Low-Latency Nested Decoding for Short Generalized Integrated Interleaved BCH Codes

Zhenshan Xie, Yok Jye Tang, and Xinmiao Zhang

***Abstract*—Generalized integrated interleaved (GII) codes nest short BCH sub-codewords to form more powerful BCH codewords. They can potentially achieve hyper-speed decoding with excellent error-correction capability. In particular, short GII-BCH codes are among the best candidates for the new fast storage class memories (SCMs). Miscorrections severely degrade the performance of short GII-BCH codes. Although they were effectively mitigated in previous designs, the involved repeated Chien search and higher-order syndrome computation cause long latency. This paper proposes efficient and low-latency nested decoding schemes for short GII-BCH codes. A strategy is developed to select sub-words for further nested decoding to mitigate miscorrections by keeping track of the error locator polynomials, instead of waiting for the lengthy Chien search. Formulas are also derived to estimate the effects on the error-correcting performance. Besides, a low-complexity linear feedback shift register architecture is developed to accelerate the higher-order nested syndrome computation. For an example GII-BCH code targeting at SCMs, the proposed design reduces the worst-case nested decoding latency by 26% with 8.5% area overhead and negligible performance loss compared to prior methods.**

***Index Terms*—BCH codes, Error-correcting decoding, Generalized integrated interleaved codes, Storage class memories.**

I. Introduction

Generalized integrated interleaved (GII) codes can nest BCH sub-codewords to form more powerful BCH codewords [1], [2]. The GII decoding starts with the decoding of individual sub-words, which can achieve very high throughput. Only when there are extra errors exceeding the correction capability of individual sub-codewords, the nested decoding stage utilizing the stronger nested codewords is activated. Compared with traditional BCH codes that achieve similar decoding throughput, GII codes have much better error-correcting performance.

The new storage class memories (SCMs) have much shorter sensing latency than Flash memories. They may bring paradigm shifts to many systems, such as computer memory architecture, high-performance computing, and big data analytics. The hyper throughput and excellent correction capability make GII codes among the best candidates to achieve the speed potential of SCMs. Low-density parity-check codes and concatenated BCH codes cannot achieve such high throughput without overwhelming complexity. For SCMs, relative short codeword length, *e.g.* several thousand bits, and high code rate, *e.g.* 90%, are required. The best GII codes with these parameters have small correction capability in the sub-codewords, such as t_0=3. In this case, a corrupted sub-codeword is more likely to be decoded as another sub-codeword. This is referred to as miscorrection. It degrades the frame error rates (FERs) of short GII-BCH codes by orders of magnitude [3].

The schemes in [4], [5] can only correct very few error patterns and the performance improvement is limited when t_0 is small. Several miscorrection-mitigating schemes were developed in [3] to close the performance gap. However, they make the worst-case decoding latency much longer. First, Chien search is needed to select the sub-words for further nested decoding. Second, higher-order syndromes are computed to detect miscorrections. Both computations are repeated over nested decoding rounds, leading to long latency. Although the overall GII decoding throughput is mainly decided by the sub-word decoding, it is essential to reduce the worst-case nested decoding latency in order to improve the quality of service.

This paper proposes two schemes to substantially reduce the worst-case nested decoding latency for short GII-BCH codes with low area overhead and negligible performance loss. First, a strategy is developed to select sub-words for further nested decoding based on how the error locator polynomials change over the previous decoding round instead of waiting for the Chien search results. By analyzing the error patterns leading to extra GII decoding failure, formulas are also derived to estimate the corresponding FER. The proposed strategy only leads to negligible FER degradation compared to those in [3]. Secondly, this paper proposes a low-complexity linear feedback shift register (LFSR) architecture to halve the higher-order syndrome computation latency. The scheduling of the computations in the GII decoding is also optimized to take advantage of the proposed schemes. For an example GII-BCH code considered for SCMs, our proposed design reduces the worst-case nested decoding latency by 26% with only 8.5% area overhead compared to prior designs.

