Low-Complexity Parallel Chien Search Architecture Based on Vandermonde Matrix Decomposition

Yok Jye Tang and Xinmiao Zhang
Department of Electrical and Computer Engineering
The Ohio State University
Columbus, OH 43210, USA

***Abstract*—Reed-Solomon (RS) and BCH codes are among the most broadly used error-correcting codes in digital communication and storage systems. The Chien search step accounts for a significant part of the overall decoder complexity of these codes. The Chien search can be expressed as a Vandermonde matrix multiplication. This paper develops a novel Vandermonde matrix decomposition that significantly reduces the number of multiplications needed for the Chien search. Further reformulation on the matrix decomposition is also proposed to enable efficient parallel processing in hardware implementation. Accordingly, a low-complexity parallel Chien search architecture is designed. For an example 9-error-correcting RS or BCH code over $GF(2^{10})$, the proposed design with $40, 60,$ and 80-parallel processing achieves 11%, 14%, and 17%, respectively, area reduction compared to the best prior design with the same throughput and similar latency.**

***Index Terms*—BCH decoder, Chien search, error-correcting codes, Reed-Solomon decoder, Vandermonde matrix decomposition.**

I. Introduction

Reed-Solomon (RS) and BCH codes are widely used in digital communication and storage systems, such as optical transport networks [1], digital video broadcasting [2], and flash memory [3]. The Chien search that finds the roots of the error-locator polynomial by exhaustively searching over all finite field elements is one of the crucial steps in RS/BCH decoding. Highly-parallel Chien search is needed to achieve high decoding throughput required by many applications, and it accounts for a significant part of the overall decoder complexity.

Chien search architectures can be found in many literatures, such as [4]–[11]. In the conventional parallel Chien search architecture [4], each polynomial coefficient is associated with a column of constant multipliers that share a common input. Substructure sharing can be applied to the multipliers in the same column [5], [6] to reduce the gate count. The design in [7] put off the modular reductions for finite field multiplications to simplify the multiplications. However, its gate count is larger than that of applying substructure sharing to the multipliers across different columns and rows as in [8]. Besides, the power consumption of the Chien search can be reduced through factorizing out the roots of the polynomials [9], [10]. When evaluating a finite field element, the design in [11] employs a two-step process and the second step is activated only if the element seems to be a possible root from the first step. However, the low-power designs in [9]–[11] do not reduce the gate count of the Chien search architecture.

The Chien search can be expressed as a Vandermonde matrix multiplication. It is reformulated to enable intermediate result sharing in [12], [13]. However, these designs are limited to Vandermonde matrices with up to 7 columns. Besides, they are targeting at software implementations and do not lead to regular parallel architectures. Reed-Muller transformation is utilized in [14] to decompose a Vandermonde matrix to two matrices that have fewer non-zero entries. However, this decomposition requires the entries in the Vandermonde matrix to be in binary order. Although the order does not matter for fully-parallel Chien search, practical applications can not adopt fully-parallel Chien search due to its overwhelming complexity and routing congestion issue. For a non-fully parallel design that carries out the Chien search over a group of elements at a time, the search needs to be done in the order of consecutive power of the primitive finite field element, so its outputs can be added up to the decoder inputs in the same order to correct errors without complex re-routing network.

This paper proposes novel methods to decompose the Vandermonde matrix for non-fully parallel Chien search in RS/BCH decoding. The Vandermonde matrix whose elements are in the order of consecutive power of the primitive finite field element is decomposed into matrices with a substantially smaller number of non-zero entries by utilizing the standard basis representation of finite field elements. In order to achieve efficient parallel design, the proposed decomposition is incorporated into a reformulated parallel processing equation. As a result, a low-complexity parallel Chien search architecture with significantly reduced number of constant multipliers is developed. For an example 9-error-correcting RS or BCH code over $GF(2^{10})$, the proposed design with $40, 60,$ and 80-parallel processing achieves 11%, 14%, and 17%, respectively, area reduction compared to the best previous design in [8] with the same throughput and similar latency.

This paper is organized as follows. Section II introduces the previous parallel Chien search architecture. The proposed Vandermonde matrix decomposition is detailed in Section III. The reformulation for parallel processing is proposed in Section IV. Section V provides complexity analyses and comparisons. Conclusions are in Section VI.

This material is based upon work supported by the National Science Foundation under Award No. 2011785.

