A MPCN-Based Parallel Architecture in BCH Decoders for NAND Flash Memory Devices

Yi-Min Lin, Chi-Heng Yang, Chih-Hsiang Hsu, Hsie-Chia Chang, and Chen-Yi Lee

***Abstract*—According to large-page-size and random-bit-error characteristics, long-block-length Bose–Chaudhuri–Hochquenghem (BCH) decoders are applied to realize error correction in NAND Flash memory devices. To accelerate the decoding process in an area-efficient architecture, a parallel architecture with *minimal polynomial combinational network (MPCN)* for long BCH decoders is presented in this brief. The proposed design utilizes MPCNs to replace constant finite-field multipliers, which dominate the hardware complexity of the high-parallel Chien search architecture. Furthermore, both the syndrome calculator and the Chien search can be merged by exploiting our MPCN-based architecture, leading to significant hardware complexity reduction. From the synthesis results in the 90-nm CMOS technology, the MPCN-based joint syndrome calculation and Chien search has 46.7% gate count saving for parallel-32 BCH (4603, 4096; 39) decoder in contrast with the straightforward design.**

***Index Terms*—Bose–Chaudhuri–Hochquenghem (BCH) code, Chien search, error correction code, NAND Flash memory.**

I. Introduction

NAND Flash memory is one of the fast growing storage systems because of its nonvolatility, shock resistance, power efficiency, and random-access capability. For the low-cost and high-storage-density requirements, the multilevel-cell technology has been developed for NAND Flash memory devices recently [1], [2]. However, the reliability of NAND Flash memory devices is significantly degraded by the higher uncertainty of charging and detecting the multiple voltage range in a single cell. As a result, Bose–Chaudhuri–Hocquenghen (BCH) [3] codes are applied to provide error correction due to the random-bit-error characteristic [4], [5].

The conventional BCH decoder contains three major blocks, i.e., *syndrome calculator*, *key equation solver*, and *Chien search* [3]. In NAND Flash memory devices, the decoding latency is dominated by the syndrome calculator and the Chien search since the large page size demands BCH codes of long block length. To enhance the decoding throughput, parallel architectures are exploited, but the hardware complexity is significantly increased, particularly for the Chien search block. The hardware complexity of the high-parallel Chien search is dominated by the large number of constant finite-field multipliers (CFFMs). There have been several research for reducing the complexity of parallel Chien search. In 2004, Chen and Parhi developed a group matching algorithm (GMA) to share the common subexpressions among groups of CFFMs with the same multiplicand in the parallel Chien search block [6]. In 2008, a strength-reduced method was proposed for sharing the modulo operation with the primitive polynomial $f(x)$ among CFFMs that contribute to the same error location calculation [7]. These previous researches focus on the methods for reducing the complexity of groups of CFFMs. In this brief, however, we decrease the number of CFFMs to improve hardware efficiency by reformulating the Chien search equation with minimal polynomials. Instead of reducing the complexity of groups of CFFMs, the proposed design utilizes the *minimal polynomial combinational networks (MPCNs)* to replace the CFFMs. The hardware complexity is significantly reduced since the XOR gate count requirement of one MPCN is at most $m-1$, whereas that of one CFFM is usually proportional to m^2 in $GF(2^m)$.

Moreover, a BCH decoder with one-staged pipeline architecture will make the Flash memory controller more convenient to access the memory cells. The operations of both the syndrome calculator and the Chien search can be scheduled in different time slots for NAND Flash memory devices, implying that the hardware can be shared. Neither GMA nor strength-reduced methods can be applied for the syndrome calculator and the Chien search with the same architecture, although they can be applied for these two blocks individually. Nevertheless, based on the proposed MPCN-based architecture, the syndrome calculator and the Chien search can be merged with small overhead. The overall hardware complexity is significantly reduced.

This brief is organized as follows. Section II gives the background of conventional parallel Chien search architectures. The proposed MPCN-based algorithms and architectures for high-parallelism BCH decoders are presented in Section III. Based on the proposed methods, Section IV demonstrates the implementation and comparison results. Finally, Section V gives a conclusion of this brief.

II. Conventional Parallel Chien Search Architectures

An $(N, K; t)$ BCH code has a block length of N bits and an information length of K bits. While operating under $GF(2^m)$, it has error correcting capability t with $N - K \leq m \times t$. Once error location polynomial $\Lambda(x)$ is obtained in the decoding process, a Chien search block shown in Fig. 1 can be used to

Manuscript received February 23, 2011; revised May 4, 2011; accepted June 22, 2011. Date of publication September 8, 2011; date of current version October 19, 2011.This work was supported in part by the National Science Council under Grant NSC 97-2221-E-009-166 and in part by the Ministry of Economic Affairs of Taiwan under Grant 99-EC-17-A-01-S1-124. This paper was recommended by Associate Editor Z. Wang.