II. GII-BCH Decoding and Miscorrections

The definition of a GII-BCH $[m,v]$ code involves $v+1$ BCH codes $\mathcal{C}_v \subseteq \mathcal{C}_{v-1} \subseteq \cdots \subseteq \mathcal{C}_1 \subset \mathcal{C}_0$. These BCH codes over $GF(2^q)$ have error correction capabilities $t_v \geq t_{v-1} \geq \cdots \geq t_1 > t_0$. A GII-BCH $[m,v]$ code can be formally defined as [2]

$$\mathcal{C} \triangleq \Big\{ [c_0(x), c_1(x), \cdots, c_{m-1}(x)] : c_i(x) \in \mathcal{C}_0,$$
$$\tilde{c}_l(x) = \sum_{i=0}^{m-1} \alpha^{il}(x) c_i(x) \in \mathcal{C}_{v-l}, 0 \leq l < v \Big\}, \quad (1)$$

where α is a primitive element of $GF(2^q)$ and $\alpha^{il}(x)$ is the polynomial form of the standard basis representation of α^{il}. In

This material is based upon work supported by the National Science Foundation under Award No. 2011785. The authors are with the Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH 43210 USA.

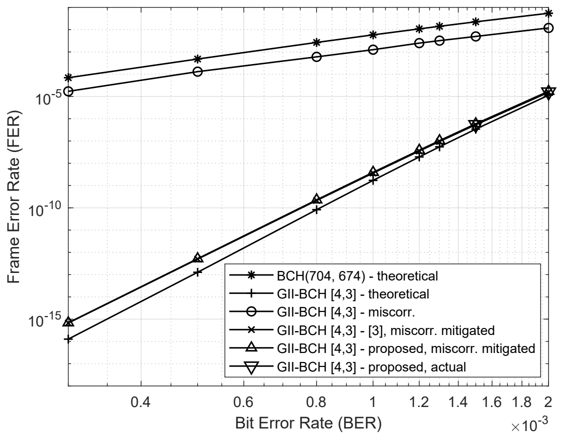

Fig. 1. FERs of GII-BCH [4,3] decoding over binary symmetric channel.

(1), $c_i(x)$ and $\tilde{c}_l(x)$ are a length-n sub-codeword and a nested codeword, respectively.

GII-BCH decoding has two stages. The first stage is the traditional t_0-error-correcting BCH decoding for each received sub-word $y_i(x)$. $2t_0$ syndromes are calculated as $S_j^{(i)}= y_i(\alpha^{j+1})$ $(0 \leq j < 2t_0)$. If any syndrome is not zero, a key equation solver (KES), such as the Berlekamp-Massey algorithm [6], computes the error locator polynomial $\Lambda^{(r)}(x)$ for $i = 1, 2, \cdots, 2t_0$ iteratively. The inverses of the roots of $\Lambda^{(2t_0)}(x)$ indicate the error locations. Let $d^{(r)}$ and $\sigma^{(r)}$ be the degree and number of roots of $\Lambda^{(r)}(x)$, respectively. If $d^{(2t_0)} \neq \sigma^{(2t_0)}$, decoding failure is declared. Let f be the XOR result of all the bits in the received sub-word. If $d^{(2t_0)}=\sigma^{(2t_0)}$ but $d^{(2t_0)}>t_0$ or $d^{(2t_0)} \bmod 2 \neq f$, the sub-word is apparently miscorrected. Miscorrections may also happen otherwise [3].

The second stage is the nested decoding for sub-words with more than t_0 errors. Denote the indices of such sub-words by $i_0, i_1, \cdots, i_{b-1}$ $(b \leq v)$. $2t$ syndromes are needed to correct t errors. Higher-order nested syndromes can be computed as $\tilde{S}_j^{(l)}=\tilde{y}_l(\alpha^{j+1})=\sum_{i=0}^{m-1} \alpha^{il(j+1)} y_i(\alpha^{j+1})$ $(0 \leq l < b$, $2t_0 \leq j < 2t_1)$, since all nested codewords are at least t_1-error-correcting. From (1), they can be converted to higher-order syndromes for those b uncorrected sub-words as

$$\left[S_j^{(i_0)}, S_j^{(i_1)}, \cdots, S_j^{(i_{b-1})}\right]^T = A^{-1}\left[\tilde{S}_j^{(0)}, \tilde{S}_j^{(1)}, \cdots, \tilde{S}_j^{(b-1)}\right]^T,$$

where the matrix entry $A_{k,w}$ is equal to $\alpha^{i_w k(j+1)}$ [2]. After that, $\Lambda^{(2t_1)}(x)$ of each uncorrected sub-word is computed to correct up to t_1 errors. If there are b' sub-words with more than t_1 errors $(b' \leq v-1)$, $2(t_2-t_1)$ higher-order syndromes are derived to correct more errors in a similar way. The nested decoding is repeated for up to v rounds.