II. Chien search in RS and BCH decoding

Consider a t-error-correcting RS or BCH code of length n constructed over finite field $GF(2^q)(q \in \mathbb{Z}^+)$. The decoder first computes $2t$ syndromes. Using these syndromes, the key-equation solver (KES) computes the error-locator polynomial. If there are $t' \leq t$ errors, the reciprocal of the error-locator polynomial denoted by $\Lambda(x)$ is in the format of

$$\Lambda(x) = (x+\alpha^{i_1})(x+\alpha^{i_2})\cdots(x+\alpha^{i_{t'}}), \tag{1}$$

where $0 \leq i_1, i_2, \cdots, i_{t'} < n$ are the indices of the erroneous symbols. The roots of $\Lambda(x)$ are computed by the Chien search, which is to evaluate $\Lambda(x)$ over n finite field elements, $1, \alpha, \cdots, \alpha^{n-1}$, where α is a primitive element of $GF(2^q)$. The evaluation results are used to decide whether the received symbols that go through a first-in-first-out buffer need to be corrected. Hence the evaluation has to be carried out on finite field elements in the order of increasing power of α, *i.e.* $1, \alpha, \alpha^2, \alpha^3, \cdots$.

To achieve high throughput, a highly-parallel Chien search architecture is needed. For t-error-correcting decoding, $\deg(\Lambda(x)) \leq t$ and $\Lambda(x)$ can be re-written as $\Lambda_t x^t + \Lambda_{t-1}x^{t-1} + \cdots + \Lambda_0$. Fig. 1 shows a P-parallel Chien search architecture. In clock cycle 0, the coefficients of $\Lambda(x)$ are sent through the multiplexers. Each row in the dotted box consists of t multipliers and computes

$$\Lambda(\alpha^j) = \Lambda_t\alpha^{tj} + \Lambda_{t-1}\alpha^{(t-1)j} + \cdots + \Lambda_0,$$

where $0 \leq j < P$. Then the evaluation values, $\Lambda(1), \cdots, \Lambda(\alpha^{P-1})$, are generated in clock cycle 0.

In later clock cycles, the multiplexers on the bottom of Fig. 1 pass through the outputs of the multipliers in the feedback loops. The l-th feedback loop from the right side iteratively computes $\Lambda_l\alpha^{liP}$ in clock cycle $i(1 \leq i < \lceil n/P \rceil)$. As a result, the j-th output of the architecture in Fig. 1 in clock cycle i is

$$\begin{aligned}&\Lambda_t\alpha^{tiP}\alpha^{tj} + \Lambda_{t-1}\alpha^{(t-1)iP}\alpha^{(t-1)j} + \cdots + \Lambda_0\\ &= \Lambda_t\alpha^{t(j+iP)} + \Lambda_{t-1}\alpha^{(t-1)(j+iP)} + \cdots + \Lambda_0\\ &= \Lambda(\alpha^{j+iP}).\end{aligned}$$

This architecture needs $\lceil n/P \rceil$ clock cycles to finish the Chien search. It consists of tP constant multipliers, tP adders, $t+1$ multiplexers, and $t+1$ registers.

Each constant multiplication over $GF(2^q)$ can be described as a $q \times q$ constant binary matrix multiplication. Substructure sharing can be applied among all the constant multipliers in Fig. 1 to reduce the gate count as in [5], [6], [8]. Among these designs, the design in [8] that finds the shareable gates of the constant binary matrix multipliers across different rows and columns achieves the lowest complexity.

III. Vandermonde matrix decomposition for low-complexity Chien search

In RS/BCH decoders, the highly-parallel Chien search accounts for a significant portion of the overall decoder complexity. The Chien search can be described as a Vandermonde

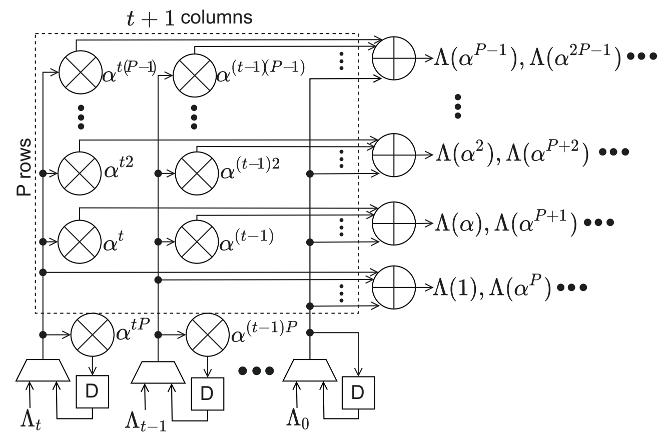

Fig. 1. Conventional P-parallel Chien search architecture for t-error-correcting RS/BCH decoding.

matrix multiplication. This section proposes a novel scheme to decompose the Vandermonde matrix into two matrices with a substantially smaller number of non-zero entries. Accordingly, the number of multiplications in the Chien search can be significantly reduced.

