Low-Complexity Triple-Error-Correcting Parallel BCH Decoder

Jaewoong Yeon, Seung-Jun Yang, Cheolho Kim, and Hanho Lee

***Abstract*—This paper presents a low-complexity triple-error-correcting parallel Bose–Chaudhuri–Hocquenghem (BCH) decoder architecture and its efficient design techniques. A novel modified step-by-step (m-SBS) decoding algorithm, which significantly reduces computational complexity, is proposed for the parallel BCH decoder. In addition, a determinant calculator and a error locator are proposed to reduce hardware complexity. Specifically, a sharing syndrome factor calculator and a self-error detection scheme are proposed. The multi-channel multi-parallel BCH decoder using the proposed m-SBS algorithm and design techniques have considerably less hardware complexity and latency than those using a conventional algorithms. For a 16-channel 4-parallel (1020, 990) BCH decoder over $GF(2^{12})$, the proposed design can lead to a reduction in complexity of at least 23 % compared to conventional architectures.**

***Index Terms*—BCH code, modified step-by-step decoding, triple-error correction, decoder, forward error correction**

I. INTRODUCTION

Bose–Chaudhuri–Hocquenghem (BCH) codes are important multiple-error-correcting cyclic codes that are widely used in communications and storage systems [1, 2]. The most widely used decoding method for BCH codes is the Berlekamp–Massey (BM) algorithm [2]. The decoding procedure of the BM algorithm consists of the following three major steps: First, for a t-error-correcting BCH code and symbols from the Galois-field $GF(2^m)$, calculate $2t$ syndrome values S_i (i=1, 2, ..., $2t$) from the received codeword. Second, determine the error locator polynomial $\sigma(x)$ from syndrome values S_i. Third, find the roots of the error locator polynomial, which are error locators, and finally correct the errors. The BM algorithm can be used to solve a key equation for an error locator polynomial $\lambda(x)$ of the BCH decoding procedure.

Another design method for BCH codes, which has a decoding procedure entirely different from the BM algorithm, is the Peterson algorithm [3] and the step-by-step (SBS) decoding algorithm [4-6]. The decoding procedure of the SBS decoding algorithm also consists of three steps, as follows: First, calculate the syndrome values S_i (i=1, 2, ..., $2t$) from the received codeword. Second, construct t-syndrome matrices, which have dimensions from 1×1 to $t \times t$. The number of errors in the received codeword can be determined from the determinants of the t-syndrome matrices. Third, temporarily change the received codeword and use the determinants of the temporarily changed syndrome matrices to test whether the number of errors is reduced. If the number of errors is reduced, the error location can be determined and the errors can be corrected. The SBS decoding algorithm corrects the errors directly in each position of a received codeword without computing the error locator polynomial.

To speed up the SBS decoding process, a fully-parallel architecture is required for high-speed data processing. Specifically, a parallel decoding algorithm is of considerable importance particularly for high-speed

Manuscript received Apr. 21, 2013; accepted Jul. 18, 2013
Dept. of Information and Communication Engr. Inha University, Incheon, 402-751, Korea
E-mail : hhlee@inha.ac.kr

applications, *e.g.*, optical communications and storage systems. Parallel decoding of BCH codes using the SBS decoding algorithm has been proposed in [7] and [8]. However, because determinants have to be calculated for every temporarily changed syndrome matrix, the computational complexity of the SBS decoding algorithm can be very high. This is the main weakness of the SBS decoding algorithm, which makes it less competitive than the BM algorithm [6]. To reduce the complexity of the SBS decoding algorithm, previous studies [5] and [6] have proposed a low-complexity SBS decoding algorithm to simplify the decoding procedure and increase the decoding speed. However, for every received symbol, at least one determinant of a $v \times v$ matrix, where v is the number of errors, still has to be calculated. When v or parallel factor p is large, the decoding complexity is still very high. In other words, a parallel BCH decoder using the conventional SBS decoding algorithm has considerably higher hardware complexity than that using the BM algorithm. Previous studies [3, 9] have shown that iterative BCH codes, which use triple-error-correcting BCH codes, can achieve better performance than other BCH or Reed–Solomon (RS) codes [10]. However, to the best of our knowledge, an efficient parallel BCH decoder architecture and its design techniques based on the SBS decoding algorithm have not been reported previously.

