IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 December 2022, accepted 6 February 2023, date of publication 22 February 2023, date of current version 27 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3246726

RESEARCH ARTICLE

On the Positional Single Error Correction and Double Error Detection in Racetrack Memories

AWAIS SAEED[1], UBAID U. FAYYAZ[1], AHSAN TAHIR[1], SEOKIN HONG[2], (Member, IEEE), AND TAYYEB MAHMOOD[1]

[1]Department of Electrical Engineering, University of Engineering and Technology, Lahore, Lahore 39161, Pakistan
[2]Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon 16419, Republic of Korea

Corresponding author: Tayyeb Mahmood (tayyeb@uet.edu.pk)

This work was supported by the National Research Foundation of Korea (NRF) Grant by the Korean Government through Ministry of Science and ICT (MSIT) under Grant 2022R1C1C1012154.

ABSTRACT In the era of non-volatile memories, the racetrack memory is a promising technology to pack hundreds of bits in a magnetic nanowire. A solid-state read head is grown alongside the nanowire to sense individual bits which are pushed across the head by a shifting force. However, the probabilistic nature of this shifting movement inflicts positional errors. Therefore, robust and low-cost error correcting codes are essential for a reliable alternative in on-chip memories and storage applications. Recent works focus on Varshamov-Tenengolts (VT) codes which can correct all single bit insertions and deletions. However, VT codes are incapable of detecting multiple deletions/insertions. Because a positional error corrupts multiple data words, multi-bit positional error detection is critical for racetrack memories. In this article, we propose a novel positional single error correction and double error detection (P-SECDED) code in the context of racetrack memories with a single read head. In particular, we adopt a postamble-based approach where a VT-encoded codeword appends a carefully selected bit-pattern, stored on the racetrack. We rigorously analyze the limitations of the postamble method, and deduce a criterion for postamble selection. We further provide a methodology to optimize this postamble selection in order to correct all single-bit errors and as much of two-bit errors as possible. Finally, we prove that all incurable two-bit errors are successfully detected. To the best of our knowledge, this work is the first attempt to provide P-SECDED fault-tolerance to three-dimensional racetrack memories which cannot afford multiple read heads.

INDEX TERMS Magnetic storage, solid-state storage, ultra-dense storage, insertion deletion channels, domain-wall memories, skyrmions memories, VT codes, synchronization errors, NVM, DWM, ECC.

I. INTRODUCTION

Ultra-dense data storage has become a crucial requirement in the modern era of artificial intelligence and metaverse [1] where computers and datacenters are always in need of an extra available capacity of storage. Flash-based solid-state storage (SSD) are replacing the rotating disks (hard disks), due to better response times, and power efficiencies. However, volumetric efficiency of hard disks is still superior to SSDs. Research into novel non-volatile memory (NVM) technologies is focused on circumventing this limitation. In particular, an emerging candidate is racetrack memory (RM) that uses minuscule magnetic domains along a nanowire to store binary data as magnetic poles [2]. When grown vertically, racetrack memory offers unprecedented ultra-high storage density, [3], [4] because it can store more than 100 bits per access point, compared to storing one bit in SLC, 2 bits in MLC and 3 bits in TLC technologies, making it an ultimate 3D solid-state storage [5].

Racetrack memory consists of magnetic nanowires which can be grown vertically or horizontally on a silicon substrate [2], [6], as shown in Fig. 1. The isolation between

The associate editor coordinating the review of this manuscript and approving it for publication was Adnan Abid.

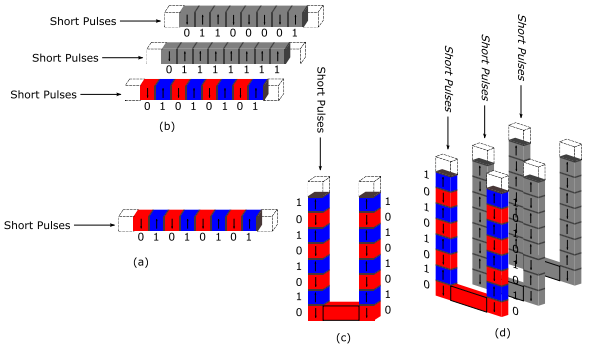

FIGURE 1. (a) Horizontal RM. (b) Horizontal RM Array. (c) Vertical RM. (d) Vertical RM Array.

individual magnetic domains is ensured by so-called domain walls. A Tunnel magnetoresistance junction (TMJ) is formed along-side nanowire to serve as a read head. Because head is fixed, the only way to sense magnetic domains is to move them along the nanowire. A spin-aligned current can induce this motion. To align a magnetic domain with the read head, nanoscopic pinning sites are etched as checkpoints along the racetrack.

Due to process variations in sub-100nm regime, the domain wall motion is probabilistic [7]. In particular, due to variations in shift current density, an arbitrary domain may step over or stick to the read head. From a data read perspective, one or more bits are deleted or inserted randomly. It is obvious the conventional block error correcting codes (ECC) that work on bit-flips cannot recover from these synchronization errors without exorbitantly large complexity. Therefore, the literature treats such information channels as deletion (insertion) channels. The impact of a bit deletion in a single racetrack may extend to multiple words when data words are interleaved, an effective method to increase throughput and fault-tolerance. It is tempting to consider block codes feasible in above scenario because out-of-sync bits will be position in-variant in interleaved words, as shown in Fig. 2, and proposed in [8]. However, block codes will quickly crumble in the presence of deletions in more than one racetracks in the block. As such, an n-racetracks array would require an n-bit error correcting code, that exposes severe scalability challenges of this approach. Recently, complex coding schemes including Low-density-parity-codes [9] and Polar codes [10] were applied to large data blocks stored on the Racetrack arrays to mitigate deletion and insertion errors. However, these codes are known to be associated with high implementation costs.

The positional error mitigation in racetrack memories has attracted a lot of interest recently. In [11], authors proposed to engineer racetrack shift mechanism in order to avoid incomplete shifts. The further proposed a p-ECC technique that leverages flexibility of horizontal racetracks to afford multiple read heads, with preambles and postambles, to demonstrate a single error correction and double error detection (SECDED). In [12] and [13], authors rigorously analyzed the limitations of multiple-head positional error corrections and found the quantum distances between heads for correction robustness. On the other hand, [8] demonstrated how to correct a single deletion error with Varshamov-Tenengolts (VT) codes. VT-codes can correct all single deletions and insertions, however, cannot detect (or possibly correct) multi-bit errors. For two deletions, authors proposed interleaving with cross-dimensional parity codes to detect and correct two deletions in a single racetrack in the group. Because VT-codes do not mandate multiple read heads, they can be used with vertically grown racetrack memories with highest storage densities. However, their sequential nature raises performance concerns. Therefore, a number of high performance and area/energy-efficient architectures of VT-encoders/decoders were presented in [14], [15], and [16]. More recently, [17] presented a Radix-2 VT coding that doubles the throughput and energy-efficiency with the same area budget as former methods. However, its single-error correction capability must be augmented with a multi-bit error detection in order to realize a true positional single error correction and double error detection (P-SECDED). Although multi-head solutions are previously proposed and thoroughly analyzed, such solutions are limited to applications with horizontal racetracks and an effort must be focused towards VT-code based P-SECDED for single-read heads in vertical racetracks.

In this article, we propose a postamble based approach towards two-bit error detection, in addition to single error correction with VT-codes. We show that postamble selection is not arbitrary and/or straight-forward and devise a mechanism to perform postamble selection. Our main contributions are as follows.