The authors are with the Department of Electronics Engineering and the Institute of Electronics, National Chiao Tung University, Hsinchu 300, Taiwan (e-mail: ymlin@si2lab.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2011.2161704

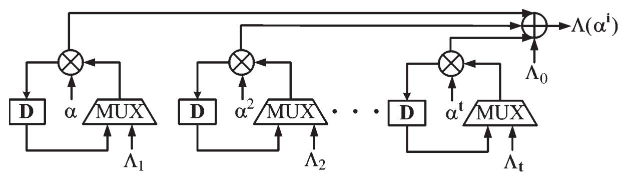


Fig. 1. Conventional Chien search architecture.

exhaustively examine whether $\Lambda(\alpha^i) = 0$ for $i = 0 \sim N-1$, where

$$\Lambda(\alpha^i) = \sum_{j=0}^{t} \Lambda_j \alpha^{(i)^j} = \sum_{j=1}^{t} \Lambda_j \alpha^{ij} + 1. \qquad (1)$$

Notice that an arbitrary element over $GF(2^m)$ is presented as $\sum_{i=0}^{m-1} a_i \alpha^i$ with binary coordinate a_i due to basis $\{\alpha^0, \alpha^1, \ldots, \alpha^{m-1}\}$. Therefore, $\Lambda_j \alpha^{ij}$ can be expressed as

$$\begin{aligned} P_{ij} &= \Lambda_j \alpha^{ij} \\ &= (\lambda_{j,0}\alpha^0 + \lambda_{j,1}\alpha^1 + \cdots + \lambda_{j,m-1}\alpha^{m-1})\alpha^{ij} \\ &= p_{ij,0}\alpha^0 + p_{ij,1}\alpha^1 + \cdots + p_{ij,m-1}\alpha^{m-1} \end{aligned} \qquad (2)$$

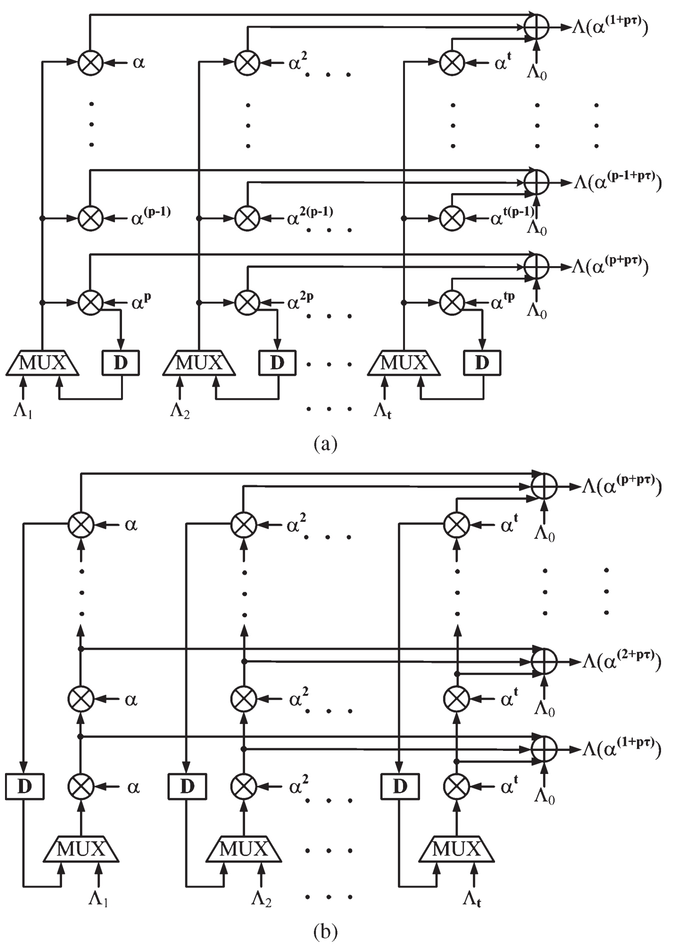
where $\{\lambda_{j,0}, \lambda_{j,1}, \ldots, \lambda_{j,m-1}\}$ and $\{p_{ij,0}, p_{ij,1}, \ldots, p_{ij,m-1}\}$ are the coordinates of Λ_j and P_{ij}, respectively, and