Let $\underline{\Lambda} = [\Lambda_0, \Lambda_1, \cdots, \Lambda_t]$ and $\underline{r} = [\Lambda(1), \Lambda(\alpha), \cdots, \Lambda(\alpha^{n-1})]$. The Chien search can be described as

$$\underline{r} = \underline{\Lambda} \cdot V, \tag{2}$$

where V is a Vandermonde matrix in the format of

$$V = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \alpha & \cdots & \alpha^{n-1} \\ \vdots & \vdots & \cdots & \vdots \\ 1 & \alpha^t & \cdots & \alpha^{t(n-1)} \end{bmatrix}. \tag{3}$$

Inspired by [14], this paper proposes to decompose V into two matrices as

$$V = E \cdot B, \tag{4}$$

where the dimension of E is $(t+1) \times n$ and that of B is $n \times n$. Unlike the decomposition method in [14] that only applies to Vandermonde matrices whose entries in the second row are finite field elements in binary order, our proposed method decomposes Vandermonde matrices whose entries are in consecutive power of α as in (3). This is needed for developing our proposed low-complexity parallel Chien search architecture for RS/BCH decoding in Section IV.

A standard basis of $GF(2^q)$ is $\{1, \alpha, \cdots, \alpha^{q-1}\}$. Each α^j in the second row of V can be represented in standard basis as $\alpha^j = a_0^{(j)} + a_1^{(j)}\alpha + \cdots + a_{q-1}^{(j)}\alpha^{q-1}$, where $a_0^{(j)}, a_1^{(j)}, \cdots, a_{q-1}^{(j)}$ are binary bits. For each row with indices $i = 2^l (l \geq 0)$, α^{ij} in columns $q \leq j < n$ of V can be written as

$$\begin{aligned}\alpha^{ij} &= (a_0^{(j)} + a_1^{(j)}\alpha + \cdots + a_{q-1}^{(j)}\alpha^{q-1})^i \\ &= a_0^{(j)} \cdot 1 + a_1^{(j)} \cdot \alpha^i + \cdots + a_{q-1}^{(j)} \cdot \alpha^{(q-1)i}\end{aligned}. \tag{5}$$

Notice that $1, \alpha^i, \cdots, \alpha^{(q-1)i}$ in (5) are the first q elements in row i of V. Therefore, the first q entries of E in row $i = 2^l$ are set to those of V. The coefficients in (5) are the same for every row i in the format of 2^l. Hence the first q entries in j-th column of B are set to $[a_0^{(j)}, a_1^{(j)}, \cdots, a_{q-1}^{(j)}]^T$, where T denotes transpose. Then the entries in row $i = 2^l$ and columns $q \le j < n$ of E are set to zero. Accordingly, the product of row $i = 2^l$ of E and column j of B equals α^{ij} in V.

To derive $\alpha^{i'j}$ in row $i' \neq 2^l$ of V, the first q entries in row i' of E are set to $1, \alpha^{i'}, \cdots, \alpha^{i'(q-1)}$ since the $q \times q$ entries in the upper-left corner of B is an identity matrix. In this case, the entry in row i' and column $q \le j < n$ of E can be calculated as

$$e_{i',j} = \alpha^{i'j} - (a_0^{(j)} + a_1^{(j)}\alpha^{i'} + \cdots + a_{q-1}^{(j)}\alpha^{(q-1)i'})$$

to make the product of row i' of E and column j of B equal to $\alpha^{i'j}$. Accordingly, the entries in the diagonal of B are set to '1' and those in rows $q \le i < n$ are set to '0'.