- We elaborate a mathematical model to classify positional errors through a two deletions insertions channel.
- We formulate postambles for each error class (group) into insertion and deletion postamble pattern (PP) sets. We establish constraints on joint PP set to successfully correct single errors and detect two errors in a racetrack.
- We further analyze postamble-based detection to note that certain postambles are able to correct a subset of two errors. We present a methodology to search for a postamble that can correct maximum number of two errors. The rest of two errors are flagged.
- Finally, we demonstrate a simple decoder for the proposed P-SECDED.

The rest of the paper is organized as follows. In next section, we describe preliminaries along with systematic VT encoder and decoder. In Section III, we provide mathematical construction of our proposed P-SECDED. It is followed by a rigorous comparison with literature and discussions. Lastly, we conclude the paper in Section V.

II. PRELIMINARIES

We consider a binary VT code $C_a(n)$ of rate $R = k/n$, length n, where $a \in \{0, 1, \ldots, n\}$ is a code parameter referred

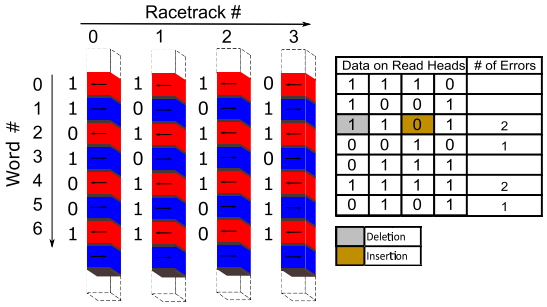

FIGURE 2. Interleaved storage on the racetrack, 4-bit words are interleaved across an array of 4 racetracks. A deletion in first racetrack and an insertion in third racetrack brings upto two errors in the stored words.

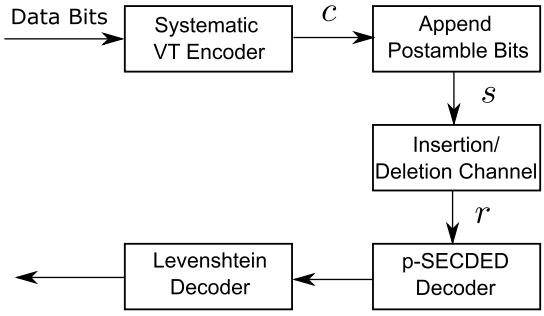

FIGURE 3. The proposed P-SECDED encoder and decoder. (a), k-bit data word is systematically encoded with VT encoder, (b) a c-bit code word is appended with postamble bits to make s-bit codeword, (c) s-bit encoded codeword is stored on the racetrack, (d) s-bit word *r* is read from the racetrack, and two-errors case flagged with P-SECDED decoder, (e) single error cases are resolved with VT decoder, to recover k-bit data word.

to as the *desired syndrome* of the code and $k = n - \lceil \log_2(n+1) \rceil$ is the message length. A valid binary VT codeword $\mathbf{c} = [c_0, c_1, \ldots, c_{n-1}] \in C_a(n,k)$ satisfies

$$S_a(\mathbf{c}) = \sum_{i=1}^{n} i c_i \mod (n+1) = a, \tag{1}$$

where $S_a(c)$ is the syndrome of the VT code. In this paper, we will consider $C_0(n)$ only, as it has the maximum cardinality among all other choices. We will omit a now onward from all notation for brevity.

Abdel-Ghaffar and Ferreira [18] proposed a systematic VT encoder to map all k-bit data combinations onto VT codewords from $C_a(n,k)$. The number of parity bits used is given as $t = \lceil \log_2(n+1) \rceil$. The length of the data sequence comes out to be $k = n - t$. Let the set $I = \{1, 2, 3, \ldots, n\}$ be the indices in the codeword $\mathbf{c}$. Let the first t dyadic indices of codeword $\mathbf{c}$ constitute the set $J = \{2^0, 2^1, 2^2, \ldots, 2^t\}$. We denote the non-dyadic indices by the set $\bar{J} = I - J$. Following are the encoding steps for systematic VT encoder:

1) Place k-bit message sequence $\mathbf{m} = [m_0, m_1, \ldots, m_k]$ in non-dyadic indices and fill all dyadic indices with zeros.
2) Calculate the deficiency d by taking the difference of the desired syndrome a and syndrome of the created vector.

$$d = a - S_a(\mathbf{c}) \mod (n+1), \tag{2}$$

3) Let us denote the binary representation of d as $[d_{t-1}, \ldots, d_1, d_0]$. Replace zeros in dyadic indices with the binary representation of deficiency d such that $c_{2^i} = d_i$. The syndrome of the resultant vector will now be equal to the desired value of a.

We pass the encoded codeword through the following three types of channels that inflict three types of errors in the transmitted codeword.

1) d-del channel that can delete at most d bits from the transmitted codewords. No insertion occurs in this type of channel.
2) i-in channel can repeat at most i bits in the transmitted codewords. No deletion occurs in this type of channel.
3) s-indel channel that either deletes at most s bits or repeats s bits, but not the both in any transmitted codeword.

In this paper, we consider 2-del, 2-in and 2-indel channels, inflicting one deletion (1D), one insertion 1I, two deletions (2D) and two insertions (2I). All VT codes can correct 1D/I in the transmitted codeword. For completeness, we describe the Levenshtein decoding algorithm. For ID in a codeword $\mathbf{c}$, we receive,

$$\mathbf{r} = [c_1, c_2, \ldots, c_{m-1}, c_{m+1}, \ldots, c_{n-1}, c_n],$$

where c_m is the deleted bit in $\mathbf{c}$. Let the Hamming weight w of the received codeword $\mathbf{r}$ is $\sum_{i=1}^{n-1} r_i$ and the deficiency d is,

$$d = a - S_a(\mathbf{r}) \mod (n+1), \tag{3}$$

In order to make syndrome of received codeword $\mathbf{r}$ equal to desired syndrome a, the Levenshtein decoder estimates the deleted bit's value and its location using the following rule which makes $S_a(\mathbf{r}) = a$:

$$\hat{c_m} = \begin{cases} 0 \text{ on } d \text{ ones from the right} & \text{if } d \leq w \\ 1 \text{ on } (d-1-w) \text{ zeros from left} & \text{otherwise.} \end{cases}$$

III. THE PROPOSED P-SECDED CODE DESIGN

The previous section demonstrated how to map a k-bit sequence onto a n-bit VT codeword from $C_a(n,k)$. Now, we define the construction of the proposed P-SECDED code that can provide a single error correction and a double error detection for a 2-indel channel. We further show that some 2D/I can also be corrected with the proposed code and then deduce a methodology to maximize the number of correctable error patterns. The end-to-end P-SECDED process is illustrated in Fig. 3 where a k-bit data word is systematically encoded with VT encoder. The resulting c-bit codeword is appended with a postamble to make an s-bit codeword. This codeword is stored on the racetrack. When read from the racetrack, the s-bit word is fed to P-SECDED

decoder that flags all 2D/I errors. Alternatively, 1D/I cases are resolved with a VT decoder, to recover the k-bit data word.

A. DEFINITIONS

In P-SECDED, after encoding the message bits to a VT codeword, $\mathbf{c}$, ℓ fixed postamble bits $[p_1, p_2, , \ldots, p_\ell]$ are appended to $\mathbf{c}$. We denote the resulting code as $Z(n, k, \ell)$. In this paper, we only consider VT codes with $a = 0$ and will omit the subscript a for brevity. When a codeword from $Z(n, k, \ell)$ is transmitted through our channel, errors can happen in the VT codeword part, in postamble or in both. At the receiver, we always read $n + \ell - 2$ bits as the received word $\mathbf{r}$. In the case of no errors, the sequence $\mathbf{v} = [r_1, r_2, \ldots, r_n]$ gets the VT codeword while sequence $\mathbf{q} = [r_{n+1}, r_{n+2}, \ldots, r_{n+\ell-2}]$ gets the postamble bits $[p_1, p_2, \ldots, p_{n+\ell-2}]$. In case of deletion errors, $\mathbf{q}$ may contain only postamble bits, whereas in case of insertion errors, $\mathbf{q}$ may contain c_{n-1} and c_n.