$$\begin{bmatrix} p_{ij,0} \\ p_{ij,1} \\ \vdots \\ p_{ij,m-1} \end{bmatrix} = \begin{bmatrix} \alpha_0^{ij} & \alpha_0^{ij+1} & \cdots & \alpha_0^{ij+m-1} \\ \alpha_1^{ij} & \alpha_1^{ij+1} & \cdots & \alpha_1^{ij+m-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m-1}^{ij} & \alpha_{m-1}^{ij+1} & \cdots & \alpha_{m-1}^{ij+m-1} \end{bmatrix} \begin{bmatrix} \lambda_{j,0} \\ \lambda_{j,1} \\ \vdots \\ \lambda_{j,m-1} \end{bmatrix}. \qquad (3)$$

Notice that binary element α_l^k stands for the lth coordinate of α^k. We define ρ_{ij} as the *density* of the matrix constructed with coordinates of $\alpha^{ij} \sim \alpha^{ij+m-1}$, as shown in (3) (i.e., the ratio between the number of 1s and the number of all entries) [7]. Then, the complexity of an α^{ij}-CFFM, i.e., a multiplier with α^{ij} as the multiplicator, is around $m \times (m-1) \times \rho_{ij}$ XOR gates according to (3).

To improve the decoding efficiency for the long BCH codes, multiple successive locations can be examined with parallel Chien search architectures. Fig. 2 depicts two conventional p-parallel Chien search architectures to shorten the operating cycles from N to $\lceil (N/p) \rceil$, where p is the parallel factor. Fig. 2(a) is the straightforward version from Fig. 1, whereas Fig. 2(b) is the direct-unfolded version with unfolded factor p [8], [9]. Both designs have $p \times t$ CFFMs and p $(t+1)$-input m-bit *finite-field adders (FFAs)*, resulting in a linear dependence of p for the hardware complexity. The directed-unfolded architecture, which utilizes α^i-CFFM to replace α^{ij}-CFFM for $j = 2 \sim p$, provides lower hardware complexity because density ρ_i is much smaller than ρ_{ij} for $i < m$. Nevertheless, the critical path in Fig. 2(b) is $(T_{\text{mux}} + p \times T_m + T_a)$, whereas that in Fig. 2(a) is only $(T_{\text{mux}} + T_m + T_a)$, where T_{mux}, T_m, and T_a represent the critical path of the multiplexer, the CFFM, and the FFA, respectively. The direct-unfolded architecture will lead to p times longer critical path if the CFFM dominates the delay path.

However, NAND Flash memory devices for next-generation applications require high-parallelism BCH decoders with large



$\Lambda(\alpha^{(1+p\tau)})$
α
α^2
α^t
Λ_0
$\Lambda(\alpha^{(p-1+p\tau)})$
$\alpha^{(p-1)}$
$\alpha^{2(p-1)}$
$\alpha^{t(p-1)}$
$\Lambda(\alpha^{(p+p\tau)})$
α^p
α^{2p}
α^{tp}
MUX
D
Λ_1
Λ_2
Λ_t
(a)
$\Lambda(\alpha^{(p+p\tau)})$
$\Lambda(\alpha^{(2+p\tau)})$
$\Lambda(\alpha^{(1+p\tau)})$
(b)

Fig. 2. Conventional p-parallel Chien search architectures. (a) Straightforward. (b) Direct unfolded.

error correcting capacity due to their degraded reliability and large page size. Either the high hardware complexity in Fig. 2(a) or the long critical path in Fig. 2(b) damages the area and the throughput performance of NAND Flash memory devices. In addition, the hardware complexity of Fig. 2(b) drastically increases in accordance with larger t. To enhance the performance under the requirements of NAND Flash memory devices, the MPCN-based architectures are provided in the next section.

III. Proposed Algorithms and Architectures for High-Parallelism BCH Decoders

The hardware complexity of the high-parallel Chien search architecture is dominated by numerous CFFMs. This section will reformulate the Chien search equation with minimal polynomials and utilize MPCNs for replacing the CFFMs in the Chien search architecture. In addition, the proposed MPCN-based Chien search architecture can merge the syndrome calculator with small overhead, leading to significant hardware complexity reduction.