Let $E_{i,j}$ and $B_{i,j}$ denote the entries of E and B, respectively, in row $0 \le i \le t$ and column $0 \le j < n$. In summary, the entries of E are

$$E_{i,j} = \begin{cases} \alpha^{ij} & \text{, for } 0 \le i \le t, 0 \le j < q \\ 0 & \text{, for } i = 2^l (l \ge 0), q \le j < n \\ e_{i,j} & \text{, for } i \neq 2^l, q \le j < n \end{cases}. \tag{6}$$

B is an $n \times n$ upper-triangular binary matrix and its entries are

$$B_{i,j} = \begin{cases} 1 & \text{, for } i = j \\ a_i^{(j)} & \text{, for } 0 \le i < q, q \le j < n \\ 0 & \text{, elsewhere} \end{cases}. \tag{7}$$

Take $n = 7$ and $t = 3$ as an example. Consider finite field $GF(2^3)$ constructed using primitive polynomial $p(x) = x^3 + x + 1$. Use the standard basis $\{1, \alpha, \alpha^2\}$, where α is the root of $p(x)$. According to (6) and (7), E and B can be formed as follows

$$E = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & \alpha & \alpha^2 & 0 & 0 & 0 & 0 \\ 1 & \alpha^2 & \alpha^4 & 0 & 0 & 0 & 0 \\ 1 & \alpha^3 & \alpha^6 & \alpha^4 & 1 & \alpha^6 & \alpha \end{bmatrix}$$

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

Since B is a binary matrix, the total number of constant finite field multiplications needed in our proposed design depends on the number of entries in E that are not '0' or '1'. For the example above, E has 9 entries that are not '0' or '1'. Accordingly, the proposed design reduces the number of multiplications in the Chien search from $t(n-1) = 18$ as needed in the direct Vandermonde matrix multiplication to 9.

From (6), the number of '1' in the first q columns of E is $t + q$. In columns $q \le j < n$, the entries in the first row are either '0' or '1'. Besides, there are $\lceil log_2(t) \rceil$ rows in E whose indices are powers of 2. For these rows, the entries in columns $q \le j < n$ are '0'. Therefore, the total number of finite field multiplications needed in our design is at most

$$\begin{aligned} &(t+1)n - (t + q + (n-q) + \lceil log_2(t) \rceil (n-q)) \\ &= tn - (t + \lceil log_2(t) \rceil (n-q)). \end{aligned} \tag{8}$$

This number is even smaller if some $e_{i,j}$ happen to be '1'. As a result, the complexity of multiplying E and B is much lower compared to that of directly multiplying the Vandermonde matrix, which needs $t(n-1)$ finite field multiplications.

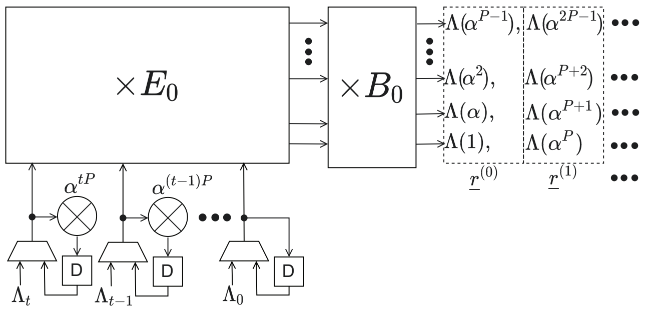

Fig. 2. Proposed P-parallel Chien search architecture based on Vandermonde matrix decomposition.

IV. Vandermonde matrix reformulation for parallel Chien search

For longer RS/BCH codes, multiplying the entire E matrix all at once to achieve fully-parallel Chien search is impractical due to very high hardware complexity and routing congestion. On the other hand, dividing the E matrix into blocks of columns and multiplying one block of columns in each clock cycle would require general finite field multipliers since the $e_{i,j}$ entries are different. A general multiplier over $GF(2^q)$ has much higher complexity than a constant multiplier [15], [16]. In this section, the decomposition proposed in the previous section is incorporated into a reformulated version of (2) for parallel processing. As a result, the products for every block of columns can be derived by multiplying the same matrices that need significantly fewer constant multipliers with an iterative update.