This paper proposes a novel modified SBS (m-SBS) decoding algorithm that can be used to significantly reduce computational complexity, as well as an efficient parallel triple-error-correcting BCH decoder architecture. Specifically, a modified determinant calculation algorithm and a self-error-detection scheme are proposed. In addition, a sharing syndrome factor calculator and self-error detection scheme are proposed. To compare the hardware complexity of the multi-channel multi-parallel BCH decoder architectures, 16-channel 4-parallel (1020, 990) BCH decoders using the proposed decoding algorithm are implemented and compared with previous architectures. For the purpose of architecture synthesis analysis, our design can achieve at least 23% hardware complexity reduction compared to 100 Gbps (1020, 990) BCH decoder using conventional Peterson algorithm.

II. MODIFIED STEP-BY-STEP DECODING ALGORITHM

Consider a t-error-correcting (n, k) BCH code over $GF(2^m)$ with length $n = 2^m - 1$, where $m \geq 3$. Then, the generator polynomial over $GF(2^m)$ has roots over $2t$ consecutive-field elements $\alpha, \alpha^2, \alpha^3, ..., \alpha^{2t}$, where α is a primitive element of $GF(2^m)$.

Suppose that $c(x)$ is the transmitted codeword polynomial, and $r(x) = c(x) + e(x) = r_0 + r_1x + ... + r_{n-1}x^{n-1}$ is the received polynomial, where $e(x) = e_0 + e_1x + ... + e_{n-1}x^{n-1}$ is the error polynomial. For a triple-error-correcting BCH code, the syndrome polynomial $S(x) = S_1 + S_2x + ... + S_6x^5$ can be calculated from $r(x)$ with $S_i(x) = r(\alpha^i) = e(\alpha^i)$, $i=1, 2, ..., 6$. For the BCH code, $S_{2i} = S_i^2$ and only half the syndromes need to be computed.

For parallel decoding using the conventional SBS decoding algorithm, the determinants of all n temporarily changed syndrome matrices need to be calculated simultaneously. Therefore, the temporarily changed syndrome $S_{i,p}$, which is obtained from changing the position of the j-th bit in the equation $r(x)$, can be calculated by intentionally adding the errors in syndrome, as follows:

$$S_{i,j} = S_i + \alpha^{ij}, \qquad i=1, 3, 5 \qquad 0 \leq j \leq n-1 \quad (1)$$

The principle of the conventional SBS decoding algorithm is that it involves changing the received bits one at a time to test whether the number of errors is reduced. For the triple-error-correcting BCH code, the 3×3 syndrome matrix is defined as follows:

$$L = \begin{bmatrix} S_1 & 1 & 0 \\ S_3 & S_2 & S_1 \\ S_5 & S_4 & S_3 \end{bmatrix} \quad (2)$$

The SBS decoding algorithm performs the decoding by checking the received codeword and the weight change of the error polynomial. In other words, the relationship between the syndrome and the weight of the error polynomial must first be determined, as stated in [4].

The SBS algorithm-based BCH decoder consists of four components: syndrome calculator (SC), determinant calculator (DC), error locator (EL) and first-in first-out

(FIFO) [7]. The SC generates syndromes S_i ($i = 1, 2, 3, ..., 2t$), and the temporarily changed syndromes $S_{i,j}$ ($i = 1, 2, 3, ..., 2t$; $0 \le j \le n-1$) are used in the DC. The DC can be used to calculate the determinants $det(L_t)$, $1 \le j \le t$ and then determine the number of errors. The EL is used to find the error position. If the corresponding bit is judged to be an erroneous bit, the decoder sends a correcting bit $e_p = 1$ to change its magnitude. In addition, FIFO is used to buffer the received vector.

The conventional SBS decoding algorithm requires the determinant values, which are calculated using the syndrome and the temporarily changed syndrome values. These two determinants can be compared to perform the decoding. However, as the parallel factor increases, the hardware complexity of the DC block will increase significantly. For example, (p + 1) DC blocks are required for parallel factor p.

