{a, k_V\}$.
   (a) If $1 \leq i \leq t$, e will always be miscorrected as a single error in the i_{th} bit of either x_1 or x_2. The number of pairs of e_1, e_2 satisfying $S_1 = H((e_1, \mathbf{0}) \oplus e_2) = h_i$, $1 \leq i \leq t$ is $t \times 2^{a+k_V}$. e_3 can be either one or zero. $e_4 = p(e_1) \oplus p(e_2) \oplus p(e_3) \oplus 1$. So there are $2t \times 2^{a+k_V}$ errors that satisfy $S_3 = 1$, $S_1 = h_i$, $1 \leq i \leq t$. $2t$ of them are correctly corrected. The number of miscorrected errors in this class is $2t(2^{a+k_V} - 1)$.
   (b) If $t < i \leq max\{a, k_V\}$, the number of errors satisfying $S_3 = 1$, $S_1 = h_i$ is $2|a - k_V| \times 2^{a+k_V}$. After flipping the i_{th} bit of either $\tilde{x}_1$ or $\tilde{x}_2$, S_1 and S_3 become zero. Denote by $\hat{e}_1, \hat{e}_2$ the new error vectors after flipping the bit for the first two parts of the codewords.
      i. If $(\hat{e}_1, \mathbf{0}) = \hat{e}_2$ and $e_3 = p(\hat{e}_1)$, S_2 is always zero. The number of errors in this class is $2^t \times |a - k_V|$ and $|a - k_V|$ of them are correctly corrected. The number of miscorrected errors is $(2^t - 1)|a - k_V|$.
      ii. If $(\hat{e}_1, \mathbf{0}) = \hat{e}_2$ and $e_3 \neq p(\hat{e}_1)$, S_2 is always one. Errors in this class are always detectable. The number of them is $2^t|a - k_V|$.
      iii. If $(\hat{e}_1, \mathbf{0}) \neq \hat{e}_2$, then S_2 can be either one or zero depending on the information bits of the code. Errors in this class will be conditionally miscorrected. The probability of miscorrection is bounded by P_f. If f is a perfect nonlinear function, the probability of miscorrection is 0.5. The number of errors in this class is $2|a-k_V|(2^{a+k_V} - 2^t)$.

□

The sizes of the detection and the correction kernel are functions of a and m. For any extended Vasil'ev codes with length n and number of information bits k, we have

$$k = a + k_V = a + m - \lceil \log_2(m + 1) \rceil,$$

$$n = a + m + 2,$$

$$a \leq m.$$

Hence $n - 2^{n-k-2} - 1 \leq a \leq \lfloor \frac{n-2}{2} \rfloor$. When $n = 39$, $k = 32$, $6 \leq a \leq 18$. Figure 3 shows the sizes of the detection and the correction kernel for $(39, 32, 4)$ extended Vasil'ev codes for different a. The minimum sizes of the detection and the correction kernel are 2^6 and $12(2^{32} - 1) + 20(2^6 - 1)$ respectively, both of which are achieved when $a = 6$. Different from traditional linear error detecting codes, extended Vasil'ev codes have conditionally undetectable or miscorrected errors. For $(39, 32, 4)$ extended Vasil'ev code with $a = 6$, the numbers of errors which are masked or miscorrected with probability 0.5 are $2(2^{32} - 2^6)$ and $40(2^{32} - 2^6)$ respectively.

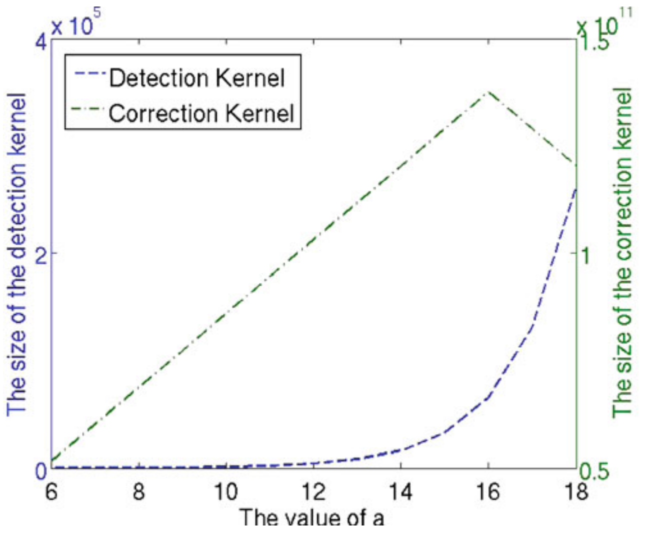

Fig. 3 Kernels of $(39, 32, 4)$ extended Vasil'ev codes as a function of "a"

