JOURNAL OF SEMICONDUCTOR TECHNOLOGY AND SCIENCE, VOL.16, NO.4, AUGUST, 2016
http://dx.doi.org/10.5573/JSTS.2016.16.4.488
ISSN(Print) 1598-1657
ISSN(Online) 2233-4866

Low-Complexity Non-Iterative Soft-Decision BCH Decoder Architecture for WBAN Applications

Boseok Jung, Taesung Kim, and Hanho Lee

Abstract—**This paper presents a low-complexity non-iterative soft-decision Bose-Chaudhuri-Hocquenghem (SD-BCH) decoder architecture and design technique for wireless body area networks (WBANs). A SD-BCH decoder with test syndrome computation, a syndrome calculator, Chien search and metric check, and error location decision is proposed. The proposed SD-BCH decoder not only uses test syndromes, but also does not have an iteration process. The proposed SD-BCH decoder provides a 0.75~1 dB coding gain compared to a hard-decision BCH (HD-BCH) decoder, and almost similar coding gain compared to a conventional SD-BCH decoder. The proposed SD-BCH (63, 51) decoder was designed and implemented using 90-nm CMOS standard cell technology. Synthesis results show that the proposed non-iterative SD-BCH decoder using a serial structure can lead to a 75% reduction in hardware complexity and a clock speed 3.8 times faster than a conventional SD-BCH decoder.**

Index Terms—BCH codes, soft-decision decoding, decoder, low-complexity, WBAN

I. INTRODUCTION

Recently, wireless body area network (WBAN) technology has drawn wide attention owing to the demand for short-range wireless communication and the advent of energy-efficient VLSI technology. The WBAN [1] is a special purpose sensor network designed to operate autonomously to connect various medical sensors and appliances, located inside and outside of a human body. It consists of small, intelligent devices attached on or implanted in the body, which are capable of establishing a wireless communication link.

Bose-Chaudhuri-Hocquenghem (BCH) codes are important multiple-error-correcting cyclic codes that are widely used in communications and storage systems [2]-[4]. In the WBAN [1], the BCH (63, 51) code and its shortened (31, 19) code are adopted to enhance transmission reliability under different channel conditions. Either hard-decision BCH (HD-BCH) decoders or soft-decision BCH (SD-BCH) decoders can be adopted at the receiver. In general, a SD-BCH decoder can achieve better error performance compared to a HD-BCH decoder, and the improvement in bit error rate (BER) performance of the SD-BCH decoder can translate into power savings at the transmitter, given the same data link requirements.

In general, there are several methods for SD-BCH codes. Maximum likelihood decoding (MLD) [5], generalized minimum distance (GMD) [6], sliding encoding-window (SEW) [7], and Chase algorithms [8] were developed to produce a list of candidate codeword. In several soft-decoding methods, a Chase algorithm can be used to correct errors with a hard-decision kernel. Generally, a SD-BCH decoder with a Chase algorithm has an iteration process and also requires a test pattern generator.

Yang et al. [9] found that a SD-BCH decoder for energy-constrained WBANs provided a 1dB coding gain, compared to a HD-BCH decoder. In order to reduce the energy dissipation and area, Yang et al. [9] presented

Manuscript received Oct. 27, 2015; accepted Mar. 14, 2016
Dept. of Information and Communication Engr. Inha University, Incheon, 22212, Korea
E-mail : hhlee@inha.ac.kr

early termination (ET), probabilistic sorting and pass-transistor logic–based Chien search. An early termination scheme was used to reduce the number of unnecessary test patterns. A probabilistic sorting scheme is utilized to reduce the architectural complexity of the sorting circuit. For the hard-decision kernel, Yang et al. used a Peterson algorithm, which uses the determinant value.

In this paper, we propose a non-iterative SD-BCH decoding algorithm and efficient decoder architecture without iteration. Specifically, the SD-BCH decoder has test syndrome computation (TSC), which eliminates the test pattern generator, a hard-decision kernel based on a modified step-by-step (m-SBS) algorithm [4], and error location decision (ELD). Although the SD-BCH decoder generally uses an iteration method, the proposed SD-BCH decoder does not require iteration. Moreover, in order to reduce hardware complexity, optimized test syndrome computation, a syndrome factor calculator, and error location decision are also presented.

The rest of this paper is organized as follows. In Section II, SD-BCH decoding algorithm is briefly described. Section III describes the proposed non-iterative SD-BCH decoding algorithm and provides analysis of a BER simulation. Section IV presents the proposed SD-BCH decoder architecture and design techniques. In Section V, the results and a performance comparison are presented. Finally, a conclusion is provided in Section VI.

II. OVERVIEW OF SOFT-DECISION BCH DECODING ALGORITHM

In this section, we introduce the basics of SD-BCH (n, k, t) decoding algorithm, n and k denote the codeword length and the information length, respectively, and t denotes the error correcting capability. Type II Chase-based SD-BCH decoding algorithm has been discussed in [8, 9]. The Chase-II algorithm is a sub-optimum soft-decision algorithm that uses an error correction only hard-decision decoding (HDD) as the kernel.