Thus, we propose a novel m-SBS decoding algorithm that reduces the hardware complexity for a highly parallel structure. The proposed m-SBS decoding algorithm is described as follows:

Modified Step-By-Step Decoding Algorithm (t = 3)

Input: S_i, ($i = 1, 3, 5$)

Start determinant calculation:

Sharing syndrome factor computation:

$S_1^3 + S_3$

$S_1^4 + S_1S_3$

$S_5 + S_1^2S_3$

$S_1^6 + S_3^2 + S_1^3S_3 + S_1S_5$

Determinant decision:

$R = A\alpha^j + B\alpha^{2j} + C\alpha^{3j}$

if $(S_1^3 + S_3 == 0)$

begin

$R = S_1^3 + S_3$

$A = S_1^2$, $B = S_1$, $C = 0$

for $j = n-1$ *until* $j=0$ *do*

begin

if $(S_1^3 + S_3 = S_1^2\alpha^j + S_1\alpha^{2j})$

$H_j = 0$

else

$H_j = 1$

end

end

else

begin

$R = S_1^6 + S_3^2 + S_1^3S_3 + S_1S_5$

$A = S_5 + S_1^2S_3$, $B = S_1^4 + S_1S_3$, $C = S_1^3 + S_3$

for $j = n-1$ *until* $j=0$ *do*

begin

if $((S_1^6 + S_3^2 + S_1^3S_3 + S_1S_5) =$

$\alpha^j (S_5 + S_1^2S_3) + \alpha^{2j} (S_1^4 + S_1S_3) + \alpha^{3j} (S_1^3 + S_3))$

$H_j = 0$

else

$H_j = 1$

end

end

Initially, we can determine from the syndrome whether the codeword is erroneous. If all syndromes S_i are zero, then the received polynomial $r(x)$ is a valid codeword $c(x)$ in which no errors have occurred. Otherwise, the number of errors and the location of errors should be determined. If there is an error, the two cases are executed independently depending on the number of errors in the codeword. In the first case, there is a single error in the number of errors in the codeword. Otherwise, the number of errors in the codeword is two or more. After finding the number of errors, the determinant decision Eq. (4) is solved.

$$R = A\alpha^j + B\alpha^{2j} + C\alpha^{3j} \tag{4}$$

In Eq. (4), the values of A, B, C, and R are determined by the computation of the sharing syndrome factor (SSF), and then it can be determined whether the errors are corrected.

First, we verify whether one or more errors have occurred in the codeword from the equation $det(A) = S_1^3 + S_3$. If $S_1^3 = S_3$, a single error has occurred. However, if $S_1^3 \neq S_3$, two or more errors have occurred. If the number of error is only one in the codeword, we can find the location of the error by calculating Eq. (5). Eq. (5) can be expressed as Eq. (6) and then be expanded, simplified, and transformed to Eq. (7).

Thus, if $S_1^3 + S_3 = S_1^2\alpha^j + S_1\alpha^{2j}$, the j-th location is erroneous. If $S_1^3 + S_3 \neq S_1^2\alpha^j + S_1\alpha^{2j}$, there is no error in the j-th location of the codeword. The elements of $GF(2^m)$ are substituted into Eq. (5) according to the order of j.

$$(S_1 + \alpha^j)^3 + (S_3 + \alpha^{3j})\ 0 \le j \le n-1 \tag{5}$$

$$(S_1^3 + S_3) + S_1^2\alpha^j + S_1\alpha^{2j} = 0 \tag{6}$$

$$(S_1^3 + S_3) = S_1^2\alpha^j + S_1\alpha^{2j} \tag{7}$$

In Eq. (7), temporarily changed syndromes are

substituted into the equation directly, and the result of Eq. (7) corresponds to that of Eq. (5).

If the number of errors in the codeword is two or more, the errors can be corrected by calculating Eq. (8). If the result of Eq. (8) is zero, the *j*-th location of the codeword is an error. If the result of Eq. (8) is not zero, then the *j*-th location of the codeword is not erroneous. Eq. (8) can be expressed as Eq. (9) and then expanded, simplified, and transformed to Eq. (10). Thus, we can find the location of the error in the codeword by calculating Eq. (10).

$$S_{1,j}^{\,6} + S_{1,j}^{\,3}S_{3,j} + S_{1,j}S_{5,j} + S_{3,j} = (S_{1,j}^{\,3} + S_{3,j})^2 + S_{1,j}S_{5,j} + S_{1,j}^{\,3}S_{3,j} \quad (8)$$

$$(S_1^{\,6} + S_3^{\,2} + S_1^{\,3}S_3 + S_1S_5) + \alpha^{3j}(S_1^{\,3} + S_3) + \alpha^{2j}(S_1^{\,4} + S_1S_3) + \alpha^{j}(S_5 + S_1^{\,2}S_3) = 0 \quad (9)$$

$$S_1^{\,6} + S_3^{\,2} + S_1^{\,3}S_3 + S_1S_5 = \alpha^{3j}(S_1^{\,3} + S_3) + \alpha^{2j}(S_1^{\,4} + S_1S_3) + \alpha^{j}(S_5 + S_1^{\,2}S_3) \quad (10)$$