A. Proposed Parallel Chien Search Scheme

To calculate $\Lambda_j \alpha^{ij}$ with minimal polynomials, the proposed new Chien search scheme defines an $m-1$ degree polynomial

$T_j(x) = t_{j,0} + t_{j,1}x^1 + \cdots + t_{j,m-1}x^{m-1}$, and the relation between Λ_j and $T_j(x)$ is defined as

$$\begin{aligned}\Lambda_j &= T_j(x)|_{x=\alpha^j}\\ &= t_{j,0}\alpha^0 + t_{j,1}\alpha^j + \cdots + t_{j,m-1}\alpha^{(m-1)j}\\ &= \lambda_{j,0}\alpha^0 + \lambda_{j,1}\alpha^1 + \cdots + \lambda_{j,m-1}\alpha^{m-1}. \end{aligned} \tag{4}$$

From (4), $\{\lambda_{j,0}, \lambda_{j,1}, \ldots, \lambda_{j,m-1}\}$ and $\{t_{j,0}, t_{j,1}, \ldots, t_{j,m-1}\}$ are viewed as coordinates with bases $\{\alpha^0, \alpha^1, \ldots, \alpha^{m-1}\}$ and $\{\alpha^0, \alpha^j, \ldots, \alpha^{(m-1)j}\}$, respectively, where

$$\begin{bmatrix}\lambda_{j,0}\\ \lambda_{j,1}\\ \vdots\\ \lambda_{j,m-1}\end{bmatrix} = \begin{bmatrix}\alpha_0^0 & \alpha_0^j & \cdots & \alpha_0^{(m-1)j}\\ \alpha_1^0 & \alpha_1^j & \cdots & \alpha_1^{(m-1)j}\\ \vdots & \vdots & \ddots & \vdots\\ \alpha_{m-1}^0 & \alpha_{m-1}^j & \cdots & \alpha_{m-1}^{(m-1)j}\end{bmatrix}\begin{bmatrix}t_{j,0}\\ t_{j,1}\\ \vdots\\ t_{j,m-1}\end{bmatrix}. \tag{5}$$

After the binary matrix operation in (5), Λ_j is represented with the basis from $\{\alpha^0, \alpha^j, \ldots, \alpha^{(m-1)j}\}$ to $\{\alpha^0, \alpha^1, \ldots, \alpha^{m-1}\}$; therefore, this operation is called as the jth *basis transformer (BT)* BT_j. As a result, the coefficients of $T_j(x)$ can be determined as

$$\begin{bmatrix}t_{j,0}\\ t_{j,1}\\ \vdots\\ t_{j,m-1}\end{bmatrix} = \begin{bmatrix}\alpha_0^0 & \alpha_0^j & \cdots & \alpha_0^{(m-1)j}\\ \alpha_1^0 & \alpha_1^j & \cdots & \alpha_1^{(m-1)j}\\ \vdots & \vdots & \ddots & \vdots\\ \alpha_{m-1}^0 & \alpha_{m-1}^j & \cdots & \alpha_{m-1}^{(m-1)j}\end{bmatrix}^{-1}\begin{bmatrix}\lambda_{j,0}\\ \lambda_{j,1}\\ \vdots\\ \lambda_{j,m-1}\end{bmatrix} \tag{6}$$

where the operation of the inverse matrix in (6) is called as the jth *inverse BT* IBT_j.

Based on our definitions, Λ_j can be represented in terms of $T_j(x)$, and (2) becomes

$$\begin{aligned}P_{ij} &= \Lambda_j\alpha^{ij}\\ &= T_j(x)|_{x=\alpha^j} \times (\alpha^j)^i\\ &= x^iT_j(x)|_{x=\alpha^j}\\ &= M_j(x) \times W_j(x) + D_j(x)|_{x=\alpha^j}\end{aligned} \tag{7}$$

where $M_j(x)$ is the minimal polynomial of α^j and $D_j(x)$ is the remainder polynomial resulting from dividing $x^iT_j(x)$ by $M_j(x)$. Since α^j is a root of $M_j(x)$, $D_j(\alpha^j)$ is the only nonzero term in (7). Then, the Chien search equation can be reformulated as

$$\Lambda(\alpha^i) = \sum_{j=1}^{t} P_{ij} + 1 = \sum_{j=1}^{t} D_j(\alpha^j) + 1. \tag{8}$$

As shown in (8), the Chien search can be simply realized by summing up all the evaluation results of first $\sim t$th BTs. Instead of executing summation after the basis transformations, the addition operation can be moved before the transformation, leading to fewer transformation operations.

Hence, (9) can be reformulated with the *group BT (GBT)* as

$$\begin{aligned}\Lambda(\alpha^i) &= \sum_{j=1}^{t}\sum_{k=0}^{m-1} d_{j,k}\alpha^{jk} + 1\\ &= \sum_{v=0}^{mt}\left(\sum_{\forall jk=v} d_{j,k}\right)\alpha^v + 1\end{aligned} \tag{9}$$

where $d_{j,k}$ is the kth coefficient of $D_j(x)$.