The procedure for the Chase-II algorithm is described as follows:

1) Find the locations of the least reliable bits (LRBs), where $p = [d_{min}/2]$, and d_{min} is the minimum Hamming distance of the codeword.

2) Generate test patterns by considering all combinations of the LRBs.

3) Decode each test pattern by using the HDD kernel. If this test pattern is decoded successfully using the HDD kernel, the decoded word is regarded as a candidate codeword.

4) Evaluate the Euclidean distance for each candidate codeword in the list and select the one with the smallest Euclidean distance as the best decision codeword.

In fact, it is unnecessary for the HDD kernel in the Chase algorithm to process all test patterns since redundant test patterns can be eliminated to save energy.

In [9, 10], the authors proposed a test pattern generator to correct more errors than HD-BCH decoder. This idea comes from adding LRBs in a codeword, which is passed a hard-decision unit. The procedure for the conventional SD-BCH decoding using test-pattern generator is described as follows:

1) Generate a candidate codeword from a received codeword $r(x) = r_0 x^0 + r_1 x^1 + r_2 x^2 + \ldots + r_{n-1} x^{n-1}$, where codeword length is n.

2) Assume that LRBs locations are x^{t0} and x^{t1}, test pattern polynomials, which generated in test pattern generator, are:

$$tp_1(x) = r_0 x^0 + r_1 x^1 + r_2 x^2 + \ldots + r_{n-1} x^{n-1} \quad \text{for test pattern 1,}$$

$$tp_2(x) = r_0 x^0 + r_1 x^1 + r_2 x^2 + \ldots + r_{n-1} x^{n-1} + r_{t0} x^{t0} = tp_1(x) + r_{t0} x^{t0} \quad \text{for test pattern 2,}$$

$$tp_3(x) = r_0 x^0 + r_1 x^1 + r_2 x^2 + \ldots + r_{n-1} x^{n-1} + r_{t0} x^{t0} + r_{t1} x^{t1} = tp_2(x) + r_{t1} x^{t1} \quad \text{for test pattern 3,}$$

$$tp_4(x) = r_0 x^0 + r_1 x^1 + r_2 x^2 + \ldots + r_{n-1} x^{n-1} + r_{t1} x^{t1} = tp_3(x) + r_{t0} x^{t0} \quad \text{for test pattern 4,}$$

where p is 2, $r_i (0 \le i \le n-1)$, $r_{ti} (0 \le i \le n-1)$, r_{t0} and r_{t1} is in $GF(2)$. By definition, the error pattern is $e(x) = r(x) - v(x) = e_{j1} x^{j1} + e_{j2} x^{j2} + e_{j3} x^{j3} + \ldots + e_{jv} x^{jv}$, where $0 \le j1 < j2 < \ldots < jv \le n-1$ and contained errors is v. So, from equation $e(x)$, error patterns are described as follows:

$$e_{tp1}(x) = x^{j1} + x^{j2} + x^{j3} + \ldots + x^{jv} \quad \text{for error pattern 1,}$$

$$e_{tp2}(x) = x^{j1} + x^{j2} + x^{j3} + \ldots + x^{jv} + x^{t0} = e_{tp1}(x) + x^{t0} \quad \text{for error pattern 2,}$$

$$e_{tp3}(x) = x^{j1} + x^{j2} + x^{j3} + \ldots + x^{jv} + x^{t0} + x^{t1} = e_{tp2}(x) + x^{t1} \quad \text{for error pattern 3,}$$

$$e_{tp4}(x) = x^{j1} + x^{j2} + x^{j3} + \ldots + x^{jv} + x^{t1} = e_{tp3}(x) + x^{t0} \quad \text{for error pattern 4,}$$

where $0 \le t0 \le n\text{-}1$ and $0 \le t1 \le n\text{-}1$. Syndromes can be calculated using the error patterns. To calculate syndrome S_i, α^i substitute for x in $e(x)$. Thus syndrome is $S_i = (\alpha^{j1})^i + (\alpha^{j2})^i + (\alpha^{j3})^i + \ldots + (\alpha^{jv})^i$. Syndromes of test patterns can be calculated as

Syndromes of test pattern 1:
$S_{i,tp1} = (\alpha^{j1})^i + (\alpha^{j2})^i + \ldots + (\alpha^{jv})^i$,
Syndromes of test pattern 2:
$S_{i,tp2} = (\alpha^{j1})^i + (\alpha^{j2})^i + \ldots + (\alpha^{jv})^i + (\alpha^{t0})^i$,
Syndromes of test pattern 3:
$S_{i,tp3} = (\alpha^{j1})^i + (\alpha^{j2})^i + \ldots + (\alpha^{jv})^i + (\alpha^{t0})^i + (\alpha^{t1})^i$,
Syndromes of test pattern 4:
$S_{i,tp4} = (\alpha^{j1})^i + (\alpha^{j2})^i + \ldots + (\alpha^{jv})^i + (\alpha^{t1})^i$,

where $1 \le i \le 2t$; α^{j1}, α^{j2}, α^{j3}, … , α^{jv} are elements of $GF(2^n)$ and unknown value, but α^{t0} and α^{t1} are already known value. Syndromes can be calculated whenever test pattern is generated.