Let $\mathcal{D}_m(n)$ and $\mathcal{I}_m(n)$ be the sets of all received $\mathbf{q}$ sequences when any $\mathbf{z} \in Z(n, k, \ell)$ is transmitted through a m-deletion and a m-insertion channel, respectively, with m errors, out of which n errors occur in $\mathbf{c}$. We refer to $\mathcal{D}_m(n)$ and $\mathcal{I}_m(n)$ as a *deletion postamble pattern* (D-PP) and *insertion postamble pattern* (I-PP) sets, respectively, for a given postamble pattern. We also denote the set $\mathcal{V}_m(n) = \mathcal{D}_m(n) \cup \mathcal{I}_m(n)$ as the *postamble pattern* (PP) set. Note that D-PP, I-PP and PP corresponds to a given postamble pattern. Therefore, we have omitted the dependence on postamble pattern in the notation of these sets for brevity and in the rest of the paper, the given postamble pattern will be clear from the context.

Example 1: Consider a code $Z(4, 2, 3)$ *with* $p_1 = p_2 = 1$, $p_3 = 0$ *given below*:

$$Z(4,2,3) = \begin{Bmatrix} 0000110 & 1001110 \\ 0110110 & 1111110 \end{Bmatrix}.$$

$\mathcal{D}_2(1) = \{0, 1\}$ *consists of all the received subsequences* $\mathbf{q} = [r_5]$ *with two deletions while one of them is in the VT codeword part.*

Example 2: Consider a code $Z(4, 2, 4)$ *with* $p_1 = p_2 = 1$, $p_3 = p_4 = 0$ *given below*:

$$Z(4,2,4) = \begin{Bmatrix} 00001100 & 10011100 \\ 01101100 & 11111100 \end{Bmatrix}.$$

$\mathcal{D}_2(1) = \{00, 10\}$ *consists of all the received subsequences* $\mathbf{q} = [r_5, r_6]$ *with two deletions while one of them is in the VT codeword part. The only possibilities for the subsequence* $\mathbf{q}$ *are*:

1) p_2 *is deleted resulting in* $r_5 = p_3 = 0$ *and* $r_6 = p_4 = 0$.
2) p_3 *is deleted resulting in* $r_5 = p_2 = 1$ *and* $r_6 = p_4 = 0$.

Example 3: Consider a code $Z(4, 2, 4)$ *with* $p_1 = p_2 = 1$, $p_3 = p_4 = 0$ *given below*:

$$Z(4,2,4) = \begin{Bmatrix} 00001100 & 10011100 \\ 01101100 & 11111100 \end{Bmatrix}.$$

$\mathcal{I}_2(1) = \{01, 11\}$ *consists of all the received subsequences* $\mathbf{q} = [r_5, r_6]$ *with two insertions while one of them is in the VT codeword part. The only possibilities for the subsequence* $\mathbf{q}$ *are*:

1) 1 *insertion in VT codeword results in* $r_5 = c_4 = \{0, 1\}$ *and* $r_6 = p_1 = 1$.

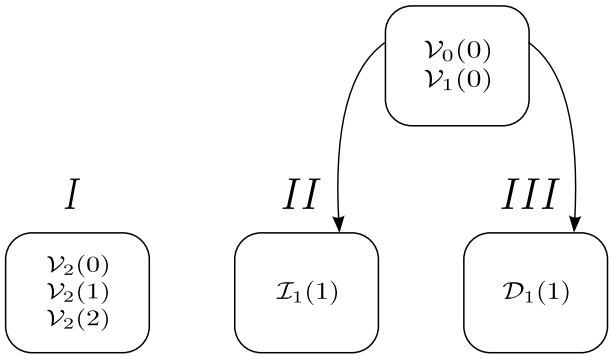

FIGURE 4. 2D/I, 1D and 1I patterns must be distinguishable. 1D and 1I need to be distinguishable because Levenshtein's decoder needs to know the type of error before decoding. $\mathcal{V}_1(0)$ and $\mathcal{V}_0(0)$ both are correctable and can be mapped to either Group II or III.

B. THE POSTAMBLE DESIGN FOR P-SECDED

By definition, P-SECDED design problem is a PP set search problem. $Z(n, k, \ell)$ should have the following constraints:

1) In the case of one deletion, sequence $\mathbf{q}$ should be different from that in the case of one insertion.
2) In the case of two-deletions/insertions, sequence $\mathbf{q}$ should be different from those in case of one-, as well as no deletions/insertions.

Mathematically,

$$\text{find} \quad p_1, p_2, \ldots, p_\ell \tag{4a}$$

$$\text{such that} \quad \mathcal{D}_1(1) \cap \mathcal{I}_1(1) = \emptyset, \tag{4b}$$

$$\left(\bigcup_{i=0}^{2} \mathcal{V}_2(i)\right) \cap \left(\bigcup_{i=0}^{1} \mathcal{V}_1(i)\right) \cap \mathcal{V}_0(0) = \emptyset, \tag{4c}$$

The search problem in (4) constraints the solution (optimal postamble pattern) in (4b) to distinguish one-deletion case from one-insertion case required by the Levenshtein's decoder as it needs to know whether a deletion or insertion has occurred. The problem further constraints in (4c) to distinguish a 2D/I case from 1D/I, and the no-error case. **A solution to the problem (4) gives us an optimal P-SECDED code**.

Lemma 1: $\ell \geq 2$

Proof: In the case of two deletions, we will be reading two bits from the postamble. As a result, we should have at least two bits in the postamble in the case of two-deletion channel implying $\ell \geq 2$. □

Theorem 1: For a P-SECDED on 2D/I channel, $\ell \geq 6$.

Proof: Consider the case of $\ell = 3$. *In this case, we read* $n + \ell - 2 = n + 1$ *and only one bit from the postamble is read as* r_{n+1}. *However, in the case of insertion channels,* $\mathcal{I}_2(2), \mathcal{I}_1(1)$ *codeword bits* c_{n-1} *and* c_n *are read, respectively as* r_{n+1}. *Hence, Eq.4c is not satisfied. Similarly for the case of*

$\ell = 4$, $I_2(2) = \{c_{n-1}, c_n\}$ *and Eq.4c will not be satisfied for any postamble patterns. In the case of* $\ell = 5$, *Table 1 shows different received bits from* r_1 *to* r_3. *To satisfy Eq. 4c,* $p_1 = p_5$, $p_1 \neq p_3$, $p_1 \neq p_4$ *and* $p3 \neq p_4$, *which is not possible.* □

Fig. 4 shows three groups to which a received word with a corresponding PP set should be mapped. As Levenshtein's decoder needs to know if a deletion or insertion has occurred, $\mathcal{I}_1(1)$ and $\mathcal{D}_1(1)$ have to be mapped to different groups. Patterns from $\mathcal{V}_1(0)$ and $\mathcal{V}_0(0)$, can be mapped to either Group II or III as both of them are correctable. All two-error patterns, i.e., $\mathcal{V}_2(0)$, $\mathcal{V}_2(1)$ and $\mathcal{V}_2(2)$ must be mapped to a separate group to flag a double error.