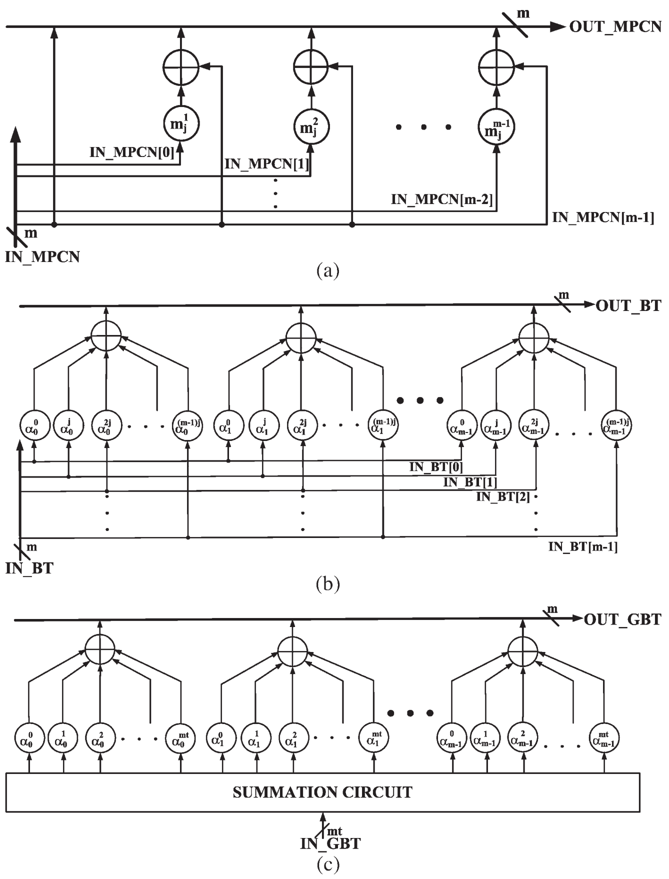


Fig. 3. Basic components in Chien search architecture. (a) MPCN_j. (b) BT_j. (c) GBT.

Fig. 3 shows the architectures of three basis components, including the jth MPCN, the jth BT, and the GBT. The jth MPCN MPCN_j shown in Fig. 3(a) executes modulo operation with divisor $M_j(x)$. It is constructed by the combinational circuit of the linear feedback shift register with the connection polynomial $M_j(x)$. Each binary element m_j^k in Fig. 3(a) is the kth coefficient of $M_j(x)$, indicating the wire connection. In the jth BT shown in Fig. 3(b), each α_l^k is a binary element as in (5) and can be represented whether the wire is connected or not. Fig. 3(c) illustrates the block diagram of the GBT. The additions are first executed with all the coefficients of $D_j(x)$ for $j = 1 \sim t$ (total mt bits), and the similar operations as a BT are applied with basis $\alpha^0 \sim \alpha^{mt}$.

In the proposed MPCN-based parallel-p Chien search architecture shown in Fig. 4, the coefficients of $\Lambda(x)$ are applied to the IBTs for transforming the operating basis. According to (7)–(9), the transformed values are evaluated with minimal polynomials for obtaining the Chien search results. All the multiplexers select the outputs of IBTs in the first cycle and then select the register data afterward. Searching from the $(N-1)$th to zeroth location, the proposed design checks p locations at each cycle. In each row, mt-bit data are fed into a GBT to examine the error locations. An error is found at the $(N + r - p(\tau + 1) - 1)$th location if the output of the rth-row GBT is equal to zero at the τth cycle. In contrast with that in Fig. 2, our proposed Chien search architecture utilizes $p \times t$ MPCNs to replace $p \times t$ CFFMs. Notice that the

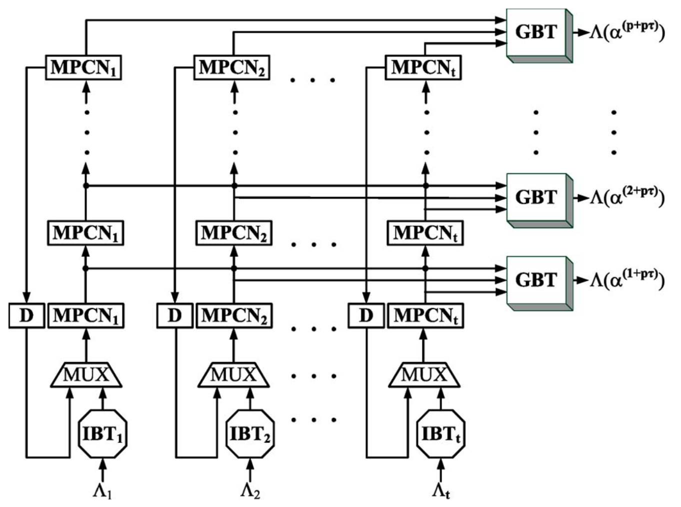


Fig. 4. MPCN-based parallel-p Chien search architecture.