III. Proposed Non-Iterative Soft-Decision BCH Decoding Algorithm

Suppose an HD-BCH (n, k, t) code has an n-bit codeword, a k-bit message and t correctable bits using algebraic decoders, operation under a Galois-field $GF(2^m)$, and that α is the primitive root over the primitive polynomial $f(x)$. Otherwise, with a SD-BCH code based on a Chase algorithm [8], it is possible that the number of corrected errors is $t + p$, which determines the position of the p least reliable bits (LRBs), where $p \le d_{min}/2$. This requires a test pattern generator and an error-correction-only hard-decision decoder (HDD) as the kernel. Chase algorithm–based soft-decision decoders (SDDs) evaluate the soft-decision metric using the HDD kernel. However, in case of BCH (63, 51, 2) code and $p = 2$, we only need a test syndrome α^{t0}, α^{t1} without a test pattern generator. The major steps for the proposed non-iterative SD-BCH (63, 51, 2) decoding are described as follows:

1) Separate the hard-decision value and magnitude in the log likelihood ratio (LLR) input bits.

2) Compute syndromes S_1, S_3 from the syndrome calculator (SC) and test syndromes α^{t0}, α^{t1} from TSC.

3) Calculate syndrome factors R_{HD}, A_{HD}, B_{HD}, R_{tp2}, A_{tp2}, B_{tp2}, R_{tp3}, A_{tp3}, B_{tp3}, R_{tp4}, A_{tp4}, B_{tp4} with syndromes and test syndromes.

4) Find the H values from a Chien search (CS), and calculate the metric value M from the H values.

5) Decide the decision codeword d from the controller and successfully correct the codeword.

In addition, the proposed non-iterative SD-BCH decoding algorithm is described in detail as follows:

Proposed Non-Iterative Soft-Decision BCH (63, 51, 2) Decoding

1) Separate hard-decision and magnitude
Input: $r_i = \{r_{i,0}, r_{i,1}, r_{i,2}, r_{i,3}\}$ ($i = n\text{-}1, \ldots, 0$)
$r_{HD,i} = r_{i,0}$, $m_i = |r_i|$

2) Syndrome Calculator and Test Syndrome Computation
Syndrome Calculator:
$S_1 = \sum r_{HD,i} \cdot \alpha^i$, $S_3 = \sum r_{HD,i} \cdot \alpha^{3 \cdot i}$ ($i = n\text{-}1, \ldots, 0$)
Test Syndrome Computation:
Initial : $\mathbf{min_0} = \{1,1,1\}$, $\mathbf{min_1} = \{1,1,1\}$

for $i = n\text{-}1$ ***until*** $i = (n\text{-}1)/2+1$ ***begin***
  if ($m_i < \mathbf{min_0}$) ***then*** $\mathbf{min_0} = m_i$, $\alpha^{t0} = \alpha^i$
end
for $i = (n\text{-}1)/2$ ***until*** $i = 0$ ***begin***
  if ($m_i < \mathbf{min_1}$) ***then*** $\mathbf{min_1} = m_i$, $\alpha^{t1} = \alpha^i$
end

3) Syndrome Factor Calculator
Input: S_1, S_3, α^{t0}, α^{t1}
Hard-decision:
$R_{HD} = S_1^3 + S_3$, $A_{HD} = S_1^2$, $B_{HD} = S_1$
Test Patterns:
$R_{tp2} = R_{HD} + A_{HD} (\alpha^{t0})^2 + B_{HD} \alpha^{t0}$
$A_{tp2} = A_{HD} + (\alpha^{t0})^2$, $B_{tp2} = B_{HD} + \alpha^{t0}$
$R_{tp3} = R_{tp2} + A_{tp2} (\alpha^{t1})^2 + B_{tp2} \alpha^{t1}$
$A_{tp3} = A_{tp2} + (\alpha^{t1})^2$, $B_{tp3} = B_{tp2} + \alpha^{t1}$
$R_{tp4} = R_{HD} + A_{HD} \cdot (\alpha^{t1})^2 + B_{HD} \alpha^{t1}$
$A_{tp4} = A_{HD} + (\alpha^{t1})^2$, $B_{tp4} = B_{HD} + \alpha^{t1}$

4) Chien Search and Metric Check
Input: R_j, A_j, B_j, (j = HD, tp2, tp3, tp4)
Chien Search:
for j = HD ***until*** tp4 ***begin***
  if $R_j = A_j \cdot \alpha^i + B_j \cdot \alpha^{2i}$ ***then*** $H_{j,i} = 1$
  else $H_{j,i} = 0$ ($i = n\text{-}1, \ldots, 0$)
end
Metric Check:
for j = HD ***until*** tp4 ***begin***
  $M_j = \sum H_{j,i}$ ($i = n\text{-}1, \ldots, 0$)
end