However, we observe that received sequences corresponding to some of the 2D/I patterns, i.e., $\mathcal{V}_2(0)$, $\mathcal{V}_2(1)$ are correctable, while the only sequences that we cannot correct are $\mathcal{V}_2(2)$. Based on the observation above, we propose a methodology to search for postamble sequences that try to correct received sequences corresponding to curable $\mathcal{V}_2(1)$ and $\mathcal{V}_2(0)$. Fig. 5 shows the updated groups, to which a received word (corresponding to a PP set) will be mapped. We observe that we need three groups: Group I that detects 2D/I errors, and Groups II and III that correct 1I and 1D errors, respectively and $\mathcal{V}_2(2)$, $\mathcal{I}_1(1)$ and $\mathcal{D}_1(1)$ strictly belongs to the three groups, respectively. However, all other PP sets can be mapped to multiple groups. For example, $\mathcal{D}_2(1)$ can be mapped to Group I and be declared as 2D/I detection event or to Group-III, to be corrected for a 1D error in the VT code part for as long as we can distinguish it separately. **Based on this observation, we relax the strict requirement of mapping $\mathcal{D}_2(1)$ to Group-I and gain additional error-correction capability**.

Ideally, we would prefer if the all the received words corresponding to $\mathcal{V}_2(1)$ and $\mathcal{V}_2(0)$ are correctable. However, only a few patterns from this PP may be distinguishable from $\mathcal{V}_2(2)$. As a straightforward objective function, we aim to maximize the number of patterns that can be distinguished from $\mathcal{V}_2(2)$. However, we observe that for a particular postamble pattern, any element of $\mathcal{V}_2(1)$ and $\mathcal{V}_2(0)$ may appear multiple times when different bits are deleted in the postamble part, while passing through a given deletion/insertion channel. Therefore, given a transmitted codeword and a channel, different patterns in a PP set may correspond to different number of received words. Hence, distinguishing one pattern from $\mathcal{V}_2(2)$ may correct more received error patterns than other patterns in the set. We elaborate with the help of an example below:

Example 4: Consider $Z(4, 2, 6)$ *with* $p_1 = p_2 = p_4 = p_6 = 0, p_3 = p_5 = 1$. *For the given code,*

$$\mathcal{D}_2(1) = \{1010, 0010, 0110, 0100\}.$$

Given a codeword is transmitted through a 2-del channel in which one deletion occurs in the VT codeword part, the received pattern will be [1010], *when* p_1 *or* p_2 *are deleted from the postamble. There are a total of four possible received patterns (corresponding to one error in the VT codeword part) when either* p_1 *or* p_2 *is deleted from the postamble part.*

TABLE 1. PP sets and their elements with $\ell = 5$.

	r_1	r_2	r_3
$\mathcal{D}_0(0)$	p_1	p_2	p_3
$\mathcal{D}_1(0)$	p_2	p_3	p_4
	p_1	p_3	p_4
	p_1	p_2	p_4
$\mathcal{D}_1(1)$	p_2	p_3	p_4
$\mathcal{D}_2(0)$	p_3	p_4	p_5
	p_2	p_4	p_5
	p_2	p_3	p_5
	p_1	p_4	p_5
	p_1	p_3	p_5
	p_1	p_2	p_5
$\mathcal{D}_2(1)$	p_3	p_4	p_5
	p_3	p_4	p_5
	p_2	p_4	p_5
	p_2	p_3	p_5
$\mathcal{D}_2(2)$	p_3	p_4	p_5
$\mathcal{I}_1(0)$	p_1	p_1	p_2
	p_1	p_2	p_2
$\mathcal{I}_1(1)$	c_n	p_1	p_2
$\mathcal{I}_2(0)$	The case is not possible as two bits cannot be inserted in postamble part when we are reading $n+3$ bits.		
$\mathcal{I}_2(1)$	c_n	p_1	p_1
$\mathcal{I}_2(2)$	c_{n-1}	c_n	p_1

Therefore, if we were able to distinguish [1010] *from* $\mathcal{V}_2(2)$, *we would be able to correct 8 error patterns. In contrast, if we distinguished* [0010] *from* $\mathcal{V}_2(2)$, *we would be able to correct only 4 error patterns.*

This leads to an objective function that maximizes the occurrence of correctable error patterns. To this end, we first define some quantities, provide some relevant results and then put forward our proposed optimization problem.

Let $\mu_{m,n}(.)$ and $\gamma_{m,n}(.)$ be the functions that yield the multiplicity of a pattern from $\mathcal{D}_m(n)$ and $\mathcal{I}_m(n)$, respectively. The multiplicity of a pattern is defined as the number of times a pattern appears when a given codeword passes through a m-deletion channel and m-insertion channel, where n deletions and insertions occur in the postamble part, respectively. Let $W_m(n)$ and $U_m(n)$ be the size of the set of received words, given a transmitted codeword, from a m-deletion channel and m-insertion channel, where n deletions and insertions occur in the VT codeword part, respectively. Now, we present the proofs of relevant lemmas that are useful to define the proposed coding.

Lemma 2: $W_2(2) = U_2(2) = {}^nC_2$

Proof: Since we are reading $n+\ell-2$ *bits, the entire VT codeword part passes through the read channel, which can introduce error on two locations out of n irrespective of the order of chosen locations. Consequently, given a transmitted codeword, we have* ${}^nC_2$ *combinations of the locations and corresponding received codewords in the 2-deletion and 2-insertion channel cases.* □

Lemma 3: $W_2(1) = U_2(1) = n$.

Proof: In the case of $W_2(1)$ *and* $U_2(1)$, *there is a single bit error in n locations for the VT codeword part and a single bit error in the postamble. So for each sequence of* $\mathcal{V}_2(1)$,

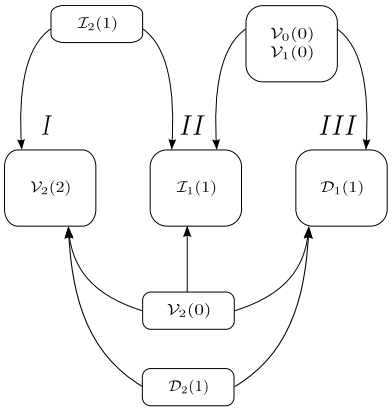

FIGURE 5. Updated groups for optimal postamble patterns. Since, $\mathcal{V}_2(0)$ and $\mathcal{V}_2(1)$ both can be corrected, we preferably want to search postamble patterns that can map as many patterns as possible from 1) $\mathcal{V}_2(0)$ in either Group II or III. 2) $\mathcal{D}_2(1)$ to Group III. 3) $\mathcal{I}_2(1)$ to Group II.

we have n possible error locations in the VT codeword part. □

Lemma 4: $W_2(0) = U_2(0) = 1$.

Proof: The proof of $W_2(0)$ *and* $U_2(0)$ *is similar to proof given in Lemma 3. However, note that there would be no deletion or insertion in the VT codeword part. So for each sequence of* $\mathcal{V}_2(0)$*, we have only* 1 *possible VT pattern.* □

The postamble patterns that fulfill the solution of the search problem defined in (4) will provide a guaranteed P-SECDED code. However, we aim to correct as many received sequences from $\mathcal{V}_2(0)$ and $\mathcal{V}_2(1)$ as possible to achieve more error correction capability. All the received words from $\mathcal{V}_2(0)$ and $\mathcal{V}_2(1)$ are correctable only if we can distinguish them from $\mathcal{V}_2(2)$. We use the proofs given in Lemma 3 and 4 to defined an extra constraint for the search problem to get optimized postamble patterns. Mathematically,

$$\underset{p_1,p_2,\ldots,p_\ell}{\arg\min} \left(\sum_{i=0}^{1} \left(\sum_{\omega\in(\mathcal{I}_2(i)\cap\mathcal{V}_2(2))} \gamma_{2,i}(\omega)U_2(i) + \sum_{\omega\in(\mathcal{D}_2(i)\cap\mathcal{V}_2(2))} \mu_{2,i}(\omega)W_2(i) \right) \right). \tag{5a}$$

$$\text{such that} \quad \mathcal{D}_1(1)\cap\mathcal{I}_1(1) = \emptyset, \tag{5b}$$

$$\mathcal{D}_1(1)\cap\mathcal{I}_2(1) = \emptyset, \tag{5c}$$

$$\mathcal{I}_1(1)\cap\mathcal{D}_2(1) = \emptyset, \tag{5d}$$

$$\mathcal{V}_2(2)\cap\left(\bigcup_{i=0}^{1}\mathcal{V}_1(i)\right)\cap\mathcal{V}_0(0) = \emptyset. \tag{5e}$$

Theorem 2: For a P-SECDED of Eq.5 on 2D/I channel, $\ell \geq 6$.