6.3.2 Hardware Implementation of the Encoder and the Decoder for the Extended Vasil'ev Codes

The encoder for the $(39, 32, 4)$ extended Vasil'ev code is similar to the encoder for the $(39, 32, 4)$ extended Hamming code. The main difference between them is that the encoder for the extended Vasil'ev code needs to realize one nonlinear function f, which requires thirteen 2-input AND gates and twelve 2-input XOR gates given the fact that $k_V = 26$. The latency penalty, the hardware overhead and the power consumption of the encoder for the extended Vasil'ev code is comparable to that of the linear extended Hamming code (Table 6).

The architecture of the decoder for the extended Vasil'ev code is shown in Fig. 4. After new data arrives, the syndrome $S = (S_1, S_2, S_3)$ is computed. If single errors occurring to the information part of the code is detected, we first assume that the error is in x_2 and attempt to correct the error by flipping the erroneous bit in x_2. Then S_2 will be re-computed. If S_2 becomes zero, then the error is successfully corrected. Otherwise the error is in x_1. The bit in x_2 will be flipped back and

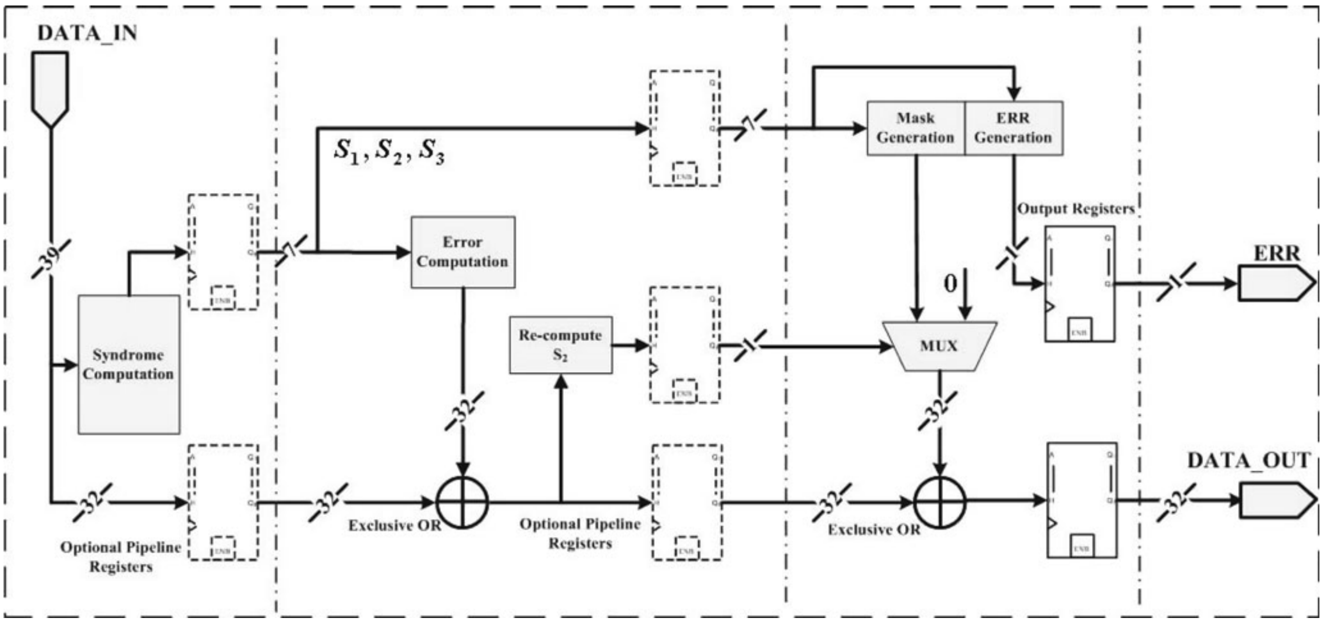

Fig. 4 The decoder architecture for the (39, 32, 4) extended Vasil'ev code

the distorted bit in x_1 will be corrected, which is done by masking the output of the first exclusive OR network with some mask pre-computed according to the value of the syndrome.

Pipeline registers can be added to increase the throughput of the system. Possible locations for the pipeline registers are shown in Fig. 4. In Section 6.4.2, the latency penalty, the hardware overhead and the power consumption of the non-pipelined decoder for the extended Vasil'ev code will be shown and compared to that of the other two alternatives.