5) Error Location Decision
Input : H_{HD}, H_{tp2}, H_{tp3}, H_{tp4}, α^{t0}, α^{t1}
for $i = n\text{-}1$ ***until*** $i = 0$ ***begin***
  if $(\alpha^{t0} + \alpha^i = 0)$ $H_{t0} = 1$ ***else*** $H\alpha^{t0} = 0$
  if $(\alpha^{t1} + \alpha^i = 0)$ $H_{t1} = 1$ ***else*** $H\alpha^{t1} = 0$

  if ($ctr = 0$) $d_i = H_{HD}$
  else if ($ctr = 1$) $d_i = H_{tp2} + H_{t0}$
  else if ($ctr = 2$) $d_i = H_{tp2} + H_{t0} + H_{t1}$
  else $d_i = H_{tp2} + H_{t1}$
end

Initially, we can separate hard-decision $r_{HD,i}$ with magnitude $|r_i|$ ($i = n$-1, …, 0) from the received LLR. The hard-decision $r_{HD,i}$, which is one bit, is a most significant bit of the LLR and is fed into the hard-

decision kernel to calculate the syndromes. On the other hand, magnitude $|r_i|$ is used to find a test syndrome value in test syndrome computation.

Yang et al. [9] demonstrated a probabilistic sorting scheme to have a correct second minimum value with a high probability. The second minimum value is generated from the last second stages of the tree. On the other hand, in the proposed method, two minimum values in serial processing can search the test syndrome value to divide the total clock cycle into halves. Therefore, α^{t0} is possibly a syndrome value, which is one value out of $\{\alpha^{n-1}, \ldots, \alpha^{(n-1)/2+1}\}$, indicating a location of the minimum magnitude value from $|r_{n-1}|$ to $|r_{(n-1)/2+1}|$. Also, α^{t1} is determined to be a syndrome value out of $\{\alpha^{(n-1)/2}, \ldots, \alpha^{0}\}$, and $\boldsymbol{min}_0$ and $\boldsymbol{min}_1$ store the minimum value of magnitude to compare with the next magnitude value. Initially, $\boldsymbol{min}_0$ and $\boldsymbol{min}_1$ are both {1, 1, 1} to be compared with $|r_i|$, where $i = n-1, \ldots, 0$.

After computing syndromes and test syndromes, the values of A, B, and R as syndrome factors are calculated by the syndrome factor calculator. In the m-SBS algorithm [4], the syndrome factors are determined from syndrome values S_1 and S_3 as follows:

$$\left(S_1^3 + S_3\right) = S_1^2\alpha^j + S_1\alpha^{2j} \quad (0 \le j \le n-1) \tag{1}$$

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

In Eqs. (1, 2), the values of A, B, and R are determined to be $A = S_1^2$, $B = S_1$, $R = S_1^3 + S_3$ and correct errors from the Chien search. Therefore, syndrome factors A_{HD}, B_{HD}, and R_{HD} of the hard decision in the proposed SD-BCH decoder are the same values as A, B, and R. However, the syndrome factors of test patterns are different from each other, because test syndromes should be added to the syndromes (S_1, S_3). The syndromes of test patterns can be reformulated as follows:

Syndrome of test pattern 1:

$$S_{i,tp1} = (\alpha^{j1})^i + (\alpha^{j2})^i + \ldots + (\alpha^{jv})^i,$$

Syndrome of test pattern 2:

$$S_{i,tp2} = (\alpha^{j1})^i + (\alpha^{j2})^i + \ldots + (\alpha^{jv})^i + (\alpha^{t0})^i = S_{i,tp1} + (\alpha^{t0})^i,$$

Syndrome of test pattern 3:

$$S_{i,tp3} = (\alpha^{j1})^i + (\alpha^{j2})^i + \ldots + (\alpha^{jv})^i + (\alpha^{t0})^i + (\alpha^{t1})^i = S_{i,tp1} + (\alpha^{t0})^i + (\alpha^{t1})^i,$$

Syndrome of test pattern 4:

$$S_{i,tp4} = (\alpha^{j1})^i + (\alpha^{j2})^i + \ldots + (\alpha^{jv})^i + (\alpha^{t1})^i = S_{i,tp1} + (\alpha^{t1})^i,$$

That is, in S_1 and S_3, tp2 syndrome factors include the value of α^{t0}, and tp3 syndrome factors include the values of α^{t0} and α^{t1}. Finally, tp4 syndrome factors include the value of α^{t1}. In the case of *tp2,* by adding α^{t0}, the values of A_{tp2}, B_{tp2}, and R_{tp2} are calculated as follows:

$$\begin{aligned} R_{tp2} &= (S_1 + \alpha^{t0})^3 + (S_3 + (\alpha^{t0})^3) \\ &= \left(S_1^3 + S_3\right) + S_1^2 \cdot \alpha^{t0} + S_1 \cdot \alpha^{2\cdot t0} \\ &= R_{HD} + A_{HD} \cdot \alpha^{t0} + B_{HD} \cdot \alpha^{2\cdot t0} \end{aligned} \tag{3}$$