The decoding scheme using Eqs. (7) and (10) is more efficient than that using Eqs. (5) and (8), because the complexity of the involved computations is minimized in the former. Eqs. (7) and (10) comprise constant terms and variable terms. Constant terms are calculated once in the decoding process and can be shared by the parallel factor. The remainder variable terms of the equation can be implemented by the constant *GF* multiplier and *GF* adder.

As mentioned above, the determinant decision equation can cover two cases: 1) the occurrence of one error, and 2) the occurrence of two or more errors. Thus, the values of *A*, *B*, *C*, and *R* in Eq. (4) can be selected from the error pattern and then be substituted into Eq. (4). Thus, errors can be corrected using Eq. (4).

III. Proposed Parallel BCH Decoder Architecture

The BCH decoder using the proposed m-SBS algorithm consists of the following three blocks: SC, DC, and EL, as shown in Fig. 1.

1. Syndrome Calculator

The SC calculates all the syndromes S_i $(1 \le i \le 2t-1)$ by inserting the roots of generator polynomial $g(x)$ into the received polynomial $r(x)$. As mentioned, S_i $(i = 1, 3, 5)$ are required in triple-error-correcting m-SBS algorithm-based BCH decoding to calculate the syndrome. Because

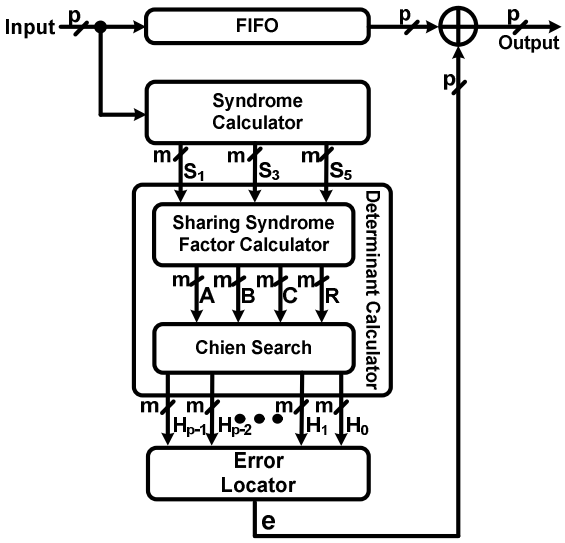


Fig. 1. Proposed triple-error-correcting parallel BCH decoder using modified step-by-step algorithm.

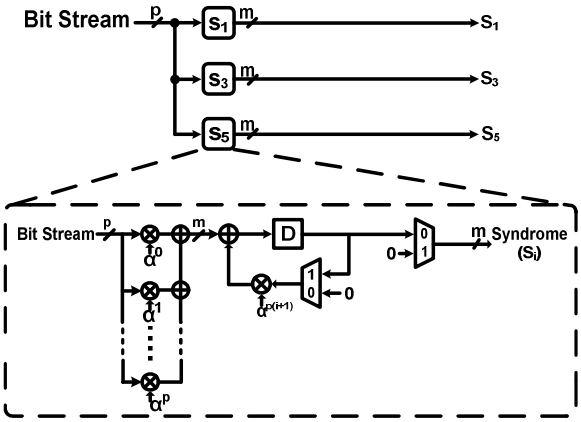


Fig. 2. Parallel syndrome calculator in $GF(2^{10})$.

the SC receives the parallelized codeword, a parallel SC cell is required, as shown in Fig. 2.