Sort the number of errors in the received sub-words as $\tau_0 \geq \tau_1 \geq \cdots \geq \tau_{m-1}$. If $\tau_l \leq t_{v-l}$ for $0 \leq l \leq v$, the errors are correctable by the GII code. SCMs require short codes with high code rate, such as that using around 256 parity bits to protect 2560 data bits. For these parameters, the GII-BCH [4,3] code over $GF(2^{10})$ with n=(2560+256)/4=704 and $[t_0, t_1, t_2, t_3]=[3, 5, 6, 11]$ achieves the best trade-off on error-correcting performance and decoding complexity [3]. Fig. 1 shows simulation results over binary symmetric channel with various input bit error rates (BERs). This GII code can achieve orders of magnitude lower FER compared to the (704, 674) BCH sub-codeword that has similar decoding throughput. Although the performance of GII codes with small t_0 suffers severely from miscorrections, the mitigation schemes in [3] can almost fully eliminate the FER gap as shown in Fig. 1.

For the example GII-BCH [4,3] code, the worst-case nested decoding latency using the best available designs [3], [7]–[9] is shown in Fig. 2. To simplify the labels, it is assumed that $y_{v-i}(x)$ is corrected in nested decoding round i and the first-stage sub-word decoding is considered as nested decoding round 0. The parallelisms of the computation units are chosen to improve the hardware utilization efficiency and the clock cycle numbers are listed in the parentheses. The design in [8] is adopted to substantially reduce the computation latency of the original nested syndromes by using short remainder polynomials calculated in parallel with the sub-word decoding. If a sub-word, say $y_k(x)$, gets corrected in nested decoding round i, then the original higher-order nested syndromes should be updated as $\tilde{S}_j^{\prime(l)}=\tilde{S}_j^{(l)}-\alpha^{lk(j+1)}S_j^{(k)}$ $(j \geq 2t_i)$, where the higher-order syndromes for the corrected sub-word are calculated as [2]

$$S_j^{(k)} = S_{j-1}^{(k)}\Lambda_0^{(2t_i)} + S_{j-2}^{(k)}\Lambda_1^{(2t_i)} + \cdots + S_{j-1-d^{(2t_i)}}^{(k)}\Lambda_{d^{(2t_i)}}^{(2t_i)}. \quad (2)$$

The nested KES architecture in [7] incorporates $2t$ higher-order syndromes in $2t$ clock cycles and it is shared to compute $S_j^{(k)}$ in Fig. 2. For the example code, miscorrections can be reliably detected by checking if $[\sigma_1, \sigma_2, \sigma_3]=[4, 3, 3]$ higher-order nested syndromes based on the current decoding result are zero after decoding round 0, 1, and 2, respectively [3]. σ_i and $t_{i+1}-t_i$ decide the number of higher-order sub-word syndromes to compute using (2) and accordingly the corresponding latency in Fig. 2. In the last decoding round, since there is no miscorrection detection, higher-order syndromes do not need to be computed for the uncorrected sub-word. To increase the hardware utilization efficiency, the scalable Chien search architecture in [9] takes more clock cycles to find the roots of a longer error locator polynomial.

III. Low-Latency Nested Decoding for Short GII-BCH Codes

In this section, low-complexity schemes are developed to reduce the worst-case nested decoding latency for short GII-BCH codes. The first scheme selects sub-words for further decoding based on how the error locator polynomial changes over the previous decoding round instead of waiting for the Chien search results. Formulas are also derived to analyze the achievable FER. Secondly, a low-complexity LFSR architecture is developed to speed up the computation of higher-order syndromes instead of sharing the nested KES architecture.

A. Sub-Word Selection Using Error Locator Polynomials

How the error locator polynomial, $\Lambda(x)$, changes over the previous decoding round tells extra information about whether the sub-word still has extra errors. This helps to decide if a sub-word needs further nested decoding without Chien search.