$$A_{tp2} = (S_1 + \alpha^{t0})^2 = S_1^2 + \alpha^{2\cdot t0} = A_{HD} + \alpha^{2\cdot t0} \tag{4}$$

$$B_{tp2} = S_1 + \alpha^{t0} = B_{HD} + \alpha^{t0} \tag{5}$$

where the $GF(2^m)$ adder element is implemented using an exclusive or (XOR) operation. Since the test syndrome values indicate the locations of LRBs, the syndrome factors of test patterns can be determined by adding test syndromes in Eqs. (3-5). For example, syndrome S_1 is changed to the value of S_1 by adding test syndrome α^{t0}, and syndrome S_3 is changed to the value of S_3 by adding test syndrome α^{3t0}. Thus, the syndrome factors of the second test pattern tp2 include syndrome factors of the hard decision and test syndrome α^{t0}. The syndrome factors of the fourth test pattern tp4 include R_{HD}, A_{HD}, B_{HD}, and test syndrome α^{t1}. On the other hand, the syndrome factors of third test pattern tp3 are different. For *tp3*, the values of A_{tp3}, B_{tp3}, and R_{tp3} can be calculated as

$$\begin{aligned} R_{tp3} &= (S_1 + \alpha^{t0} + \alpha^{t1})^3 + (S_3 + \alpha^{3\cdot t0} + \alpha^{3\cdot t1}) \\ &= (S_1^3 + S_3 + S_1^2 \cdot \alpha^{t0} + S_1 \cdot \alpha^{2\cdot t0}) \\ &\quad + (S_1^2 + \alpha^{2\cdot t0}) \cdot \alpha^{t1} + (S_1 + \alpha^{t0}) \cdot \alpha 2 \cdot t1 \\ &= R_{tp2} + A_{tp2} \cdot \alpha^{t1} + B_{tp2} \cdot \alpha^{2\cdot t1} \end{aligned} \tag{6}$$

$$\begin{aligned} A_{tp3} &= (S_1 + \alpha^{t0} + \alpha^{t1})^2 = (S_1^2 + \alpha^{2\cdot t0}) + \alpha^{2\cdot t1} \\ &= A_{tp2} + \alpha^{2\cdot t1} \end{aligned} \tag{7}$$

$$B_{tp3} = (S_1 + \alpha^{t0}) + \alpha^{t1} = B_{tp2} + \alpha^{t1} \tag{8}$$

In Eqs. (6-8), the syndrome factors of tp3 are arranged by the syndrome factor of tp2 and α^{t1}. Consequentially,

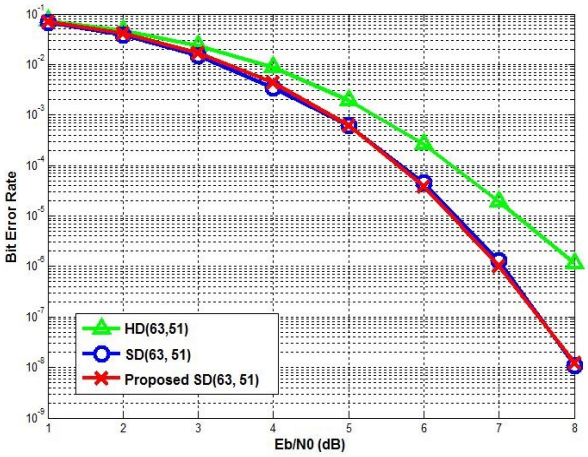

Fig. 1. BER performance for the proposed SD-BCH (63, 51) decoder, a conventional SD-BCH (63, 51) decoder, and a HD-BCH (63, 51) decoder.

the equations of syndrome factors are simple and can make an efficient area of the syndrome factor calculator (SFC) block. The Chien search calculates the H value that has the information about error location in each codeword of the hard decision and test pattern.

In Eq. (2), if $R + A\alpha^j + B\alpha^{2j} = 0$, where $j = n\text{-}1, \ldots, 0$, the H value is 1. If $R + A\alpha^j + B\alpha^{2j} \neq 0$, the H value is zero. While the H value is calculated, the metric check computes the M value, which is the number of errors, by adding the H value. Finally, in the error location decision, the decision codeword d is set to 1 between the hard decision and test patterns by the controller.

Fig. 1 shows the BER performance comparison for the proposed (63, 51) SD-BCH decoder, a conventional SD-BCH decoder [9] and a HD-BCH decoder. Additive white Gaussian noise (AWGN) channel and BPSK are considered. The proposed SD-BCH decoder provides a 0.75~1dB coding gain (BER = 10^{-5}) compared to the HD-BCH decoder, and almost similar coding gain compared to a conventional SD-BCH decoder.

IV. Proposed Non-iterative Soft-Decision BCH Decoder Architecture

The proposed SD-BCH decoder architecture consists of four major units: TSC, hard-decision kernel, ELD, and controller, as shown in Fig. 2. The hard-decision kernel consists of a syndrome calculator (SC), a syndrome factor calculator (SFC), and Chien search (CS) and metric check (MC).