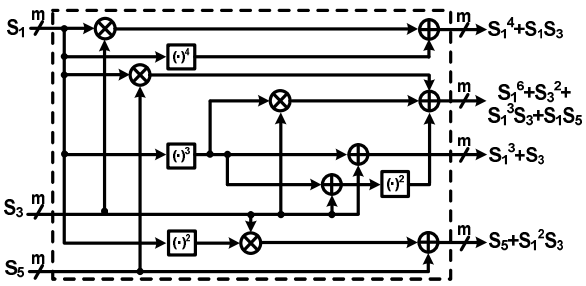
2. Determinant Calculator

The DC calculates the determinant value using the syndrome value to check whether the bit position is erroneous. In the conventional decoding method, a single DC block is needed for one-bit processing, because the DC can check only a single bit position in one clock cycle. The parallel architecture requires an independent matrix calculation block depending on parallel factors to calculate the determinant value of a multi-bit position. The hardware complexity of the DC is significantly higher than that of other blocks, which results in very

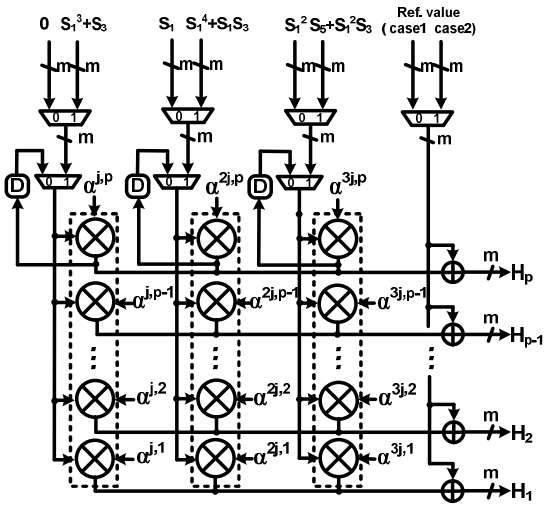
high hardware complexity for the parallel BCH decoder. Thus, minimizing the hardware complexity of the DC is critical in order to reduce the hardware complexity of the parallel BCH decoder.

We present an area-efficient parallel DC architecture using a sharing technique. The proposed DC consists of the sharing syndrome factor calculator (SSFC) block and the Chien search (CS) block, as shown in Fig. 3. In the SSFC block, the SSFs are calculated using the syndrome values. Because the SSFs are constant terms, only one clock cycle latency is required in the decoding process. Thus, when applying this scheme to the multi-channel multi-parallel BCH decoder, the SSFC block is not dependent on the parallel factor, and it can be shared as much as possible. Therefore, this scheme exhibits more area efficiency and less latency than the conventional method for the parallel decoding process.

The CS block calculates the ***H*** value that has the information of error location in the codeword. The values of sharing syndrome factors *A*, *B*, *C*, and *R*, which are outputs of the SSFC block, are fed to the CS block. The CS block checks whether error has occurred. If the output of the CS block is zero, the ***H*** value is zero. Otherwise, the ***H*** value is non-zero. The proposed parallel CS block consists of constant *GF* multipliers, *GF* adders, multiplexers, and D-FFs. The constant *GF* multipliers have common inputs that can be merged into one block, and then the common factors among constant *GF* multipliers can be shared using the iterative matching algorithm (IMA) [11]. In this way, hardware complexity can be significantly reduced for the parallel CS block. Thus, the proposed DC block shows better hardware efficiency as the parallel factor is increased.



(a)



(b)

Fig. 3. Determinant calculator (a) sharing syndrome factor calculator, (b) parallel Chien search block using the sharing constant multiplier.

3. Error Locator with Self-Error Detection

The EL checks the error location and corrects errors according to ***H*** values. The *m*-bit ***H*** value can be transformed to a one-bit value by bitwise OR. If the ***H*** value is non-zero, it is one. Otherwise, it is zero. In the proposed method, two types of ***H*** values are required to check the error location and correct errors. The first type of ***H*** value is the reference value that has information about the number of errors in the codeword. The second type of ***H*** value is compared with the reference ***H*** value to check whether the bit location is erroneous. The proposed self-error-detection method is not necessary to calculate the reference ***H*** value. That is, the reference ***H*** value can be obtained by itself, because the number of errors in the codeword can be detected by observing the first to fourth ***H*** values using the comparator. Fig. 4 shows the EL with self-error detection.

IV. Results and Comparison

The proposed parallel (1020, 990) BCH decoder

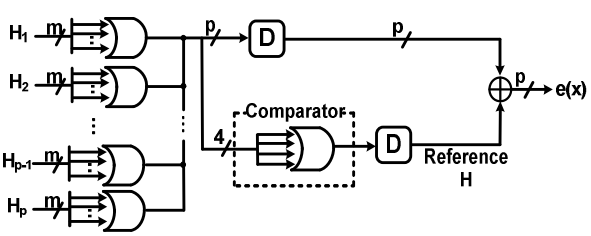


Fig. 4. Error locator with self-error detection.

architecture using the m-SBS algorithm was modeled in the Verilog HDL and then simulated to verify its functionality using a test pattern generated from a C simulator. After complete verification of the design functionality, the proposed architecture was synthesized using appropriate time and area constraints. Both the synthesis and simulation steps were executed using a SYNOPSYS design tool and 90-nm CMOS technology.