At the end of the KES of nested decoding round 1, a sub-word falls into one of the following categories:
i). $\Lambda^{(2t_1)}(x) \neq \Lambda^{(2t_0)}(x)$ & $(d^{(2t_1)}>t_1$ or $d^{(2t_1)} \bmod 2 \neq f)$;
ii). $\Lambda^{(2t_1)}(x)=\Lambda^{(2t_0)}(x)$ & $(d^{(2t_0)} \neq \sigma^{(2t_0)}$ or $d^{(2t_0)}>t_0$ or

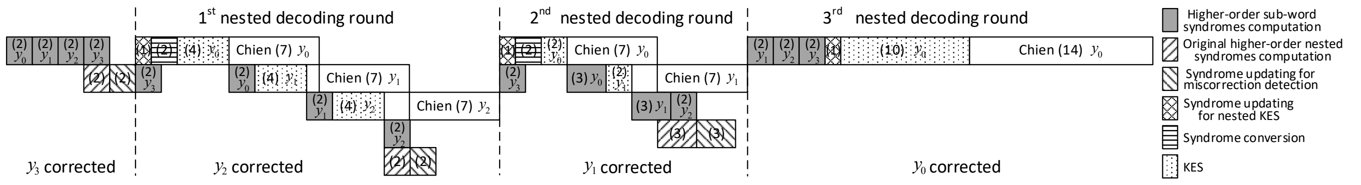
1st nested decoding round
2nd nested decoding round
3rd nested decoding round
Chien (7) y_0
Chien (7) y_1
Chien (7) y_2
Chien (14) y_0
y_3 corrected
y_2 corrected
y_1 corrected
y_0 corrected
Higher-order sub-word syndromes computation
Original higher-order nested syndromes computation
Syndrome updating for miscorrection detection
Syndrome updating for nested KES
Syndrome conversion
KES

Fig. 2. Worst-case nested decoding latency for GII-BCH [4,3] code with $[t_0, t_1, t_2, t_3]$=$[3, 5, 6, 11]$ using prior designs.

$d^{(2t_0)} \bmod 2 \neq f$);
iii). $\Lambda^{(2t_1)}(x) \neq \Lambda^{(2t_0)}(x)$ & excluding type i);
iv). $\Lambda^{(2t_1)}(x) = \Lambda^{(2t_0)}(x)$ & excluding type ii).
For case i), the decoding apparently fails in the first nested round. In case ii), since $\Lambda^{(2t_0)}(x)$ from the sub-word decoding is not the correct error locator polynomial, neither is $\Lambda^{(2t_1)}(x)$. Hence, a sub-word of type i) or ii) definitely needs further nested decoding. Up to $v-1$ sub-words are correctable by the second nested decoding round. If this quota has not been filled by type i) and ii) sub-words, sub-words of type iii) are selected before those of type iv) are considered since they have more than t_1 errors with higher probability as will be proved in the following paragraphs. If there are multiple sub-words of the same type, they are picked in the order of decreasing $\deg(\Lambda(x))$ since a sub-word with higher $\deg(\Lambda(x))$ is more likely to be miscorrected [3].

A sub-word of type iii) must have more than t_0 errors. Otherwise, $\Lambda^{(2t_0)}(x)$ is already the correct error locator polynomial and $\Lambda^{(2t_1)}(x)$ would be the same as $\Lambda^{(2t_0)}(x)$. Therefore, the probability that a type-iii) sub-word has more than t_1 errors can be computed as

$$P_1 = \frac{E_1}{D_1} = \frac{\sum_{w=t_1+1}^{t_v} \phi_w K_w}{\sum_{w=t_0+1}^{t_1} \phi_w + \sum_{w=t_1+1}^{t_v} \phi_w K_w}, \quad (3)$$

where $\phi_w = \binom{n}{w} p_b^w (1-p_b)^{n-w}$ is the probability of an n-bit sub-codeword corrupted with w errors at input BER p_b. K_w is the probability that a w-error-corrupted sub-word belongs to type iii). D_1 and E_1 are the probabilities of having a type iii) sub-word and such a sub-word with more than t_1 errors, respectively. Up to t_v errors are considered in (3) since GII codes cannot correct more than t_v errors in any sub-word. Unlike E_1 or D_1, K_w is relatively large and hence it can be derived from simulations over a limited number of random samples, such as 10^8.