Assuming n is divisible by P, the Vandermonde matrix in (3) can be divided into n/P sub-matrices as follows

$$V = [V_0, V_1, \cdots, V_{n/P}]. \tag{9}$$

V_0 is a Vandermonde matrix in the format of (3) with P columns. Hence the decomposition proposed in the previous section can still be applied. It was shown in [8] that V_i for $i > 0$ in (9) can be re-written as

$$V_i = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & \alpha^{iP} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha^{tiP} \end{bmatrix} \cdot V_0. \tag{10}$$

TABLE I
COMPLEXITIES AND SYNTHESIS RESULTS USING TSMC 65NM PROCESS UNDER $T = 1ns$ TIMING CONSTRAINT FOR P-PARALLEL CHIEN SEARCH ARCHITECTURES WITH $n = 1023$ AND $t = 9$ OVER $GF(2^{10})$

	# of constant multipliers	# of adders	# of multiplexers	# of register	total # of equivalent XORs with sub-struc. sharing	total area (μm^2) (normalized)	latency (# of clock cylces)	critical path (# of gates)
				$P = 40$				
Chien search architecture in [8]	360	360	10	10	8216	23693.64 (1.00)	26	7
Proposed Chien search architecture	240	318	10	50	7100	21076.61 (0.89)	27	7
				$P = 60$				
Chien search architecture in [8]	540	540	10	10	12874	35317.68 (1.00)	18	7
Proposed Chien search architecture	340	492	10	70	10411	30206.08 (0.86)	19	7
				$P = 80$				
Chien search architecture in [8]	720	720	10	10	17391	45008.30 (1.00)	13	7
Proposed Chien search architecture	440	690	10	90	13726	37318.06 (0.83)	14	7

Divide the vector $\underline{r}$ in (2) into n/P groups as $\underline{r} = [\underline{r}^{(0)}, \underline{r}^{(1)}, \cdots, \underline{r}^{(n/P)}]$, where $\underline{r}^{(i)} = [\Lambda(\alpha^{iP}), \Lambda(\alpha^{iP+1}), \cdots, \Lambda(\alpha^{(i+1)P-1})]$. Then $\underline{r}^{(i)} = \underline{\Lambda} \cdot V_i$. By using (10), $r^{(i)} (i > 0)$ can be reformulated as

$$\underline{r}^{(i)} = [\Lambda_0, \Lambda_1\alpha^{iP}, \cdots, \Lambda_t\alpha^{tiP}] \cdot V_0. \tag{11}$$

V_0 can be decomposed into two sub-matrices as $V_0 = E_0 \cdot B_0$ by using the method proposed in the previous section. Then (11) becomes

$$\underline{r}^{(i)} = [\Lambda_0, \Lambda_1\alpha^{iP}, \cdots, \Lambda_t\alpha^{tiP}] \cdot E_0 \cdot B_0. \tag{12}$$

As a result, each $\underline{r}^{(i)}$ can be computed by multiplying the same E_0 and B_0 matrices. If n is not divisible by P, the extra $P - n\%P$ outputs from the last group are ignored.

Consider the same example used in the previous section with $n = 7$ and $t = 3$. For $P = 4$, $\underline{r}$ is divided into two groups as $\underline{r} = [\underline{r}^{(0)}, \underline{r}^{(1)}]$. Using the proposed decomposition, $V_0 = E_0 \cdot B_0$, where

$$E_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 & 0 \\ 1 & \alpha^2 & \alpha^4 & 0 \\ 1 & \alpha^3 & \alpha^6 & \alpha^4 \end{bmatrix}$$

$$B_0 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Then

$$\underline{r}^{(0)} = [\Lambda_0, \Lambda_1, \cdots, \Lambda_t] \cdot E_0 \cdot B_0,$$

and $\underline{r}^{(1)}$ is derived by $[\Lambda_0, \Lambda_1\alpha^P, \cdots, \Lambda_t\alpha^{tP}] \cdot E_0 \cdot B_0$ with the last $P - n\%P = 1$ output ignored.

Fig. 2 shows the proposed P-parallel Chien search architecture based on Vandermonde matrix decomposition. It calculates $\underline{r}^{(i)}$ for $i = 0, 1, \cdots, n/P - 1$ in clock cycles $0, 1, \cdots, n/P - 1$, respectively. The entries in the row vector, $[\Lambda_0, \Lambda_1\alpha^{iP}, \cdots, \Lambda_t\alpha^{tiP}]$, are generated by the feedback loops located at the bottom in Fig. 2 in clock cycle i. This vector is multiplied by the E_0 and then B_0 matrices. Since the entries of E_0 are constant, its multiplication can be implemented by constant finite field multipliers. The multiplication by B_0 is implemented by finite field adders since B_0 is a binary matrix.