XOR gate count requirement of one MPCN is at most $m-1$, which is much smaller than that of one CFFM. Therefore, it is area efficient to apply the MPCNs, particularly in the large parallelism conditions.

B. Joint Syndrome Calculator and Chien Search Architecture

The proposed MPCN-based architecture can merge the syndrome calculator and the Chien search in the same hardware with small overhead. In the BCH decoding process, received polynomial $R(x)$ is fed into the syndrome calculator to generate syndrome polynomial $S(x) = S_1 + S_2x^1 + \cdots + S_{2t}x^{2t-1}$, which is expressed as

$$\begin{aligned} S_j &= R(x)|_{x=\alpha^j} \\ &= M_j(x) \times Q_j(x) + B_j(x)|_{x=\alpha^j} \\ &= B_j(\alpha^j) \end{aligned} \tag{10}$$

where $B_j(x)$ is the remainder polynomial resulting from dividing $R(x)$ by $M_j(x)$. Consequently, the jth syndrome value can be calculated with $M_j(x)$.

Fig. 5 illustrates our parallel-p joint syndrome calculator and Chien search with the MPCN-based architecture. The syndrome calculator and Chien search phases are determined by the SEL signal. When the SEL signal is high, the jth syndrome value is formulated as

$$\begin{aligned} S_j = \Big(\big(\big((R_{N-1}x^{p-1} + \cdots + R_{N-p-1}) \bmod M_j(x)\big)\, x^p \\ +(R_{N-p-2}x^{p-1} + \cdots + R_{N-2p-1})\big) \bmod M_j(x)\big)x^p \\ + \cdots \big)x^p + R_{p-1}x^{p-1} + \cdots + R_0\Big) \bmod M_j(x)|_{x=\alpha^j} \end{aligned} \tag{11}$$

The partial remainder stored in the register is multiplied by x^p and accumulated with the received symbols. After all the received symbols are processed, BT$_j$ transforms the accumulated result to the jth syndrome value. In contrast with Fig. 4, t BTs are applied instead of one GBT in the first row to evaluate individual syndrome value. Note that the FFA in Fig. 5 is only a 1-bit operation because each coefficient of $R(x)$ is a binary value. Therefore, except for the difference between the BT and the GBT, the overhead of the supporting syndrome calculation is only p NAND and $p \times t$ XOR gates.

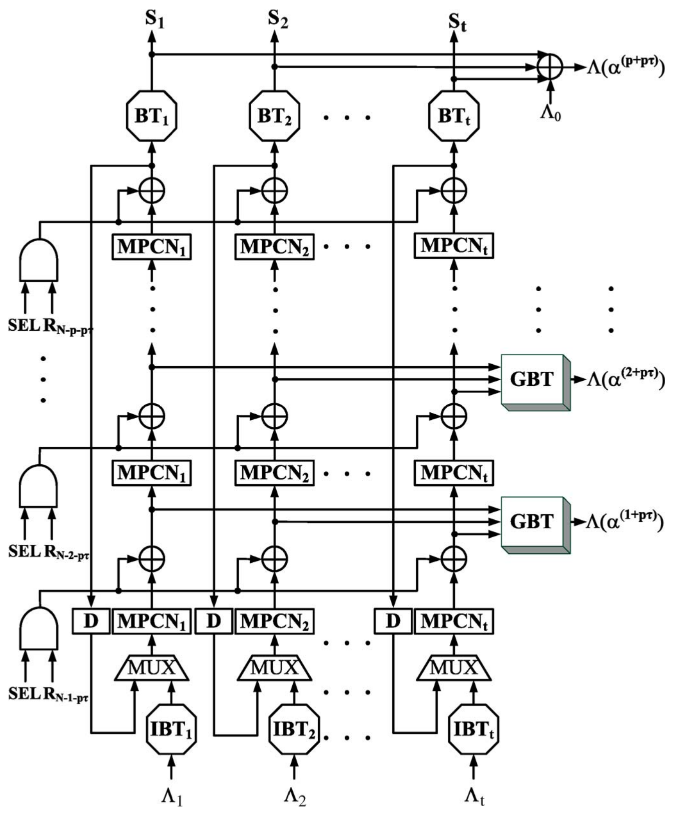


Fig. 5. Parallel-p joint syndrome calculator and Chien search with MPCN-based architecture.