A type iv) sub-word has either up to t_0 errors or more than t_1 errors. Otherwise, $\Lambda(x)$ must have changed in the first nested decoding round since it is t_1-error-correcting. Further nested decoding is only required when the sub-word has more than t_1 errors and the corresponding probability is analyzed as follows. In each iteration of the nested KES [7], $\Lambda(x)$ is only updated when the conditions $\hat{\Delta}_0^{(r)} \neq 0$ and $k^{(r)} \geq -1$ are satisfied. Here $\hat{\Delta}_0^{(r)}$ is the discrepancy coefficient and $k^{(r)} = L_B^{(r)} - L_\Lambda^{(r)}$ is the length difference between the auxiliary polynomial $B^{(r)}(x)$ and $\Lambda^{(r)}(x)$. Hence, if a sub-word is of type iv), these conditions are not satisfied in each iteration of the nested KES in the first decoding round. Additionally, since $d^{(2t_0)} \leq t_0$ in type iv), it can be derived from the nested KES algorithm [7] that $L_B^{(2t_0)} \geq t_0 - 1$. Also $L_\Lambda^{(2t_0)} + L_B^{(2t_0)} = 2t_0 - 1$ from [10]. Thus, $L_\Lambda^{(2t_0)} \leq t_0$. Accordingly $k^{(2t_0)} = L_B^{(2t_0)} - L_\Lambda^{(2t_0)} \geq -1$ holds at the beginning of the nested KES in decoding round 1 and hence $\hat{\Delta}_0^{(2t_0)}$ must be zero if $\Lambda(x)$ is not updated. Following the nested KES algorithm in [7], if $k^{(2t_0)} \geq -1$, $k^{(r)} \geq -1$ for each of the later iterations. Hence, $\hat{\Delta}_0^{(r)}$ must be also zero for each iteration. For random and independent input errors, the discrepancy coefficient is a random value over $GF(2^q)$ and it can be zero with probability around 2^{-q} [10]. Since the nested KES in decoding round 1 has $t_1 - t_0$ iterations, given a type iv) sub-word, the probability that it has more than t_1 errors is

$$P_2 = \frac{E_2}{D_2} = \frac{\left(\sum_{w=t_1+1}^{t_v} \phi_w M_w\right) 2^{-q(t_1-t_0)}}{\sum_{w=0}^{t_0} \phi_w + \left(\sum_{w=t_1+1}^{t_v} \phi_w M_w\right) 2^{-q(t_1-t_0)}}, \quad (4)$$

where M_w is the probability that a w-error-corrupted sub-word is miscorrected and not detected in the sub-word decoding. It can be also derived from simulations over a limited number of random samples.

The denominator of (4) is much larger than that of (3) since the probability of having a smaller number of errors is much higher. Also K_w is in general larger than M_w and the numerator of (4) is multiplied with $2^{-q(t_1-t_0)}$. As a result, P_1 is much larger than P_2. For the example GII-BCH code, it can be calculated that $P_1 = 1.4 \times 10^{-3}$ and $P_2 = 4.5 \times 10^{-13}$ at BER=10^{-3}. Hence, a type iii) sub-word is much more likely to have more than t_1 errors than a type iv) sub-word. A similar sub-word selection strategy can be also extended to each of the other nested decoding rounds.

The additional FER degradation caused by the nested decoding using the proposed scheme is analyzed as follows. Nested decoding round i can handle at most $v+1-i$ sub-words with extra errors. Denote the total number of sub-words of type i) and ii) at the end of nested decoding round 1 by h. If $h > v-1$, the errors are not correctable anyway and the proposed scheme does not cause additional FER degradation. If $h = v-1$, all the sub-words to be sent to nested decoding round 2 indeed have more than t_1 errors and there is no performance loss either. When $h < v-1$, the remaining $v-h-1$ sub-words to be sent to nested decoding round 2 are selected from type iii) and then type iv) sub-words. If all of them are of type iii), performance loss occurs when the sub-word not passed to nested decoding round 2 but selected for round 1 has more than t_1 errors. Such FER degradation is upper bounded by

$$F_1 = \sum_{h=0}^{v-2} \binom{v}{h} \Big(\sum_{w=t_0+1}^{t_v} \phi_w H_w \Big)^h D_1^{v-h-1} E_1,$$

where H_w is the probability that a w-error-corrupted sub-word belongs to type i) or ii) in nested decoding round 1 and it can be derived from simulations over a limited number of samples.

If some of the remaining $v-h$ sub-words are of type iv), performance degradation also happens when the type iv) sub-word not passed to nested decoding round 2 but selected for round 1 has more than t_1 errors. Similarly, the upper bound