A constant multiplication over $GF(2^q)$ can be implemented as a $q \times q$ constant binary matrix multiplication. Similar to the best prior design in [8], sub-structure sharing can be applied to the constant multipliers corresponding to the entries in E_0 and B_0 in two dimensions to further reduce the gate count in our design.

V. COMPLEXITY ANALYSIS

In this section, the complexity of the proposed P-parallel Chien search architecture based on Vandermonde matrix decomposition is analyzed and compared.

In the proposed design shown in Fig. 2, the maximum number of constant multipliers required for multiplying E_0 is $tP - (t + \lceil log_2(t) \rceil (P - q))$ from (8). Taking into account the constant multipliers in the bottom feedback loops, the proposed P-parallel Chien search architecture has $t + tP - (t + \lceil log_2(t) \rceil (P - q)) = tP - \lceil log_2(t) \rceil (P - q)$ constant multipliers. A constant multiplier over $GF(2^q)$ can be implemented as a

$q \times q$ binary matrix multiplication. Sub-structure sharing can be applied to the multipliers across different rows and columns as in [8] to further reduce the gate count. Besides, the number of required finite field adders is decided by the number of non-zero entries in E_0 and B_0. As shown in Fig. 2, the proposed design also has $t+1$ multiplexers and $t+1$ registers.

Table I lists the numbers of multipliers, adders, multiplexers, and registers needed by the proposed parallel Chien search architectures with $P = 40, 60$, and 80 for a 9-error-correcting RS/BCH code with $n = 1023$ over $GF(2^{10})$. From simulations, it was found that the $q(t+1) \times qP$ constant binary matrix for the multiplication of E_0 has no more than $q(t+1)/2$ '1's in each column. Hence, it has $\lceil log_2(q(t+1)/2) \rceil = 6$ XOR gates in the data path. Besides, only the first q rows of B_0 may have multiple non-zero entries and each of the other rows has at most a single non-zero entry. Therefore, the data path of the B_0 matrix multiplication has at most $\lceil \log_2(q+1) \rceil = 4$ XOR gates. Pipelining is applied between the multiplications of E_0 and B_0 to reduce the critical path. Considering the multiplexers located at the bottom in Fig. 2, the critical path of the proposed P-parallel Chien search architecture for a t-error-correcting RS/BCH code over $GF(2^q)$ has at most $\lceil log_2(q(t+1)/2) \rceil + 1 = 7$ gates. The pipelining leads to one extra clock cycle on the latency and the Chien search is completed in $\lceil n/P \rceil + 1$ clock cycles. The pipelining registers are also included in Table I. Each adder over $GF(2^q)$ is implemented by q XOR gates. A q-bit 2-to-1 multiplexer and a q-bit register have around the same area as q and $3q$ XOR gates, respectively. Substructure sharing is applied to the E_0 and B_0 matrices and the total numbers of XOR gates required by our design can be estimated as listed in Table I. To further verify our design, it is synthesized using TSMC $65nm$ process under $T = 1ns$ timing constraint and the areas are also reported in Table I.

For comparisons, the best previous parallel Chien search architecture [8] is considered. It applies sub-structure sharing to the constant binary matrix multiplications across different rows and columns in the architecture of Fig. 1. This design has lower complexity than those only apply substructure sharing among the multiplications of the same column [5], [6]. The complexity and synthesis results of the design in [8] are listed in Table I. Compared to this design, our proposed design has a much smaller number of constant multipliers, since the entries of E_0 in row $2^l (l \geq 0)$ and columns $q \leq j < P$ are all zero. The number of '1's in column $q \leq j < P$ of B_0 is dependent on the standard basis representation of α^j. For small j, it typically has a small number of non-zero bits. As a result, the number of adders in our design is also reduced as shown in Table I. From the synthesis results, our design can reduce the area requirement by $11\%, 14\%$, and 17% compared to the architecture in [8] for $P = 40, 60$, and 80, respectively, with the same critical path and only one more clock cycle in the latency.

For a larger P, the area reduction achieved by the proposed design is more significant as shown in Table I, since all entries in column $q \leq j < P$ of the rows with indices 2^l in E_0 are zero. There is a bigger percentage of rows whose indices are in the format of 2^l when t is smaller. Hence, the area saving achievable by our proposed design would also be more significant for codes with smaller t.