Table 1 shows the hardware complexity comparison for the proposed triple-error-correcting (1020, 990) BCH decoder architecture using the m-SBS algorithm, conventional SBS algorithm, and simplified inverse-free BM (SiBM) algorithm. Because all BCH decoders have the same SC, the number of operators for SC is excluded in Table 1. The conventional SBS algorithm-based BCH decoder is not efficient for parallel SBS BCH decoding, because the number of complex operation units is increased significantly depending on the parallel factor. Compared with the BM algorithm-based BCH decoder [12], the proposed architecture has a considerably fewer number of variable *GF* multipliers, *i.e.*, only four, regardless of the parallel factor. As mentioned, the complexity of constant *GF* multipliers can be reduced by IMA-based optimization. Thus, the proposed m-SBS algorithm-based BCH decoder has substantially less hardware complexity than conventional BCH decoders do. Fig. 5 shows the effect of the parallel factor on complexity by comparing three different BCH decoders. As the parallel factor increases, the proposed BCH decoder significantly reduces the hardware complexity because the proposed sharing technique covers the SSF and CS blocks.

Table 2 shows the performance comparison of the 4-parallel BCH decoder architectures using the proposed m-SBS algorithm and the conventional SiBM algorithm. The architectures have been implemented using 90-nm CMOS technology. The proposed m-SBS algorithm-based BCH decoder architecture has 1,310 XOR gate counts excluding the FIFO from the synthesized result, which shows 58% less hardware complexity than the SiBM algorithm-based BCH decoder. The proposed BCH decoder operates at a clock rate of 400 MHz, has a latency of 258 clocks, and a throughput of 1.6 Gb/s.

To compare the hardware complexity of the multi-channel multi-parallel BCH decoder architecture, we implemented a 16-channel 4-parallel (1020, 990) BCH decoder using the m-SBS algorithm, as shown in Fig. 6.

Table 1. Hardware complexity comparison for triple-error-correcting (1020, 990) BCH decoders (excluding syndrome calculator)

	Proposed m-SBS	Conventional SBS [8]	SiBM [12]
Adder	*3p+2*	*6p*	*6+(3-1)p*
Variable mult.	*4*	*2p*	*12*
Constant mult.	*3p*	-	*3p*
$(\cdot)^2$ operator	*4*	*p*	*3*
$(\cdot)^3$ operator	*1*	*p*	-

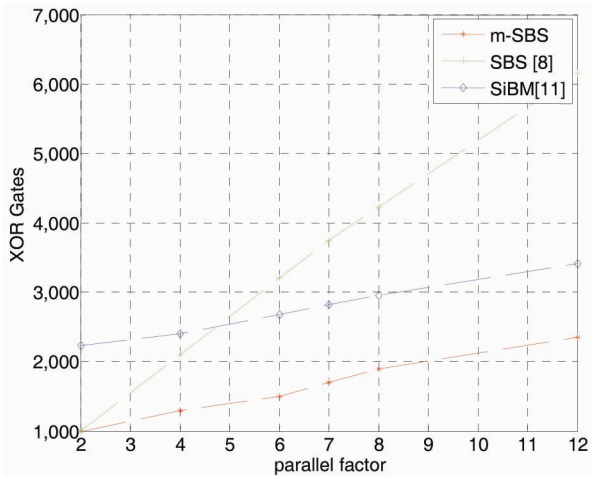


Fig. 5. Complexity of t=3 BCH decoder versus the parallel factor for the (1020, 990) BCH code.

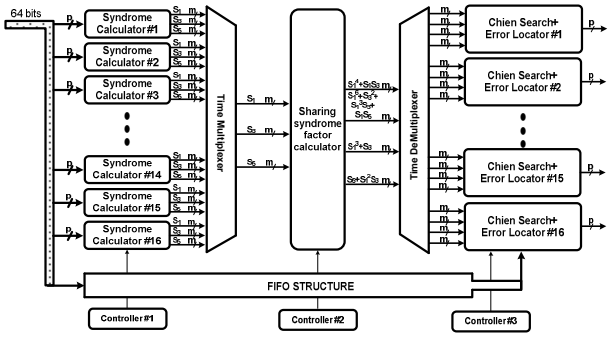


Fig. 6. 16-channel 4-parallel (1020, 990) BCH decoder.