Proof: The proof is exactly similar to that presented in Theorem 1 for $\ell \geq 4$*. For* $\ell = 5$*, we note that we cannot satisfy Eq. 5e as it will require* $p_1 = p_5$, $p_1 \neq p_3$, $p_1 \neq p_4$ *and* $p3 \neq p_4$*, which is not possible.* □

Example 5: Consider a codeword [1001001010] *from* $Z(4, 2, 6)$ *with* $p_1 = p_2 = p_4 = p_6 = 0$, $p_3 = p_5 = 1$.

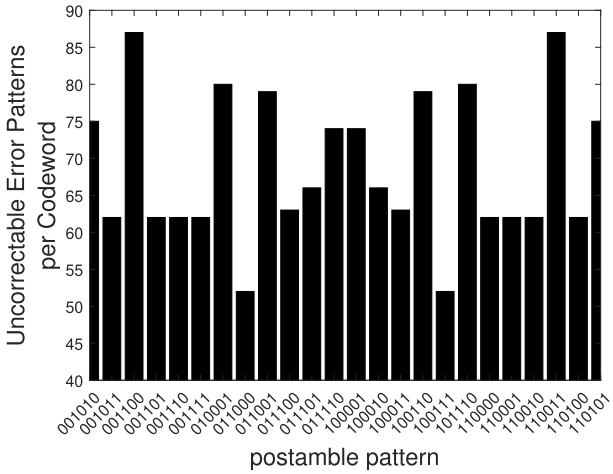

FIGURE 6. The bars represents the sum of uncorrectable error patterns per codeword for $\ell = 6$. It helps in determining which postamble patterns minimizes the summation defined in Eq. 5 the most. The plot shows that the half of the postamble patterns are the complements of the other half of postamble patterns.

For the given code,

$$\mathcal{D}_2(1) = \{1010, 0010, 0110, 0100\},$$
$$\mathcal{V}_2(2) = \{1010, 0000, 0100, 1000, 1100\},$$
$$\mathcal{D}_2(1)\cap\mathcal{V}_2(2) = \{0100, 1010\},$$
$$\mu_{2,1}(x) = \begin{cases} 2, & x = 1010, \\ 1, & otherwise. \end{cases}$$

It is clear that if we receive a pattern from {0100, 1010}*, we cannot correct the received word even though there is only one error in the VT codeword part as we cannot distinguish it from* $\mathcal{V}_2(2)$*. The number of error patterns that cannot be corrected when we receive these postamble patterns is given by*:

$$\sum_{\omega\in(\mathcal{D}_2(1)\cap\mathcal{V}_2(2))} \mu_{2,1}(\omega)W_2(1) = 3n = 12.$$

Hence, for a given transmitted codeword through a 2-deletion channel in which one deletion occurs in the VT codeword part, there will be 12 *error patterns in which we will receive a postamble pattern from* {0100, 1010}*. None of these error patterns will be correctable and we aim to minimize their number. The ideal postamble candidate would achieve the solution to the problem defined in (5).*

Figure 6 shows a bar plot depicting the summation of uncorrectable error patterns per codeword for postamble patterns with $\ell = 6$. Postamble pattern [011000] at index 8 (and its complement and index 17) is the one that minimizes the number of uncorrectable error patterns as shown in the bar plot.

C. P-SECDED DECODER

For our P-SECDED decoder, we employ a binary decision tree for decoding algorithm. We choose the best postamble pattern, i.e. p = [011000], for decision tree walk because

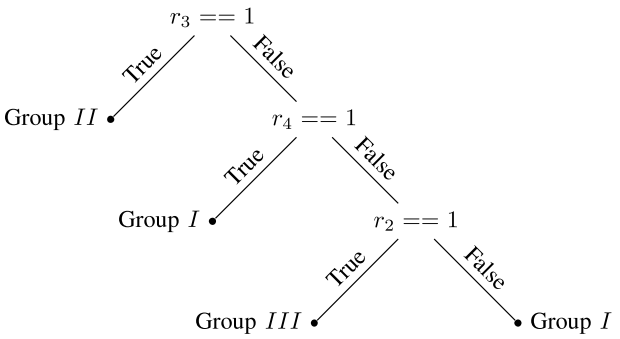

FIGURE 7. Binary Decision Tree of P-SECDED decoder.

we aim to correct maximum double-errors. Table 2 shows a dataset for all possibilities of $\mathcal{I}_1(1)$, $\mathcal{D}_1(1)$, and $\mathcal{V}_2(2)$ sets for the chosen postamble pattern. The vector $[r_1, \ldots, r_4]$ corresponds to $[r_{n+1}, \ldots, r_{n+4}]$ and is known as the feature or predictor variables for a given dataset whereas the Groups column represents the target. Decision tree generator algorithms use entropy and information gain to produce a decision tree. Fig. 7 illustrates a binary decision tree that we obtained using our decision tree algorithm and we illustrate the decoding process with the help of an example.

$$\begin{aligned}\mathcal{I}_1(1) &= \{c_n\ p_1\ p_2\ p_3\}\\ &= \{c\ 0\ 1\ 1\}\\ \mathcal{D}_1(1) &= \{p_2\ p_3\ p_4\ p_5\}\\ &= \{1\ 1\ 0\ 0\}\\ \mathcal{V}_2(2) &= \{p_3\ p_4\ p_5\ p_6,\ c\ c\ p_1\ p_2\}\\ &= \{1\ 0\ 0\ 0,\ c\ c\ 0\ 1\}\end{aligned}$$

Example 6: Consider a codeword [1001011000] *from* $Z(4, 2, 6)$ *with* $p_1 = p_4 = p_5 = p_6 = 0$, $p_2 = p_3 = 1$. *Assuming a single deletion at index* 3, *the received word* $\mathbf{r}$ *would be*:

$$\mathbf{r} = [1\ 0\ 1\ 0\ \overbrace{1}^{r_{n+1}=p_2}\ 1\ 0\ 0\ 0]$$

We observe that $r_{n+3} \neq 1$ *and* $r_{n+4} \neq 1$ *as well. Next we observe that* $r_{n+2} = 1$ *and from Fig. 7, we know that the error pattern belongs to Group III which is for single deletions. Decoding* $[r_1, \ldots, r_{n-1}]$ *using Levenshtein decoder will reliably decode the erroneous codeword.*

D. FAILURE ANALYSIS

We have simulated our scheme on probabilistic deletion channels characterized by the probability of deletion e_d. Such channels delete an input bit with a probability e_d whereas different realizations of this channel are independently and identically distributed (iid). In other words, the channels are characterized by a Bernoulli(e_d) distribution.

Suppose, a $n+\ell$-bit long codeword of the proposed scheme is input to a deletion channel with deletion probability e_d.