VI. Conclusions

This paper develops a novel scheme that decomposes a Vandermonde matrix into two matrices with a smaller number of non-zero entries such that the number of finite field multiplications in the Chien search can be substantially reduced. Also a further reformulation on the matrix decomposition is proposed to enable efficient parallel Chien search, and a low-complexity implementation architecture is designed. Compared to the best previous architecture, the proposed design achieves substantial reduction in the silicon area and the achievable saving increases for higher parallelism. Future work will investigate different Vandermonde matrix decompositions that can further reduce the complexity of highly-parallel Chien search.

References

[1] X. Zhang and Z. Wang, "A low-complexity three-error-correcting BCH decoder for optical transport network,"*IEEE Trans. on Circuits and Syst.-II*, vol. 59, no. 10, pp. 663-667, Oct. 2012.
[2] X. Zhang, "High-speed and low-complexity parallel long BCH encoder," *IEEE Int. Symp. on Circuits and Syst.*, Seville, Spain, 2020, pp. 1-5
[3] D. Kim, K. R. Narayanan and J. Ha, "Symmetric block-wise concatenated BCH codes for NAND flash memories,"*IEEE Trans. on Comm.*, vol. 66, no. 10, pp. 4365-4380, Oct. 2018.
[4] X. Zhang, *VLSI Architectures for Modern Error-Correcting Codes*, CRC Press, Boca Raton, FL, USA, 2015.
[5] Y. Chen and K. K. Parhi, "Small area parallel Chien search architectures for long BCH codes," *IEEE Trans, on Very Large Scale Integ. (VLSI) Systems*, vol. 12, no. 5, pp. 545-549, May 2004.
[6] Q. Hu, Z. Wang, J. Zhang and J. Xiao, "Low complexity parallel Chien search architecture for RS decoder," *IEEE Int. Symp. on Circuits and Syst.*, pp. 340-343, Kobe, 2005.
[7] J. Cho and W. Sung, "Strength-reduced parallel Chien search architecture for strong BCH codes," *IEEE Trans. on Circuits and Syst.-II*, vol. 55, no. 5, pp. 427-431, May 2008.
[8] Y. Lee, H. Yoo and I. -C. Park, "Low-complexity parallel Chien search structure using two-dimensional optimization,"*IEEE Trans on Circuits and Syst-II*, vol. 58, no. 8, pp. 522-526, Aug. 2011.
[9] S. -Y. Wong, C. Chen and Q. M. Jonathan Wu, "Low power Chien search for BCH decoder using RT-level power management," *IEEE Trans. on Very Large Scale Integ. (VLSI) Syst."*, vol. 19, no. 2, pp. 338-341, Feb. 2011.
[10] X. Zhang, I. Dror and S. Alterman, "Low-power partial-parallel Chien search architecture with polynomial degree reduction," *IEEE Int. Symp. on Circuits and Syst.*, pp. 2459-2462, Montreal, QC, Canada, 2016.
[11] H. Yoo, Y. Lee and I. -C. Park, "Low-Power parallel Chien search architecture using a two-step approach," *IEEE Trans. on Circuits and Syst.-II*, vol. 63, no. 3, pp. 269-273, March 2016.
[12] L. Yu, Z. Lin, S. -J. Lin, Y. S. Han and N. Yu, "Fast encoding algorithms for Reed–Solomon codes with between four and seven parity symbols," *IEEE Trans. on Comput.*, vol. 69, no. 5, pp. 699-705, 1 May 2020.
[13] Y. J. Tang and X. Zhang, "Fast En/Decoding of Reed-Solomon Codes for Failure Recovery,"*in IEEE Trans. on Comput.*, vol. 71, no. 3, pp. 724-735, 1 March 2022.
[14] L. Yu, S. -J. Lin, H. Hou and Z. Li, "Reed-Solomon coding algorithms based on Reed-Muller transform for any number of parities," *IEEE Trans. on Computers*, vol. 72, no. 9, pp. 2677-2688, Sept. 2023.
[15] X. Zhang and Z. Xie, "Efficient architectures for generalized integrated interleaved decoder," *IEEE Trans. on Circuits and Syst.-I*, vol. 66, no. 10, pp. 4018-4031, Oct. 2019.
[16] Z. Xie and X. Zhang, "Reduced-complexity key equation solvers for generalized integrated interleaved BCH decoders,"*IEEE Trans. on Circuits and Syst-I*, vol. 67, no. 12, pp. 5520-5529, Dec. 2020.