6.4 Comparison of Memory Architectures Based on Extended Hamming Codes, Extended Vasil'ev Codes and Extended Phelps Codes

6.4.1 Error Detection and Correction Capabilities

In Table 4 we summarize Q_{mc}, the size of the detection kernel K_d and the size of the correction kernel K_C of the $(39, 32, 4)$ extended Hamming code, the $(39, 32, 4)$ extended Vasil'ev code with $\omega = 6$, $Q_{mc} = 0.5$ (Example 6) and the $(39, 32, 4)$ extended Phelps code with $\omega_d = 27$, $Q_{mc} = \frac{1}{16}$ (Example 8). The extended Hamming code has the largest size of K_d and the largest size of K_C among the three alternatives. The extended Vasil'ev code has only 2^6 undetectable errors. The size of K_C for the extended Vasil'ev code is nearly one third of that for the extended Hamming code. The extended Phelps code has the smallest Q_{mc} and the smallest size of K_C. Its K_d is larger than that of the extended Vasil'ev code but is only $\frac{1}{32}$ of the size of K_d for the extended Hamming code.

Table 5 shows the number of undetectable and miscorrected errors with multiplicities three to six for the three alternatives. All the three codes have no undetectable single, double and triple errors. Only errors with odd multiplicities are miscorrected. The total number of miscorrected errors with multiplicities less or

Table 4 Detection and correction kernels of the $(39, 32, 4)$ extended Hamming code, the $(39, 32, 4)$ extended Vasil'ev code with $\omega_d = 6$, $Q_{mc} = 0.5$ and the $(39, 32, 4)$ extended Phelps code with $\omega_d = 27$, $Q_{mc} = \frac{1}{16}$

	Q_{mc}	Size of K_d	Size of K_C
Hamming	1	2^{32}	$32 \cdot 2^{32} - 1$
Vasil'ev	0.5	2^6	$\approx 12 \cdot (2^{32} - 1)$
Phelps	$\frac{1}{16}$	2^{27}	$32 \cdot (2^{27} - 1)$

Table 5 Number of undetectable and miscorrected errors with multiplicities less or equal to six for the $(39, 32, 4)$ extended Hamming code, the $(39, 32, 4)$ extended Vasil'ev code with $\omega_d = 6$, $Q_{mc} = 0.5$ and the $(39, 32, 4)$ extended Phelps code with $\omega_d = 27$, $Q_{mc} = \frac{1}{16}$

	Code	$\|\|e\|\|=3$	$\|\|e\|\|=4$	$\|\|e\|\|=5$	$\|\|e\|\|=6$
$\|K_d\|$	Ext. Hamming	0	1583	0	51744
	Ext. Vasil'ev	0	21	0	0
	Ext. Phelps	0	364	0	3362
$\|K_c\|$	Ext. Hamming	5176	0	254432	0
	Ext. Vasil'ev	1635	0	108993	0
	Ext. Phelps	2263	0	42692	0

equal to six for the extended Vasil'ev code is less than one half of the corresponding number for the extended Hamming code. The number of miscorrected errors with multiplicity three to six for the extended Phelps code is the smallest of the three, which makes it a good choice for error correction purpose.

We note that errors $e = (e_1, e_2, e_3, e_4)$ in the detection kernel of the extended Phelps codes satisfy

$$e_1 \in C, e_2 = p(e_1), e_3 \in B, e_4 = p(e_3).$$

Errors in the correction kernel of the extended Phelps codes satisfy either

$$[e_1] = [e_i^*], e_2 = p(e_1) \oplus 1, e_3 \in B, e_4 = p(e3),$$

or

$$e_1 \in C, e_2 = p(e_1), [e_3] = [e_i^*], e_4 = p(e_3) \oplus 1.$$

Obviously, the number of undetectable triple errors and miscorrected quadruple errors of the extended Phelps codes can be further reduced by minimizing the number of codewords of Hamming weight three and four in codes C and B.

6.4.2 Area Overhead, Power Consumption and Latency

The encoder and the non-pipelined decoder for all the three codes have been modeled in Verilog and synthesized in Cadence Encounter RTL Compiler using the Nangate 45nm Opencell library (http://www.nangate.com). The designs were placed and routed using Cadence Encounter.