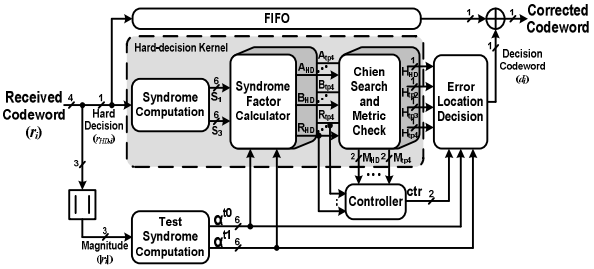

Fig. 2. Proposed non-iterative SD-BCH decoder architecture.

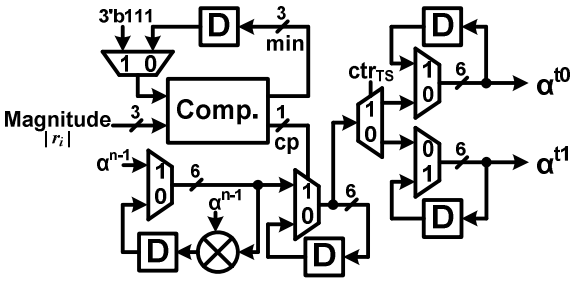

Fig. 3. Test syndrome computation.

1. Test Syndrome Computation

In this subsection, we discuss the detailed hardware architecture of the proposed TSC architecture using a loop circuit. Fig. 3 shows the proposed TSC block. Since the test pattern generator requires a lot of resisters, the proposed TSC architecture only uses the test syndrome values by the loop circuit generating a syndrome value without the location of $\min_1$, $\min_2$. The proposed TSC block generates test syndromes α^{t0}, α^{t1} by inserting the magnitude of the received LLR. The TSC block needs control signal ctr_{TS} and comparison signal cp to get test syndrome α^{t0} and α^{t1}; ctr_{TS} is 1, while magnitude is inserted from $|r_{n-1}|$ to $|r_{(n-1)/2}|$. If cp is 1, the inserted magnitude value is smaller than the previous magnitude value. Then, test syndrome α^{t0} is updated to a syndrome value that indicates the location of the smallest magnitude. If cp=0, the stored magnitude value is smaller than the inserted magnitude value, and α^{t0} holdsthe previous value.

2. Hard-Decision Kernel

The hard-decision kernel uses a HD-BCH decoding structure based on an m-SBS algorithm [4]. It consists of the SC block, the SFC block, and the CS and MC block, as shown in Fig. 2.

A. Syndrome calculator

The SC block calculates all the syndromes S_i *(1 ≤ i ≤ 2t-1)* by inserting a hard-decision bit.

As mentioned, S_i *(i=1, 3)* is required in m-SBS algorithm-based HD-BCH decoding [4] to calculate the syndrome. The SC consists of a parallel SC cell to receive the parallelized codeword, as shown in Fig. 4(a).

B. Syndrome Factor Calculator

Fig. 4(b) shows the detailed hardware architecture of the proposed SFC architecture. The SFC block calculates the syndrome factors, which are R, A, and B using the syndrome values S_1, S_3 from the SC block, and the test syndrome values α^{t0}, α^{t1} from the TSC block. The SFC block consists of *GF* multipliers and *GF* adders. Therefore, the architecture of the SFC block is comparatively simple and has an efficient area, because it remains unaffected by a serial or parallel structure.

C. Chien Search and Metric Check

Fig. 4(c) shows the CS and MC block, which consists of four parallel blocks because the proposed SD-BCH decoder operates without iteration. The values of the syndrome factor, which are outputs of the SFC block, are

fed to the CS block. The function of a NOR gate replaces the syndrome value with the H value using Eq. (2). The output H values can be calculated with the CS block to search the error location information. The CS block checks whether an error has occurred. If $R + A\alpha^j + B\alpha^{2j} = 0$, then the H value is 1; that is, an error has occurred in the j^{th} location. Otherwise, the H value is zero; that is, no error has occurred. The CS block consists of constant *GF* multipliers, *GF* adders, multiplexers, NOR gates, and D flip-flops. In the MC block, metric value M, which is the number of errors, is computed by adding the H values during 63 clock cycles. For example, if the number of errors is two, the M value becomes 2.

3. Error Location Decision

The ELD block consists of a constant GF multiplier, GF adders, multiplexers, NOR gates and D-FFs, as shown in Fig. 5. The ELD block 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 NOR. If the H value is non-zero, otherwise, it is zero. In