TABLE 2. Dataset for $\mathcal{V}_2(2)$, $\mathcal{I}_1(1)$, $\mathcal{D}_1(1)$ sets, respectively, with predictor variables and target.

r_1	r_2	r_3	r_4	Group
1	0	0	0	I
0	0	0	1	
0	1	0	1	
1	0	0	1	
1	1	0	1	
0	0	1	1	II
1	0	1	1	
1	1	0	0	III

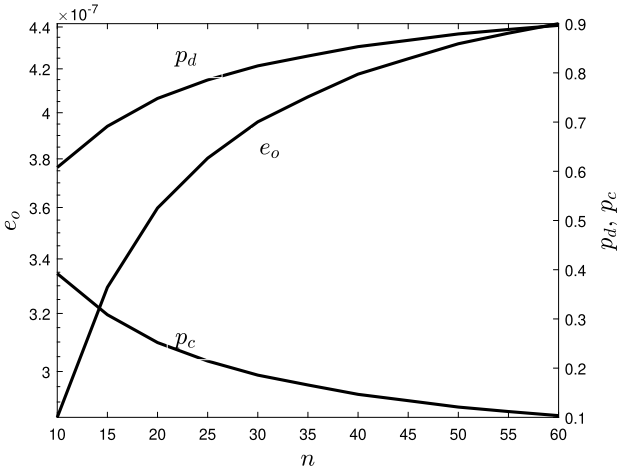

FIGURE 8. e_o increases with increasing n while keeping $e_i = 10^{-3}$ but stays well below 10^{-6}. It is also clear that the proposed scheme provide error correction for two errors at lower values of n compared to higher n.

We first define the probability of error at the output of such channels and input to the decoder as

$$e_i = 1 - (1 - e_d)^{n+\ell},$$

where we have used the fact that the probability that no error occurs at the output of such a channel is $(1 - e_d)^{n+\ell}$.

We have calculated the probability of error at the output of the decoder e_o by transmitting 10^5 codewords through the channel and averaging the probability of error. In addition to e_o, we have also calculated the detection and correction probabilities of two deletions p_d and p_c, respectively by only considering two errors in the received codewords.

Fig. 8 shows the error correction performance of the proposed scheme on a deletion channel with $e_i = 10^{-3}$. The output error probability is well below 10^{-6} for a wide range of block lengths n. We have also shown the two error correction and detection performance of the scheme by plotting p_d and p_c. It is clear that the scheme offers more two error correction at lower block lengths n than at higher ones.

Fig. 9 shows the error correction performance of the proposed scheme on different deletion channels for $n = 32$. It is clear that for a given n, e_o is linearly related to e_i in log-domain. We p_d and p_c stays almost constant for the entire range of e_i values showing that the two-error correction performance of the scheme largely depends on the block length n.

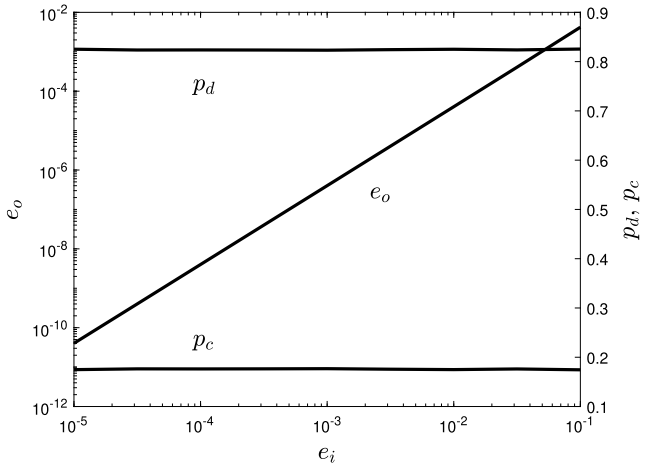

FIGURE 9. e_o increases with increasing $e_i n$ for $n = 32$. It is also clear that the proposed scheme provides the same two error correction and detection performance on the entire range of simulated $e_i n$.

E. IMPLEMENTATION OVERHEADS

To evaluate the implementation overheads of P-SECDED, we revisit Figure 2. At write, the data word is encoded with VT codeword and then postamble is added. At read, the postamble first helps with the classification presented in Figure 7 and Tables 1, 2. With deletions and insertions already decided, VT decoder (Levenshtein decoder) then helps to recover the correct data word. Uncorrected double errors are appropriately flagged.

In this work, we use a Radix-2 VT encoder [17] that significantly improves the area, power and energy consumption of earlier implementations, including SSC [15] and Greenflag [14] (and Foosball [16]). We add the implementation overheads of postamble decoding. Then we compare the implementation metrics of P-SECDED with the previous schemes in the literature, over a range of racetrack length. In our analysis, datapath elements including adders and counters were modeled in Verilog HDL whereas large shift registers were modeled with dual-port SRAM arrays (with 16-bit wordlines). Synopsys Design Compiler was leveraged to produce post-synthesis reports of the former, at a 45nm technology node, while CACTI [19], an open-source simulator, generated SRAM area, access time, static power and access energies. For dynamic energy consumption, the activity factors were estimated through Monte-Carlo simulations. Compiled results for VRM length N, ranging from 16-bit to 256-bit, in 16-bit increments, are plotted in Fig. 10, in a manner similar to [17].

Figure 10(a) plots the area overhead of Greenflag (and Foosball), SSC and P-SECDED in terms of standard cell (gate) utilization. As data block size increases, the silicon overhead of P-SECDED scales gracefully. Because postemble decoding is trivially low cost, the overhead is primarily determined by the implementation of VT encoder and decoder. Therefore, we argue that VT codes offer excellent scalability with respect to area overhead. However, datapath elements consume energy and may increase the latency when placed in the critical path, therefore, minimizing the area overhead is beneficial. GreenFlag does not optimize the VT encoder and is therefore, area-efficient only when $N = 2^n$, manifested as notches in the figure.

The processing time is computed by multiplying the number of iterations of the syndrome generation kernel with the delay of its critical path. The number of iterations are N for SSC and GreenFlag, $N/2$ for P-SECDED due to its Radix-2 kernel. In addition, $N/4$ iterations on average are incurred in GreenFlag in lieu of $\mod N + 1$ adjustments when N is not 2^n, with a larger critical path delay. Fig. 10(b) plots the syndrome generation time (processing time in nanoseconds) vs. N. The critical path does not scales up much with N. However, the number of iterations dominate the slope and therefore, Radix-2 VT encoder utilized in P-SECDED shows a clear advantage in terms of performance and its latency is more than 50% less than other architectures.

The sum of static and dynamic energy is the total energy consumption. Static energy is calculated by multiplying the processing time with the total static power of the datapath, whereas the later is computed from the switching activities and the dynamic energy consumed by each datapath element. As a result, the processing time strongly influences static energy consumption, whereas, the dynamic energy of the process is determined by the size of the datapath elements and their access patterns, which were obtained with Monte Carlo simulations and combined with access rates to provide switching activity factors. Total energy consumption is plotted in Fig. 10(c). With Radix-2 optimization, P-SECDED achieves energy efficiency as it retains a low activity profile and its processing time is half of what is achievable with competitive architectures. Without such optimizations, GreenFlag (and Foosball) bears high costs when $N \neq 2^n$, where Radix-2 P-SECDED scales more gracefully with N. Because the slope of total energy consumption has processing time as a factor, and that, in turn, is a function of the iteration count, an architecture that has lower complexity, lower activity, and fewer iterations is economic with a smaller energy footprint.