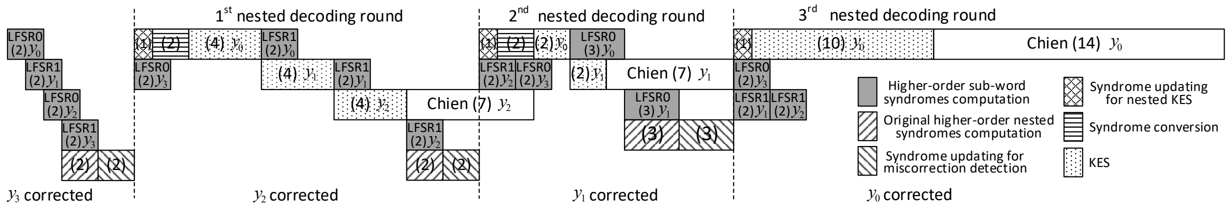

Fig. 3. Worst-case nested decoding latency utilizing the proposed schemes for the GII-BCH [4,3] code with $[t_0, t_1, t_2, t_3]$=$[3, 5, 6, 11]$.

of the extra FER for this case is

$$F_2 = \sum_{h=0}^{v-2} \sum_{\eta=0}^{v-h-1} \binom{v}{h} \Big(\sum_{w=t_0+1}^{t_v} \phi_w H_w \Big)^h \binom{v-h}{\eta} D_1^{\eta} D_2^{v-h-\eta-1} E_2,$$

where η represents the number of sub-words of type iii). The performance loss of applying the proposed strategy in later nested decoding rounds can be derived in a similar way.

In the first-stage sub-word decoding, since there is no $\Lambda(x)$ from previous decoding rounds, the selection of the sub-words for nested decoding is still carried out using the Chien search results as in [3]. Adding up the FERs from all decoding rounds, the overall FER of GII decoding can be estimated. For the example GII-BCH code, the estimated FER incorporating the proposed strategy is shown in Fig. 1. The estimations match well with the simulation results at higher BERs. Compared to selecting the sub-words based on Chien search results in every decoding round as in [3], the performance degradation caused by the proposed scheme is negligible.

Fig. 3 shows the worst-case nested decoding latency after applying the proposed scheme. Although Chien search is still needed to find the roots of those correct error locator polynomials, their results are not needed in order to decide which sub-words to be sent to the next nested decoding round in our proposed scheme. As a result, the next nested decoding round can start right after the miscorrection detection is done and significant latency reduction is achieved.

B. Efficient LFSR for Higher-Order Syndrome Computation

Testing if higher-order nested syndromes of the decoding results are zero is required in order to detect miscorrections for short GII-BCH codes. Besides, higher-order syndromes of the corrected sub-words are needed for the nested KES. Hence higher-order syndromes need to be repeatedly computed according to (2) over the decoding rounds. As shown in Fig. 2, prior designs reuse the nested KES architecture for the computations in (2). When the Chien search result is no longer needed for sub-word selection as proposed in Section III.A, the higher-order syndrome computation becomes the latency bottleneck. To reduce the latency, (2) can be computed by using an LFSR as shown in Fig. 4(a). For short GII codes, since the highest degree of $\Lambda(x)$ involved in (2) is at most t_{v-1}, which is small, such an LFSR does not suffer from long critical path and requires small area.

For the KES in nested decoding round i, $2(t_i - t_{i-1})$ higher-order syndromes are needed for each uncorrected sub-word. For BCH codes, an odd syndrome S_{2j+1} equals S_j^2. The squaring can be simply implemented by cyclical bit shift in normal basis representation [11]. Hence, only even syndromes need to be computed from (2). The miscorrection detection can be also done by using all even syndromes. However, the traditional LFSR requires two clock cycles to compute each even syndrome and is not efficient.

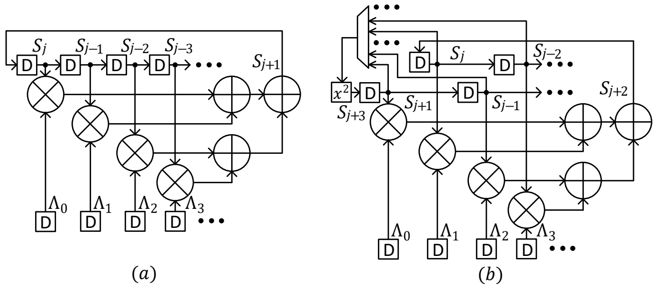

Fig. 4. LFSR architectures: (a) traditional; (b) proposed.

Next, an efficient LFSR architecture is developed to reduce the even syndrome computation latency by half. To skip the computation of the odd syndromes, each syndrome in Fig. 4(a) needs to be moved to the second register on the right in the next clock cycle. This can be achieved by splitting the even and odd syndromes into the upper and lower rows, respectively, as shown in Fig. 4(b). The sum of the products between the syndromes and $\Lambda(x)$ coefficients is S_{j+2}. This even syndrome is fed back to the leftmost register in the upper row for the next clock cycle. Besides, S_{j+3} is needed as the input to the register in the lower row. This odd syndrome is derived by squaring a previous syndrome, which is selected by a multiplexer.