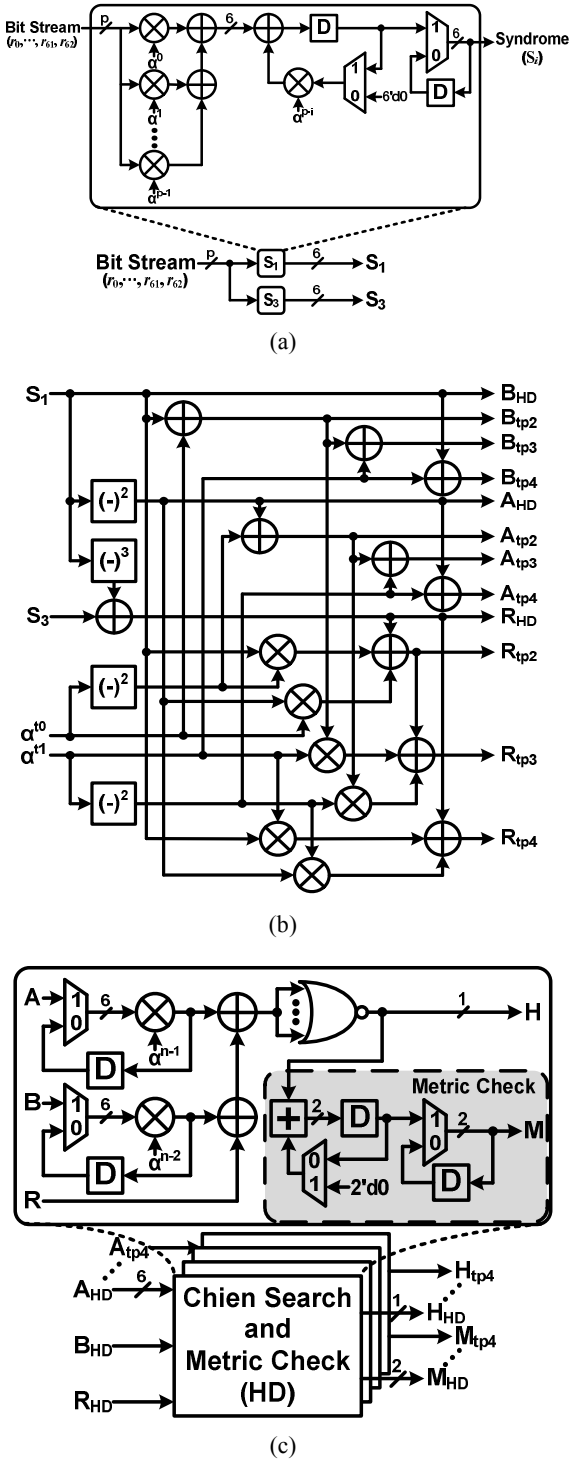

Fig. 4. Hard-decision kernel (a) the syndrome calculator, (b) the syndrome factor calculator, (c) Chien search and metric check.

the proposed method, two types of H value 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.

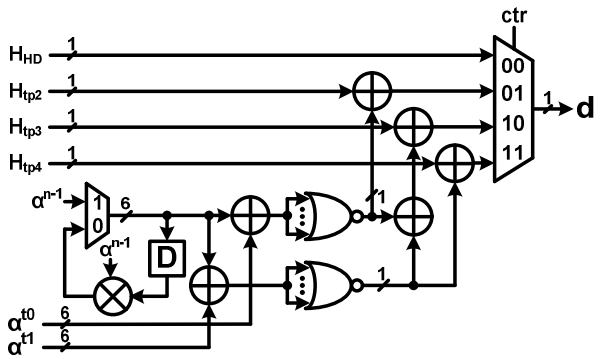

Fig. 5. Error location decision.

The second type of H value is compared with the reference H value to check whether the bit location is erroneous. The decision codeword d is decided by control signal ctr, which is selected by the controller. The main loop circuit generates α^j, where j = n-1, … , 0.

4. Controller

In the controller, *ctr* is selected by R_{HD}, R_{tp2}, R_{tp3}, and R_{tp4}, which are the syndrome factors from the SFC block, and metric value *M*. First, control signal *ctr* is decided by *R* values. For example, if R_{HD} is zero, *ctr* is 1 and selects the hard-decision candidate codeword. In addition, if R_{HD} is not zero and R_{tp2} is zero, *ctr* is 1 and decides the second test pattern. If R_{HD} and R_{tp2} are not zero and R_{tp3} is zero, *ctr* is 2. Finally, if R_{tp4} is zero and R_{HD}, R_{tp2}, and R_{tp3} are not zero, *ctr* is 3 and selects the third test pattern. It means that control signal *ctr* decides one of the test patterns, because *R* value = zero indicates that the number of errors is none or 1.

V. Results and Comparison

The proposed SD-BCH decoder architecture was modeled in 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, it was then synthesized using appropriate time and area constraints. Both simulation and synthesis steps were carried out using the SYNOPSYS design tool and 90-nm CMOS technology.

Table 1 shows the hardware complexity comparison of the proposed SD-BCH decoder for a P-parallel factor. The basic building blocks of the SD-BCH decoder are the GF multiplier, GF adder, multiplexer, D flip-flop, and the comparator. The number of complex operation units (e.g., GF multiplier, GF adder) increases depending on the parallel factor. However, the number of GF multipliers and GF adders in the SFC block is the same, regardless of the parallel factor.