IV. LITERATURE REVIEW AND DISCUSSIONS

A. SHIFT PROCESS IMPROVEMENT

Approaches have been proposed to increase the efficiency of the shifting process in order to reduce the probability of errors. In [11], authors devised a technique called subthreshold shift (STS) to perform a more reliable shift in two stages. STS aimed to eliminate the stop-in-middle errors. The detection of such errors, however, has significant technological challenges. In [20], authors proposed a transverse reading mechanism to reduce the impact of shifting process. On the other hand, many recent works in the context of skyrmion-based racetrack memories are focused on the reliable movement of skyrmions across their docking sites. These include voltage-induced movements [21], etching notches across the racetrack [22], and steps across the racetrack [23]. Nevertheless, a recent study on the novel skyrmion-based racetracks reports that the stochastic depinning process of any

FIGURE 10. (a) Area (in number of 1x Inverter gates), (b) Computation latency in nanoseconds and, (c) Total energy consumption (Static + Dynamic) in nanojoules. Results compiled by leveraging post-synthesis reports in 45nm technology.

racetrack-like device requires an extremely narrow depinning time distribution smaller than 6% of the current pulse length to reach practical bit error rates [7]. Therefore, forward error correction is inevitable for racetrack memories.

B. ERROR CORRECTION WITH MULTIPLE READ HEADS

Many researches have focused on architecting Random Access Memories around RM which are sequential in nature. Multiple read heads at distributed along the racetrack can simultaneously read multiple fragments of stored data words, hence increasing the throughput [24]. However, as all domains move simultaneously, these individual fragments can be considered as individual racetracks, joined head to tail as a linked list, whose each node is a smaller RM with a single head. Alternatively, when whole data stream is swept across the racetrack, all heads get opportunity to sample every data bit and this sample redundancy can be exploited for error correction. The earliest of such attempts was p-ECC [11] that appended a sequence of guard bits to the data word stored on RM. A separate read port could read guard bits to detect whether a deletion has happened or not. However, as the selection of guard bits was rather empirical, without a theoretical backing, it was imperfect, as the deletions in the guard bits themselves sabotaged the whole ECC. Therefore, np-ECC [25] focused to fortify these guard bits, by using adjacent read ports. It is pertinent to note that multiple read heads (ports) are not providing read efficiency in these schemes. Rather, these are providing read redundancy for error correction.

As analytical backing was still missing, [12] first produced a b-burst-deletion/insertion-correcting code that was able to correct a burst of deletions/insertions of any b consecutive bits and found a new asymptotically optimal code redundancy in the context of racetracks, that was much better than previously established worst case. In [13], authors theoretically demystified the error correction in the presence of deletion and insertions and deduced the bounds on the numbers of read heads specifically required for error correction, and also the quantum distance between them. These works established the theoretical foundations on which the earlier works in [11] and [25] can be analyzed. However, for high density vertical racetrack memories, multiple heads cannot be arranged, as the U-shaped racetrack touches the silicon substrate on a single point (Figure 1). There was a need to mathematically found the postamble based error correction in the context of single-head memories, in order to offer greater flexibility for technology adopters and architects. Our P-SECDED technique is mathematically founded and require no additional heads. Moreover, it is capable to correct many double errors, yielding a higher performance in the presence of errors.

C. ERROR CORRECTION WITH FEC

Conventional Forward error correction schemes are also explored for positional errors, especially in the domain of DNA storage [26]. Recent examples are Low-density parity-check codes (LDPC) [9] and polar codes [10] which are applied to insertion and deletion channels like racetrack. In [9], authors present a new coding scheme that can effectively decode corrupted data with non-iterative detection. They utilize an inner 2-D marker code, which is specialized for PEs to mitigate the effect of insertion and deletion errors, and an outer irregular LDPC code. However, it suffers from code rate loss with a low-complexity decoding algorithm, that strictly limits the minimum code block size in tens of kilo-bits. Below this block size, the amortized implementation overheads and code rate become restrictive to their application to on-chip memories.

In [10], authors build two typical error-correction models for the skyrmion memory and a polar coding scheme for error-correction is applied, whereas the proposed successive-cancellation decoding algorithm is used to correct the deletion errors. Authors propose an efficient decoding algorithm to improve the decoding performance and reduce the decoding complexity of polar codes in insertion deletion channels. With a tighter bound of the number of segmented codewords, the decoding performance is improved and the decoding complexity is reduced. However, the drawbacks are same as in the case of LDPC, as the hardware complexity of successive cancellation decoder is often restrictive to latency sensitive applications. The decoder appears in the read path which is critical in main memory applications and where a low cost mechanism is imperative.

D. ERROR CORRECTION WITH VT-CODES

Lately, it was established that VT-codes which were specifically proposed for deletion channels, can be applied to racetrack memories. In [8], authors first applied these codes to correct single deletions and insertions in racetrack memories.

To detect and possibly correct double errors in a single racetrack, they further proposed to use racetrack arrays and to interleave data words, protected with parity (conventional block codes). However, it is difficult to guaranty that there would be a single racetrack with multiple errors at a time. This is demonstrated in Fig. 2 where errors in n racetracks require upto n-bit error correction.

Compared to using multiple read heads, VT coding is a non intrusive approach and an architectural solution, as no technological modifications are required. Moreover, many efficient implementations of VT-code exist in the literature [15], [16], [17]. The proposed method adds only six postamble bits to add DED in P-SECDED and can leverage existing high performance VT-coding architectures. These features make the proposed P-SECDED superior than previously proposed schemes.

V. CONCLUSION

Last section highlights many recent works that attempt to mitigate positional errors in horizontal and vertical racetrack memories. We show that the proposed P-SECDED is a highly-efficient and robust method to correct a single deletion or insertion error and to detect upto two errors in a single read of the single-headed racetrack, that despite its significance in ultra-dense storage applications, finds lesser attention in the literature. P-SECDED is backed by an elaborate mathematical model to classify positional errors, followed by establishing the constraints on the proposed postambles. By rigorous analysis, we discover a postamble that can maximally correct double errors and successfully flags the remaining. We also demonstrate a simple decoding algorithm for the proposed P-SECDED. Our analytical approach provides a template to further research into double error correction, triple error detection (DECTED) and beyond, specifically in the context of racetrack memories.

REFERENCES

[1] Y. Cai, J. Llorca, A. M. Tulino, and A. F. Molisch, "Compute- and data-intensive networks: The key to the metaverse," in *Proc. 1st Int. Conf. 6G Netw. (6GNet)*, Jul. 2022, pp. 1–8.

[2] R. Bläsing, A. A. Khan, P. C. Filippou, C. Garg, F. Hameed, J. Castrillon, and S. S. P. Parkin, "Magnetic racetrack memory: From physics to the cusp of applications within a decade," *Proc. IEEE*, vol. 108, no. 8, pp. 1303–1321, Aug. 2020.

[3] Z. Sun, X. Bi, W. Wu, S. Yoo, and H. Li, "Array organization and data management exploration in racetrack memory," *IEEE Trans. Comput.*, vol. 65, no. 4, pp. 1041–1054, Apr. 2016.

[4] C. Zhang, G. Sun, W. Zhang, F. Mi, H. Li, and W. Zhao, "Quantitative modeling of racetrack memory, a tradeoff among area, performance, and power," in *Proc. 20th Asia South Pacific Design Autom. Conf.*, Jan. 2015, pp. 100–105.

[5] S. Parkin, "Racetrack memory: A high capacity, high performance, non-volatile spintronic memory," in *Proc. IEEE Int. Memory Workshop (IMW)*, May 2022, pp. 1–4.

[6] S. Parkin, "Racetrack memory: A storage class memory based on current controlled magnetic domain wall motion," in *Proc. Device Res. Conf.*, Jun. 2009, pp. 3–6.