The latency, area overhead and the power consumption of the encoders and decoders for the memory protection architectures based on the three alternatives are shown in Table 6 and 7. The data is for the typical operation condition assuming a supply voltage of 1.1 V and a temperature of 25°C.

The encoder for the extended Vasil'ev code has almost the same area overhead and power consumption as that of the linear extended Hamming code. The decoder for the extended Vasil'ev code requires 23% more area overhead and consumes 17.15% more power than that of the extended Hamming code. In terms of the latency, the architectures based on the extended Vasil'ev code results in 21.2% more latency for the decoder and 26.5% more latency for the encoder compared to the extended Hamming code. We note that the output of the memory can be directly forwarded to the processors before it goes through the decoder. When the decoder detects or corrects errors, the processor can be stalled and the data can be re-fetched from either the output of the decoder (error correction) or the memory (error detection). In this case the latency penalty for the decoder, which is the main factor that affects the performance of the memory system, is only incurred when there are errors.

For both the encoder and the decoder, the Phelps code has the largest penalty in all the three aspects. Compared to the extended Hamming code, the latency penalty of the extended Phelps code is increased by 47.9% for the encoder and increased by 24.5% for the decoder. The area overhead and the power consumption of the decoder for the extended Phelps code is almost three times of that for the extended Hamming code. However, for reliable memory systems the area and power consumption overhead is mainly contributed by the redundant memory cells, which is determined by the number of redundant bits of the error correcting code. Since all the three alternatives have the same number of redundant bits, they have similar area and power consumption overhead.

Thereby, compared to the extended Hamming code, the extended Vasil'ev code and the extended Phelps code can achieve better error detection and correction capabilities at the cost of only a small penalty in latency, area overhead and the power consumption.

Table 6 Latency, area overhead and power consumption for the encoders for (39, 32, 4) SEC-DED codes (voltage = 1.1 V, temperature = 25°C)

	Latency (ns)	Area (μm^2)	Power (mW)
Extended Hamming	0.290	282.2	0.2898
Extended Vasil'ev	0.367	296.1	0.2916
Extended Phelps	0.429	383.0	0.4728

Table 7 Latency, area overhead and power consumption for the decoders for (39, 32, 4) SEC-DED codes (voltage = 1.1 V, temperature = 25°C)

	Latency (ns)	Area (μm^2)	Power (mW)
Extended Hamming	0.538	620.3	0.7119
Extended Vasil'ev	0.652	763.2	0.8340
Extended Phelps	0.670	1,799.8	1.774

6.5 Further Discussions

We note that errors in the decoders for the proposed nonlinear SEC-DED codes may also compromise the reliability of the memories. To protect the decoder against errors caused by temporary faults like SEU and MBU, a *fault secure* design is required. A circuit is fault secure for a fault set ϵ if every fault in ϵ can be detected as long as it manifests at the output of the circuit. The design of fault secure circuits is well studied in the community (see e.g. [34]). Most of the existing technologies of designing fault secure circuits (e.g. methodologies based on duplication and two-rail code checker [34]) can be directly applied to build fault secure decoders for the nonlinear SEC-DED codes.

We also note that all the algorithms and architectures proposed in this paper can be applied to memories with larger word sizes (e.g. 64-bit or 128-bit). Moreover, the extended Vasil'ev codes can be further generalized to correct multi-bit errors. For more information about nonlinear multi-bit error correcting codes, please refer to [41], where a (8281, 8201, 11) nonlinear 5-error-correcting code has been applied for the protection of multi-level cell (MLC) NAND flash memories.

7 Conclusion

In this paper memory protection architectures based on nonlinear SEC-DED codes are proposed. We generalize the constructions of the existing perfect nonlinear Hamming codes to generate nonlinear partially robust codes with any length. The optimality of the codes is demonstrated. The error detection and correction capabilities of these codes are analyzed and compared to linear codes.

The two nonlinear SEC-DED codes generalized in the paper – the extended Vasil'ev codes and the extended Phelps codes—are viable alternatives to replace the traditional linear extended Hamming codes to protect memories for situations where the likelihood of multi-bit errors is high or errors tend to repeat themselves. We present error detecting and correcting algorithms for these nonlinear codes and describe their hardware implementation. The number of undetectable and miscorrected multi-bit errors for these codes is much smaller than that for traditional linear error correcting codes. In the presence of multi-bit distortion or in the case of repeating errors, these codes can provide much better protection with a small increase in latency, area overhead and the power consumption. Different from linear codes, the nonlinear SEC-DED codes have conditionally detectable (miscorrected) errors. The detection (correction) of these errors is message-dependent. This makes these codes useful to detect (correct) repeating errors, e.g. hard errors caused by permanent faults.