IV. EXPERIMENTAL RESULTS

This section demonstrates the synthesis results for our joint syndrome and Chien search with the MPCN-based architecture in a 90-nm CMOS technology. Table I demonstrates the comparison results among the straightforward, direct-unfolded, GMA-optimized, strength-reduced, and our MPCN-based methods for the parallel-32 BCH (4603, 4096; 39) code, which is the shortened code of BCH (8191, 7683; 39). Moreover, our proposal can be reconfigured for the syndrome calculation and the Chien search. Except for the direct-unfolded design, all other designs can reach a 133-MHz operating frequency for NAND Flash applications. Aside from that, if higher operating frequency is required in the future applications, the critical path can be shortened with folding techniques. Notice that both the GMA-optimized and strength-reduced designs in Table I have the Chien search and the syndrome calculator separately because CFFMs with common multiplicands or with sharable modulo-$f(x)$ computations will not concurrently exist in these two blocks. For example, the GMA-optimized approach can be applied in Fig. 2(a) to share the computation of CFFMs in each column since these CFFMs have common multiplicands during the Chien search phase. In the syndrome phase, however, the CFFMs with common multiplicands are in each row. As a result, an extra syndrome calculator is required to support syndrome values evaluation.

According to the synthesis optimization of the straightforward design and the requirement of an extra syndrome calculator, the gate count savings of GMA-optimized and strength-reduced designs are less than that in [6] and [7]. The

TABLE I
SYNTHESIS RESULTS FOR THE JOINT SYNDROME CALCULATOR AND CHIEN SEARCH IN THE PARALLEL-32 BCH (4603, 4096; 39) CODE

	Straight-Forward [8]	Direct-Unfolded [9]	GMA-Optimized [6]	Strength-Reduced [7]	Proposed MPCN-Based
Reconfiguration	YES	YES	NO *	NO *	YES
Technology	90 nm	90 nm	90 nm	90 nm	90 nm
Operating Frequency	133 MHz	33 MHz	133 MHz	133 MHz	133 MHz
Power Consumption	44.7mW @133MHz	30.4mW @33MHz	32.6mW @133MHz	23.7mW @133MHz	32.6mW @133MHz
Latency (Clock Cycle)	144	144	144	144	144
Gate Count	192.4 K	143 K	130.4 K / 114.5 K **	144.9 K / 129K **	102.6 K
Gate Count saving over [8]	-	25.7%	32.3% / 40.5% **	24.7% / 33% **	46.7%

* An extra syndrome calculator is required to support syndrome values evaluation.
** By using the property, $S_{2i} = S_i^2$, to calculate even syndrome values.

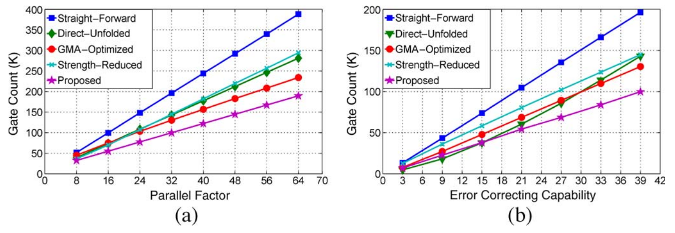


Fig. 6. Complexity analysis of joint syndrome calculator and Chien search for parallel BCH (N, 4096; t) decoder. (a) BCH (4603, 4096; 39) decoder with different parallel factors. (b) Parallel-32 BCH (N, 4096, t) decoder with different error correcting capacity.

straightforward design has a gate count of 158.2 K with 21.4% overhead for supporting syndrome calculation. However, our proposed MPCN-based Chien search has a 97.8-K gate count and only requires 4.9% overhead for the syndrome phase support. As a result, the joint syndrome calculator and Chien search with the MPCN-based architecture can save 46.7% gate count, as compared with the straightforward design.