For the example code with $[t_0, t_1, t_2, t_3]$=$[3, 5, 6, 11]$, the latency of the worst-case nested decoding process incorporating the proposed LFSR higher-order syndrome computation is shown in Fig. 3. Although the required higher-order syndromes remain the same in the proposed scheme, they are calculated using separate LFSRs in parallel with the KES instead of after the KES. Accordingly, significant latency reduction is achieved with small area overhead. As mentioned previously, $[\sigma_1, \sigma_2, \sigma_3]$=$[4, 3, 3]$ higher-order nested syndromes are needed before decoding round 1, 2, and 3, respectively, to detect miscorrections. v+1−i nested words are utilized in decoding round i and hence $\lceil \sigma_i/(v+1-i) \rceil$ higher-order syndromes need to be computed for each of them before decoding round i. Since each nested syndrome is a linear combination of all sub-word syndromes of the same order, $\lceil \sigma_i/(v+1-i) \rceil$ higher-order syndromes need to be calculated using (2) for each sub-word. If this number is at least $t_i - t_{i-1}$, then no additional higher-order sub-word syndrome needs to be computed for the KES.

The most efficient KES architecture for sub-word decoding is fully pipelined [12] and it finishes the KES for one sub-word in each clock cycle. Since the proposed LFSR computes $\lceil \sigma_1/v \rceil$=2 higher-order syndromes in 2 clock cycles, only 2 LFSRs are required from all the m=4 sub-words as shown in Fig. 3. By the end of nested decoding round 1,

TABLE I
COMPLEXITIES OF NESTED DECODERS FOR GII-BCH [4,3] CODE WITH $[t_0, t_1, t_2, t_3]$=$[3, 5, 6, 11]$

	worst-case latency (# clks)	total area (μm^2)	total power (μW)
prior best design	88	120722	214106
proposed design	65	131076	230752

$\lceil \sigma_2/(v{-}1) \rceil$=2 higher-order syndromes of each sub-word are needed for miscorrection detection. Since $y_3(x)$ got corrected in the sub-word decoding, its $\Lambda(x)$ does not change and its higher-order syndrome computation can start early at the beginning of this decoding round. The higher-order syndromes for the other three sub-words are computed right after the corresponding KES is completed. For nested decoding round 2, $\lceil \sigma_3/(v{-}2) \rceil$=3 higher-order syndromes from every sub-word are required for miscorrection detection. However, $S_{12}^{(2)}$ and $S_{12}^{(3)}$ have already been calculated in the previous round and the error locator polynomials for $y_2(x)$ and $y_3(x)$ remain the same. Hence, only 2 more higher-order syndromes need to be computed for $y_2(x)$ and $y_3(x)$ each in round 2. In the last nested decoding round, although $t_3{-}t_2$=5 higher-order even syndromes, $S_{12}^{(l)}$ through $S_{20}^{(l)}$, are needed for the KES, syndromes of order up to $2(t_2{-}1{+}\lceil \sigma_3/(v{-}2) \rceil)$=16 have already been computed previously for miscorrection detection. Therefore, only 2 more higher-order syndromes need to be computed in this round for each corrected sub-word.

IV. LATENCY AND COMPLEXITY COMPARISONS

Using the GII-BCH [4,3] code as an example, this section compares the nested decoder using the proposed schemes with the best previous possible designs [3], [7]–[9].

As shown in Fig. 2, the previous design needs 88 clock cycles for the worst-case decoding process. This latency is reduced to 65 clock cycles as shown in Fig. 3. Hence our new design achieves (88-65)/88=26% reduction in the worst-case nested decoding latency. Also, the only different computation unit in the proposed design is the LFSR. Its critical path is not longer than that of the KES architecture for short GII codes. Therefore, the proposed design does not degrade the achievable clock frequency. Our design does not reduce the average GII decoding latency much, since only the sub-word decoding is activated most of the time. However, the average latency is much shorter and is not a concern.