The proposed protection architectures are not targeted for any special memory architecture. They can be applied to nearly all types of memories such as RAM, ROM, FLASH and disk memories.

As far as we know, this paper and or previous paper [40] are the only papers discussing application of efficient nonlinear codes for design of reliable memories.

Acknowledgment We would like to thank Professor Ajay Joshi of Boston university for his helpful suggestions to this paper.

References

1. Berger JM (1961) A note on an error detection code for asymmetric channels. Inf Control 4:68–73
2. Bhattacharryya D, Nandi S (1997) An efficient class of SEC-DED-AUED codes. In: Third international symposium on parallel architectures, algorithms, and networks
3. Bose B (1984) Unidirectional error correction/detection for VLSI memory. In: Proceedings of the 11th annual international symposium on computer architecture
4. Carlet C, Ding C (2004) Highly nonlinear mappings. J Complex 20(2–3):205–244
5. Chen CL (1983) Error-correcting codes with byte error-detection capability. IEEE Trans Comput C-32:615–621
6. Chen CL (1996) Symbol error correcting codes for memory applications. In: Proceedings of the twenty-sixth annual international symposium on fault-tolerant computing (FTCS '96)
7. Dunning LA (1985) SEC-BED-DED codes for error control in byte-organized memory systems. IEEE Trans Comput 34:557–562
8. Dutta A, Touba NA (2007) Multiple bit upset tolerant memory using a selective cycle avoidance based SEC-DED-DAEC code. In: 25th IEEE VLSI test symposium (VTS'07)
9. Eto A, Hidaka M, Okuyama Y, Kimura K, Hosono M (1998) Impact of neutron flux on soft errors in mos memories. In: Electron devices meeting
10. Etzion T, Vardy A (1994) Perfect binary codes: constructions, properties, and enumeration. IEEE Trans Inf Theory 40: 754–763
11. Gaubatz G, Sunar B, Karpovsky MG (2006) Non-linear residue codes for robust public-key arithmetic. In: Workshop on fault diagnosis and tolerance in cryptography (FDTC '06)
12. Georgakos G, Huber P, Ostermayr M, Amirante E, Ruckerbauer F (2007) Investigation of increased multi-bit failure rate due to neutron induced seu in advanced embedded srams. In: Symposium on VLSI circuits digest of technical paper
13. Halfhil TR (2005) Z-ram shrinks embedded memory. Microprocessor Report, Tech. Rep.
14. Hamming RW (1950) Error correcting and error detecting codes. Bell Syst Tech J
15. Hsiao MY (1970) A class of optimal minimum odd-weight-column SEC-DED codes. IBM J Res Develop 14:395–401
16. Johnston AH (2000) Scaling and technology issues for soft error rates. In: Ser. 4th annual research conference on reliability
17. Karpovsky MG, Taubin A (2004) A new class of nonlinear systematic error detecting codes. IEEE Trans Inf Theory 50(8):1818–1820
18. Karpovsky M, Kulikowski K, Taubin A (2004) Differential fault analysis attack resistant architectures for the advanced encryption standard. In: Ser. proc. IFIP world computing congress, Cardis, pp 177–193