Fig. 6 illustrates the effect of parallel factor p and error correcting capacity t to the hardware complexity in the joint syndrome calculator and Chien search among five designs. The BCH (4603, 4096; 39) decoder with p that ranged from 8 to 64 is analyzed in Fig. 6(a). The gate count of each design is proportional to p due to the almost unchanged average density ρ_{avg} of CFFMs (or MPCNs) with different p. However, the proposed design has the smallest area because more MPCNs take place of CFFMs with larger p, leading to the smoothest slope (Δcomplexity/Δp). In Fig. 6(b), the parallel-32 BCH decoder with a 4096-bit information length is discussed under t that ranged from 3 to 39. The direct-unfolded design only utilizes $\alpha^1 \sim \alpha^t$-CFFMs, whose average density ρ_{avg} is quite small (in the vicinity of $1/m$) while $t \leq m-1$. Therefore, the complexity of the direct-unfolded design is the smallest while $t \leq 15$ and drastically increases in accordance with larger t. The slope of our proposed design is the smoothest, indicating higher gate count saving in accordance with increasing t. As a result, after replacing CFFMs with MPCNs, as well as applying the joint syndrome calculator and Chien search architecture, our proposed design can provide the lowest hardware complexity to meet both the high-parallelism and error-correcting-capacity requirements for NAND Flash memory devices.

V. CONCLUSION

This brief has provided a novel MPCN-based parallel architecture in long BCH decoders for NAND Flash memory devices. Unlike previous approaches performing CFFMs calculations, the proposed design has exploited MPCNs to improve the hardware efficiency since the XOR gate count requirement of one MPCN is at most $m-1$, whereas that of one CFFM is usually proportional to m^2. The proposed MPCN-based architecture can merge the syndrome calculator and the Chien search with small hardware overhead. In contrast with the straightforward design for the parallel-32 BCH (4603, 4096; 39) decoder, the proposed joint syndrome calculator and Chien search has 46.7% gate count saving according to the synthesis results in the 90-nm CMOS technology.

ACKNOWLEDGMENT

The authors would like to thank Prof. C.-C. Chung for the electronic design automation tool assistance and the United Microelectronics Corporation and the Chip Implementation Center for providing standard CMOS technology data and computer-aided design tools.

REFERENCES

[1] N. Shibata, H. Maejima, K. Isobe, K. Iwasa, M. Nakagawa, M. Fujiu, T. Shimizu, M. Honma, S. Hoshi, T. Kawaai, K. Kanebako, S. Yoshikawa, H. Tabata, A. Inoue, T. Takahashi, T. Shano, Y. Komatsu, K. Nagaba, M. Kosakai, N. Motohashi, K. Kanazawa, K. Imamiya, H. Nakai, M. Lasser, M. Murin, A. Meir, A. Eyal, and M. Shlick, "A 70 nm 16 Gb 16-level-Cell NAND Flash memory," *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 929–937, Apr. 2008.

[2] H. Kim, J. hoon Park, K.-T. Park, P. Kwak, O. Kwon, C. Kim, Y. Lee, S. Park, K. Kim, D. Cho, J. Lee, J. Song, S. Lee, H. Yoo, S. Kim, S. Yu, S. Kim, S. Lee, K. Kyung, Y.-H. Lim, and C. Chung, "A 159 mm^2 32 nm 32 Gb MLC NAND-Flash memory with 200 MB/s asynchronous DDR interface," in *Proc. IEEE ISSCC*, Feb. 2010, pp. 442–443.

[3] C. R. Baugh and B. A. Wooley, *Theory and Practice of Error Control Codes*. Reading, MA: Addison-Wesley, 1983.

[4] H. Choi, W. Liu, and W. Sung, "VLSI implementation of BCH error correction for multilevel cell NAND Flash memory," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 5, pp. 843–847, May 2010.

[5] S. Li and T. Zhang, "Improving multi-level NAND Flash memory storage reliability using concatenated BCH-TCM coding," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 10, pp. 1412–1420, Oct. 2010.

[6] Y. Chen and K. Parhi, "Small area parallel Chien search architectures for long BCH codes," *IEEE Trans. VLSI Syst.*, vol. 12, no. 5, pp. 545–549, May 2004.

[7] J. Cho and W. Sung, "Strength-reduced parallel Chien search architecture for strong BCH codes," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 5, pp. 427–431, May 2008.

[8] H.-C. Chang, C.-C. Lin, and C.-Y. Lee, "A low-power Reed–Solomon decoder for STM-16 optical communications," in *Proc. IEEE APASIC*, 2002, pp. 351–354.

[9] L. Song, M.-L. Yu, and M. Shaffer, "10- and 40-Gb/s forward error correction devices for optical communications," *IEEE J. Solid-State Circuits*, vol. 37, no. 11, pp. 1565–1573, Nov. 2002.

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