Table 2. Performance comparison of the 4-parallel (1020, 990) BCH decoders using M-SBS algorithm and SiBM algorithm

	Proposed m-SBS	SiBM [11]
CMOS technology	*90-nm*	*90-nm*
Gate counts (XOR)	*1,310*	*3,130*
Latency (Clocks)	*258*	*256 + t*
Clock rate (MHz)	*400*	*400*
Throughput (Gb/s)	*1.6*	*1.6*

Table 3. Performance comparison of 16-channel 4-parallel (1020, 990) BCH decoders

	Proposed m-SBS		Peterson [3]		ECRiBM [13]	
CMOS technology	*90-nm*		*90-nm*		*90-nm*	
Gate count (XOR)	Synd.	*11,499*	Synd.	*30,660*	Synd.	*30,660*
	SSFC	*573*	Err.loc. comp.	*984*	KES	*6,804*
	CS	*23,049*	Root comp.	*13,790*	CS	*112,530*
Total gate count (XOR)	*35,121*		*45,434*		*149,994*	
Latency (Clocks)	*251 + 16*		-		-	
Max. clock rate (MHz)	*400*		*250*		*250*	
Throughput (Gb/s)	*102.4*		*109*		*102.4*	

Table 3 shows the performance comparison of 16-channel 4-parallel BCH decoder architectures using the m-SBS, Peterson, and ECRiBM algorithms. The proposed 16-channel 4-parallel (1020, 990) BCH decoder operates at a clock rate of 400 MHz, has a latency of 267 clocks, and a throughput of 102.4 Gb/s. The proposed architecture shows 23% less hardware complexity than the Peterson algorithm-based BCH decoder for 100 Gb/s data processing.

V. CONCLUSIONS

This paper presented a novel low-complexity triple-error-correcting parallel BCH decoder architecture and its efficient design techniques. A hardware-friendly m-SBS decoding algorithm was proposed and adopted for a multi-channel multi-parallel BCH decoder. In addition, a novel DC with a SSFC and an EL with self-error detection are shown to reduce hardware complexity. The proposed multi-channel multi-parallel BCH decoder has significantly less hardware complexity than those using conventional algorithms.

ACKNOWLEDGMENTS

This research was partly supported by the IT R&D program of MOTIE/KEIT [10044092] and Basic Science Research Program through the NRF funded by the Ministry of Education, Science and Technology (2012R1A1A2007740).

REFERENCES

[1] S. Lin and D. Costello, *Error Control Coding: Fundamentals and Applications*, Englewood Cliffs, NJ: Prentice-Hall, 1983.

[2] K. Lee, H.-G. Kang, J.-I. Park, H. Lee, "A high-speed low-complexity concatenated BCH decoder architecture for 100Gb/s optical communications," *Jour. OF SIGNAL PROCESSING SYSTEMS*, vol. 6, no. 1, pp. 43-55, Jan. 2012.

[3] X. Zhang, Z. Wang, "A Low-Complexity Three-Error-Correcting BCH Decoder for Optical Transport Network," *IEEE Trans. on Circuits and Systems*, vol. 59, no. 10, pp. 663-667, Oct. 2012.

[4] J. Massey, "Step-by-step decoding of the Bose-Chaudhuri-Hocquenghem codes," *IEEE Trans. Inf. Theory*, vol. 11, no. 4, pp. 580-585, Apr.1965.

[5] C.-L. Chr, S.-L. Su, and S.-W. Wu, "A Low-Complexity Step-By-Step Decoding Algorithm for Binary BCH Codes," *IEICE Trans. Fundamentals*, vol.E88-A, no.1, pp. 359-365, Jan. 2005.

[6] X. Liu, C. Lu, T. H. Cheng, and S. N. Koh, "A Simplified Step-by-Step Decoding Algorithm for Parallel Decoding of Reed-Solomon Codes," *IEEE Tran. on Communications*, vol. 55, no. 6, pp. 1103-1109, Jun. 2007.