[7] D. Suess, C. Vogler, F. Bruckner, P. Heistracher, F. Slanovc, and C. Abert, "Spin torque efficiency and analytic error rate estimates of skyrmion racetrack memory," *Sci. Rep.*, vol. 9, no. 1, Mar. 2019, Art. no. 4827.

[8] A. Vahid, G. Mappouras, D. J. Sorin, and R. Calderbank, "Correcting two deletions and insertions in racetrack memory," 2017, *arXiv:1701.06478*.

[9] R. Shibata, G. Hosoya, and H. Yashima, "Protograph-based LDPC coded system for position errors in racetrack memories," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E102.A, no. 10, pp. 1340–1350, 2019.

[10] H. Sun, R. Liu, K. Tian, T. Zou, and B. Feng, "Deletion error correction based on polar codes in skyrmion racetrack memory," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Mar. 2021, pp. 1–6.

[11] C. Zhang, G. Sun, X. Zhang, W. Zhang, W. Zhao, T. Wang, Y. Liang, Y. Liu, Y. Wang, and J. Shu, "Hi-Fi playback: Tolerating position errors in shift operations of racetrack memory," in *Proc. 42nd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2015, pp. 694–706.

[12] C. Schoeny, A. Wachter-Zeh, R. Gabrys, and E. Yaakobi, "Codes correcting a burst of deletions or insertions," *IEEE Trans. Inf. Theory*, vol. 63, no. 4, pp. 1971–1985, Apr. 2017.

[13] Y. M. Chee, H. M. Kiah, A. Vardy, V. K. Vu, and E. Yaakobi, "Coding for racetrack memories," *IEEE Trans. Inf. Theory*, vol. 64, no. 11, pp. 7094–7112, Nov. 2018.

[14] G. Mappouras, A. Vahid, R. Calderbank, and D. J. Sorin, "GreenFlag: Protecting 3D-racetrack memory from shift errors," in *Proc. 49th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2019, pp. 1–12.

[15] T. Mahmood, U. U. Fayyaz, and A.-U.-R. Makhdoom, "An efficient syndrome calculator for Varshamov–Tenengolts codes in vertical racetrack memory," *IEEE Magn. Lett.*, vol. 10, pp. 1–5, 2019.

[16] S. Archer, G. Mappouras, R. Calderbank, and D. Sorin, "Foosball coding: Correcting shift errors and bit flip errors in 3D racetrack memory," in *Proc. 50th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2020, pp. 331–342.

[17] U. U. Fayyaz and T. Mahmood, "On the radix-2 syndrome generators for Varshamov–Tenengolts-coded vertical racetrack memories," *IEEE Trans. Magn.*, vol. 57, no. 12, pp. 1–5, Dec. 2021.

[18] K. A. S. Abdel-Ghaffar and H. C. Ferreira, "Systematic encoding of the Varshamov–Tenengol'ts codes and the Constantin–Rao codes," *IEEE Trans. Inf. Theory*, vol. 44, no. 1, pp. 340–345, Jan. 1998.

[19] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "Cacti 6.0: A tool to model large caches," *HP Laboratories*, vol. 27, p. 28, Apr. 2009.

[20] S. Ollivier. (Jan. 2019). *Derived ECC for Protection of on-Demand Data*. [Online]. Available: http://d-scholarship.pitt.edu/35523/

[21] W. Kang, Y. Huang, C. Zheng, W. Lv, N. Lei, Y. Zhang, X. Zhang, Y. Zhou, and W. Zhao, "Voltage controlled magnetic skyrmion motion for racetrack memory," *Sci. Rep.*, vol. 6, Mar. 2016, Art. no. 23164.

[22] M. G. Morshed, H. Vakili, and A. W. Ghosh, "Positional stability of skyrmions in a racetrack memory with notched geometry," *Phys. Rev. Appl.*, vol. 17, no. 6, Jun. 2022, Art. no. 064019. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevApplied.17.064019

[23] W. Al Saidi and R. Sbiaa, "Stabilizing magnetic skyrmions in constricted nanowires," *Sci. Rep.*, vol. 12, no. 1, Jun. 2022, Art. no. 10141.

[24] R. Venkatesan, V. Kozhikkottu, C. Augustine, A. Raychowdhury, K. Roy, and A. Raghunathan, "TapeCache: A high density, energy efficient cache based on domain wall memory," in *Proc. ISLPED*, Jul. 2012, pp. 185–190.

[25] X. Wang, C. Zhang, X. Zhang, and G. Sun, "np-ECC: Nonadjacent position error correction code for racetrack memory," in *Proc. IEEE/ACM Int. Symp. Nanosc. Archit. (NANOARCH)*, Jul. 2016, pp. 23–24.

[26] M. Blawat, K. Gaedke, I. Hütter, X.-M. Chen, B. Turczyk, S. Inverso, B. W. Pruitt, and G. M. Church, "Forward error correction for DNA data storage," *Proc. Comput. Sci.*, vol. 80, pp. 1011–1022, Jan. 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050916308742

AWAIS SAEED received the B.Sc. degree in electrical engineering from COMSATS University Islamabad, Lahore, in 2019. He is currently pursuing the M.Sc. degree in electronics and communication with the University of Engineering and Technology, Lahore. In 2020, he joined a private enterprise as a Technical Engineer, where his core responsibility is research and development in the domains of electronics and wireless communication. His research interests include visible light communication (VLC), MIMO-VLC, computer networks, channel coding theory, and in particular, coding for racetrack memories.

UBAID U. FAYYAZ received the B.S. degree in electrical engineering from the University of Engineering and Technology, Lahore, Pakistan, in 2005, and the M.S. and Ph.D. degrees in electrical engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2013 and 2016, respectively. From 2006 to 2009, he worked with the Center for Advance Research in Engineering (CARE), Islamabad, Pakistan, where he was responsible for the algorithm design and FPGA-based implementations of communication systems. He is currently working as an Assistant Professor with the Electrical Engineering Department, University of Engineering and Technology, Lahore. His current research interests include coding theory, information theory, and signal processing.

AHSAN TAHIR received the M.S. degree in computer engineering from the University of Michigan, Ann Arbor, MI, USA, and the Ph.D. degree from Glasgow Caledonian University, Glasgow, U.K. He is currently working as an experienced Researcher and an Associate Professor with the Department of Electrical Engineering, University of Engineering and Technology, Lahore, Pakistan. He is also an Honorary Guest Researcher with the James Watt School of Engineering, University of Glasgow, U.K. He has research/teaching experience with prestigious institutes, including the University of Edinburgh and the University of Glasgow. His research interests include machine learning/deep learning for healthcare applications and natural language processing. He also develops re-configurable and ASIC hardware accelerators for machine and deep learning applications.

SEOKIN HONG (Member, IEEE) received the Ph.D. degree in computer science from the Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 2015. From 2015 to 2017, he was a Senior Engineer with Samsung Electronics, where he was involved in a project that developed the 3D-stacked memory. In 2017, he moved to the IBM Thomas J. Watson Research Center, where he worked on secure processor architectures and emerging memory/storage systems. He is currently an Assistant Professor with Sungkyunkwan University, South Korea. His current research interests include the design of low-power, reliable, and high-performance processor architectures and memory systems. He received the Best Paper Awards from the International Conference on Computer Design (ICCD), in 2010, and the Design Automation and Test in Europe (DATE), in 2013.

TAYYEB MAHMOOD received the Ph.D. degree in information and communication engineering from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea. He is currently working as an Assistant Professor with the Department of Electrical Engineering, University of Engineering and Technology, Lahore, Pakistan. His research interests include low-power computing, the Internet of Things, and embedded systems.