Two LFSRs are utilized to compute all the involved higher-order syndromes in Fig. 3. The $\Lambda(x)$ of the sub-word corrected in decoding round 0 is of degree t_0 at most and it does not change. LFSR0 is used to compute the higher-order syndromes for this sub-word and hence it has $t_0{+}1$ taps. LFSR1 is employed to compute the higher-order syndromes for all the other sub-words. $\deg(\Lambda(x))$ grows over the nested decoding rounds and can reach t_{v-1} before the last round. Hence, LFSR1 has $t_{v-1}{+}1$ taps. Overall, the two LFSRs need $t_0{+}1{+}t_{v-1}{+}1$=11 multipliers, 9 adders, $2(t_0{+}1{+}t_{v-1}{+}1)$=22 registers, a 4-to-1 multiplexer, and a 7-to-1 multiplexer. These are the only area overheads of the proposed design.

The proposed and prior designs are synthesized using TSMC $65nm$ process under T=$1ns$ timing constraint and the results are listed in Table I. The LFSRs are very small compared to other decoder components. Hence, the proposed design has only (131076-120722)/120722=8.5% area overhead under the same timing constraint.

The latency reduction achievable by the proposed schemes changes with code parameters and parallelisms of computation units. However, the Chien search is the most hardware-consuming unit. Using a higher parallelism increases the area substantially and may also cause routing congestion. By eliminating the Chien search from the sub-word selection decision and carrying out higher-order syndrome computation in parallel with KES using small-area LFSR, significant latency reduction can be also achieved under other settings. For a GII-BCH [4,2] code with $[t_0, t_1, t_2]$=$[4, 6, 11]$, it can be calculated that the worst-case nested decoding latency is reduced from 62 to 49 clock cycles. The saving is more significant for codes with more nested decoding rounds, since many clock cycles are eliminated by the proposed schemes in each round.

V. CONCLUSIONS

This paper proposes two new schemes to reduce the nested decoding latency of short GII-BCH codes. In our schemes, sub-words with extra errors are identified based on how the error locator polynomial changes over the previous decoding round instead of Chien search results. Analysis shows that the resulted error-correcting performance loss is negligible. Additionally, a low-complexity LFSR is developed to compute the even syndromes with half of the latency. Utilizing the proposed schemes, the nested decoding latency is substantially reduced with very small area increase. Future work will investigate more efficient implementations of GII decoders.

REFERENCES

[1] X. Tang and R. Koetter, "A novel method for combining algebraic decoding and iterative processing," *IEEE Int. Symp. Inf. Theory*, Seattle, WA, USA, Jul. 2006, pp. 474-478.

[2] Y. Wu, "Generalized integrated interleaved codes," *IEEE Trans. Inf. Theory*, vol. 63, no. 2, pp. 1102-1119, Feb. 2017.

[3] Z. Xie and X. Zhang, "Miscorrection mitigation for generalized integrated interleaved BCH codes," *IEEE Commun. Lett.*, vol. 25, no. 7, pp. 2118-2122, Jul. 2021.

[4] X. Tang and R. Koetter, "On the performance of integrated interleaving coding schemes," *IEEE Int. Symp. Inf. Theory*, Chicago, IL, USA, Jul. 2004, pp. 329-329.

[5] W. Li, J. Lin, and Z. Wang, "A 124-Gb/s decoder for generalized integrated interleaved codes," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 66, no. 8, pp. 3174-3187, Aug. 2019.

[6] D. V. Sarwate and N. R. Shanbhag, "High-speed architecture for Reed-Solomon decoders," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 9, no. 5, pp. 641-655, Oct. 2001.

[7] Z. Xie and X. Zhang, "Efficient nested key equation solver for short generalized integrated interleaved BCH codes," *IEEE Int. Symp. Circuits Syst.*, in press.

[8] Y. J. Tang and X. Zhang, "Low-complexity resource-shareable parallel generalized integrated interleaved encoder," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 69, no. 2, pp. 694-706, Feb. 2022.

[9] X. Zhang and Z. Xie, "Efficient architectures for generalized integrated interleaved decoder," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 66, no. 10, pp. 4018-4031, Oct. 2019.

[10] Y. Wu, "New scalable decoder architectures for Reed-Solomon codes," *IEEE Trans. Commun.*, vol. 63, no. 8, pp. 2741-2761, Aug. 2015.

[11] X. Zhang and Z. Wang, "A low-complexity three-error-correcting BCH decoder for optical transport network," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 59, no. 10, pp. 663-667, Oct. 2012.

[12] Z. Xie and X. Zhang, "Efficient sub-codeword key equation solver for generalized integrated interleaved BCH decoder," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 69, no. 1, pp. 85-89, Jan. 2022.