[7] S.-W. Wei, C.-H. Wei, "High-speed hardware decoder for double-error-correcting binary BCH codes," *Proc. of IEE Commun*., vol. 136, pp. 227-231, Jun. 1989.

[8] T. Hwang, "Parallel decoding of binary BCH codes," *Electron. Lett*., vol. 7, no. 24, pp. 2223-2225, 1991.

[9] J. Justesen, K. J. Larsen, and L. A. Pederson, "Error correcting correction for 100G transport networks," *IEEE Commun. Mag*., vol. 48, no. 3, pp. S48-S55, Mar. 2010.

[10] J.-I. Park, K. Lee, C.-S. Choi, H. Lee, "High-Speed Low-Complexity Reed-Solomon Decoder using Pipelined Berlekamp-Massey algorithm and Its Folded Architecture," *Journal of Semiconductor Technology and Science*, vol. 10, no. 3, pp. 193-202, Sep. 2010.

[11] M. Potkonjak, M. B. Srivastava, and A. P. Chandrakasan, "Multiple constant multiplications: Efficient and versatile framework and algorithms for exploring common subexpression elimination,"

IEEE Trans. Computer-Aided Design Integr. Circuits Syst., vol. 15, no. 2, pp. 151–165, Feb. 1996.

[12] W. Liu, J. Rho, and W. Sung, "Low-Power High-Throughput BCH Error Correction VLSI Design for Multi-Level Cell NAND Flash Memories," *IEEE Workshop on Signal Processing Systems Design and Implementation (SIPS)*, pp. 303-308, Oct. 2006.

[13] S. Yoon, H. Lee, K. Lee, "High-Speed Two-Parallel Concatenated BCH-based Super-FEC Architecture for Optical Communications," *IEICE Trans. on Fundamentals of Electronics, Communications, and Computer Sciences, Systems*, vol. E92-A, no. 4, pp.769-777, Apr. 2010.

Jaewoong Yeon received B.S and M. S. degrees, both in Information & Communication engineering from Inha University, Incheon, Korea, in 2011 and 2013, respectively. Since January 2013, he has been with Chief Technology Office, LG Electronics, where he is currently Research engineer. His research interests VLSI architecture design for digital signal processing and communications.

Seung-Jun Yang received B.S degree in Information & Communication engineering from SangMyung University in 2011 and M.S. degree in information and communication engineering from Inha University, Incheon, Korea, in 2013, respectively. Since January 2013, he has been with Chief Technology Office, LG Electronics, where he is currently Research engineer. His research interests VLSI architecture design for digital signal processing and communications.

Cheolho Kim received the B.S. degree in Information & Communication Engineering in 2012, from Inha University, Incheon, Korea, where he is currently working toward the M.S degree. His interests include not only digital VLSI circuits and systems design for communications such as forward error correction codes but also their efficient hardware imple menttation.

Hanho Lee received Ph.D. and M.S. degrees, both in Electrical & Computer Engineering, from the University of Minnesota, Minneapolis, USA, in 2000 and 1996, respectively. In 1999, he was a Member of Technical Staff-1 at Lucent Technologies, Bell Labs, Holmdel, New Jersey. From April 2000 to August 2002, he was a Member of Technical Staff at the Lucent Technologies (Bell Labs Innovations), Allentown, where he was responsible for the development of high-performance DSP multi-processor architecture. From August 2002 to August 2004, he was an Assistant Professor at the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, USA. Since August 2004, he has been with the Department of Information and Communication Engineering, Inha University, where he is currently Professor. He was a visiting researcher at Electronics and Telecommunications Research Institute (ETRI), Korea, in 2005. From August 2010 to August 2011, he was a visiting scholar at Bell Labs, Alcatel-Lucent, Murray Hill, New Jersey, USA. He authored over 120 papers and also holds 16 patents. He received the best paper award at the International SoC Design Conference (ISOCC) in 2006 and 2009. He served the Secretary General of the IEEE ISCAS2012, Special Session Chair of the ISOCC2012 and Guest Editor of the Journal of Electrical and Computer Engineering. He has served the Technical Committee Member, IEEE Signal Processing Society, Design and Implementation of Signal Processing Systems (DISPS) (Jan. 2011 ~ Dec. 2013). He is a Technical Committee Member, 2013 IEEE Workshop on Signal Processing Systems (SiPS2013). His research interest includes VLSI architecture design for digital signal processing and forward error correction code.