19. Karpovsky M, Kulikowski K, Taubin A (2004) Robust protection against fault-injection attacks on smart cards implementing the advanced encryption standard. In: Ser. proc. int. conference on dependable systems and networks (DNS 2004)
20. Karpovsky MG, Kulikowski K, Wang Z (2007) Robust error detection in communication and computation channels. In: Int. workshop on spectral techniques
21. Karpovsky MG, Stankovic RS, Astola JT (2008) Spectral logic and its applications for the design of digital devices. Wiley
22. Kulikowski K, Wang Z, Karpovsky MG (2008) Comparative analysis of fault attack resistant architectures for private and public key cryptosystems. In: Proc of int. workshop on fault-tolerant cryptographic devices
23. Lala P (1978) An adaptive double error correction scheme for semiconductor memory systems. Digit Process 4:237–243
24. Lala PK (2003) A single error correcting and double error detecting coding scheme for computer memory systems. In: Proceedings of the 18th IEEE international symposium on defect and fault tolerance in VLSI systems
25. Lisbôa, CA, Erigson MI, Carro L (2007) System level approaches for mitigation of long duration transient in future technologies. In: 12th IEEE European test symposium (ETS'07)
26. MacWilliams F, Sloane N (1983) The theory of error-correcting codes. North Holland
27. Maiz J, Hareland S, Zhang K, Armstrong P (2003) Characterization of multi-bit soft error events in advanced srams. In: IEEE int'l electronic device meeting, pp 519–522
28. Maxwell MS (2005) Almost perfect nonlinear functions and related combinatorial structures. PhD dissertation, ISU
29. Mollard M (1986) A generalized parity function and its use in the construction of perfect codes. SIAM J Algebr Discrete Methods 7:113–115
30. Moore SK (2007) Masters of memory. IEEE Spectrum 44(1):45–49
31. Penzo L, Sciuto D, Silvano C (1995) Construction techniques for systematic SEC-DED codes with single byte error detection and partial correction capability for computer memory systems. IEEE Trans Inf Theory 41(2):584–591
32. Phelps KT (1983) A combinatorial construction of perfect codes. SIAM J Algebr Discrete Methods 4:398–403
33. Phelps KT, Levan M (1995) Kernels of nonlinear hamming codes. Designs Codes Cryptogr 6(3):247–257
34. Rao TRN, Fujiwara E (1989) Error-control coding for computer systems. Prentice-Hall, Upper Saddle River, NJ
35. Reddy S (1978) A class of linear codes for error control in byte-per-card organized digital systems. IEEE Trans Comput C-27:455–459
36. Satoh S, Tosaka Y, Wender SA (2000) Geometric effect of multiple-bit soft errors induced by cosmic ray neutrons on drams
37. Swift M, Guertin, SM (2000) In-flight observations of multiple-bit upset in DRAMs. IEEE Trans Nucl Sci 47(6): 2386–2391
38. Tam S (2006) Application note: single error correction and double error detection. XILINX
39. Vasil'ev JL (1962) On nongroup close-packed codes. Probl Kibern 8:375–378
40. Wang Z, Karpovsky M, Kulikowski K (2009) Replacing linear hamming codes by robust nonlinear codes results in a reliability improvement of memories. In: IEEE/IFIP international conference on dependable systems and networks, 2009. DSN '09, 29 June–2 July 2009, pp 514–523
41. Wang Z, Karpovsky M, Joshi A (2010) Reliable MLC NAND flash memories based on nonlinear t-error-correcting codes. In: IEEE/IFIP international conference on dependable systems and networks, 2010. DSN '10
42. Whitaker S, Cameron K, Maki G, Canaris J, Owsley P (1991) VLSI reed-solomon processor for the hubble space telescope. In: VLSI signal processing IV. IEEE Press

Zhen Wang received his B.S and M.S degree in Information and Communication Engineering from Zhejiang University, China in 2006. He is now a Ph.D candidate in the Reliable Computing Laboratory of Boston University, United States under the supervision of Prof. Mark G. Karpovsky. His research is focused on the design of robust codes and the application of robust codes in building reliable and secure devices. He also conducts research in the implementation of hardware systems based on ASIC design flows.

Mark Karpovsky has been Professor of Electrical and Computer Engineering, and Director of the Reliable Computing Laboratory since 1983. He conducts research in the areas of cryptography, side-channel attacks, design of cryptographic hardware resistant to attacks, new techniques for design of reliable multiprocessors, networks of workstations and local area networks, routing in computer and communications networks, testing, and diagnosis of computer networks combining on-line and off-line techniques for error detection and/or location, and fault-tolerant message routing for computer networks. Dr. Karpovsky teaches graduate courses on interconnection networks, computer hardware testing, fault-tolerant computing, and error-correcting codes. Dr. Karpovsky has been a consultant for IBM, Digital Corp., Honeywell, and AT&T, and is currently Director of the Reliable Computing Laboratory. He has published more than 200 papers and several books in the areas of design of secure devices, testing and diagnosis, fault-tolerant computing and error-correcting codes. Dr. Karpovsky is a Fellow of the IEEE.

Konrad J. Kulikowski was born in Warsaw, Poland in 1980 and grew up in a small town called Nowy Dwor Mazowiecki. He moved to the United States with his family in 1989. In 1999 he attended Boston University where in 2003 he graduated with Bachelor of Science and Masters of Science degrees in Computer Systems Engineering. The summer after his graduation he started working toward his Ph.D in the Reliable Computing Laboratory at Boston University under the supervision of Prof. Mark G. Karpovsky and Prof. Alexander Taubin. He received his Ph.D degree in the summer of 2008.