Table 1. Hardware complexity of the SD-BCH (63, 51) decoder with P-parallel factor

	SC	TSC	SFC	CS and MC	ELD
GF mult.	P + 1	P	10	8P	P
GF adder	P	-	10	8P - 2	P + 2
Mux	2	6	-	16	1
D-FF	4	5	-	16	1
Comparator	-	P	-	-	-

Table 2. Implementation results of the SD-BCH (63, 51) decoder using a hard-decision kernel

Designs	Proposed SD-BCH			[9] P = 63
	P = 1	P = 3	P = 7	
CMOS Tech.	90 nm	90 nm	90 nm	90 nm
Voltage (V)	1	1	1	1
Total Gate Count (NAND)*	4,121	4,930	6,171	16,698
Area (μm^2)	11,630	13,915	17,416	34,768
Latency (cycles)	126 (504 ns)	42 (168 ns)	18 (72 ns)	1 ~ 4 (15 ~ 60 ns)
Max. Clock Speed (MHz)	250	250	250	66
Throughput (Mbps)†	202	627	1,417	842
Efficiency (Mbps/ K gates)	49	127	230	50

* Gate count is calculated based on area information of Yang et al. [9], unit gate area in the TSMC 90-nm CMOS library.
† Throughput is calculated by P×f×R, where P = levels of parallelism = number of bits in one clock cycle , f = clock speed, and R = k/n.

Table 2 shows the performance comparisons between the proposed SD-BCH decoders using levels of parallelism (P) = 1, 3, 7 and the conventional SD-BCH decoder using a fully-parallel hard-decision kernel [9]. The proposed SD-BCH decoder using a serial structure has 4,121 NAND gate counts from the synthesized results, which shows a 75% gate count saving, compared with the conventional SD-BCH decoder. The proposed BCH decoder operates at a clock rate of 250 MHz, has a latency of 126 clocks, and throughput of 202 Mbps. In addition, the proposed SD-BCH decoder using a 7-parallel structure performs 1.68 times throughput with gate-count savings of 50%, compared with a

conventional SD-BCH decoder. If there are single error in the input, the conventional SD-BCH decoder [9] has 1 processing cycle (= 15 ns) due to a fully-parallel (P = 63) hard-decision kernel and early termination scheme. However, if there are more than two errors in the input, the processing cycles of a conventional SD-BCH decoder will be increased with iteration. The proposed SD-BCH decoder has the same processing cycles regardless of the number of errors. That is, the proposed SD-BCH (63, 51) decoders with 3-parallel (P = 3) and 7-parallel (P=7) have 42 and 18 processing cycles, respectively. In addition, the efficiency of the SD-BCH decoder is calculated by the throughput-to-gate-count ratio (Mbps/M gates). The proposed SD-BCH decoders with P=3 and P=7 have much better efficiency compared to the conventional SD-BCH decoder using a fully-parallel (P = 63) hard-decision kernel. For the same BER performance, the proposed SD-BCH decoder has an efficient area and continuously operates without iteration.

VI. CONCLUSIONS

This paper presents a low-complexity non-iterative SD-BCH decoder architecture and its efficient design techniques. A hardware-friendly, non-iterative SD-BCH decoding algorithm is proposed and adopted for the SD-BCH decoder. In addition, a novel test syndrome computation, a hard-decision kernel, and an error location decision block are proposed. The proposed SD-BCH decoder has better BER performance than a HD-BCH decoder and significantly less hardware complexity than a conventional SD-BCH decoder.

ACKNOWLEDGMENTS

This research was supported by MSIP, Korea, under the ITRC support program (IITP-2016-H8501-16-1019) supervised by the IITP, and by the Basic Science Research Program through the NRF funded by the Ministry of Science, ICT and Future Planning (2013R1A2A2A01068628).

REFERENCES

[1] J. Y. Khan, and M. R. Yuce, "Wireless Body Area Network (WBAN) for Medical Applications," *New Developments in Biomedical Engineering*, Ch. 13, InTech, Jan. 2010.

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

[3] 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.

[4] J. Yeon, S.-J. Yang, C. Kim, H. Lee, "Low-Complexity Triple-Error-Correcting Parallel BCH Decoder," *Jour. of Semiconductor and Science Technology*, vol. 13, no. 5, pp. 465-472, Oct. 2013.

[5] A. Vardy, Y. Be'ery, "Maximum-Likelihood Soft Decision Decoding of BCH codes," *IEEE Trans. Inform. Theory*, vol. 40, no. 2, pp. 546-554, Mar. 1994.

[6] G. D. Forney, "Generalized Minimum Distance Decoding," *IEEE Trans. Inform. Theory*, vol. 43, pp. 125-131, Apr. 1966.

[7] M. Lalam, K. Amis, D. Leous, D. Feng, and J. Yuan, "An Improved Iterative Decoding Algorithm for Block Turbo Codes," *in Proc. IEEE Int. Symp. Inform. Theory*, pp. 2403-2407, Jul. 2006.

[8] D. Chase, "A class of algorithm for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol. 18, no. 1, pp. 170-182, Jan. 1972.

[9] C.-H. Yang, T.-Y. Huang, M.-R. Li, and Y.-L. Ueng, "A 5.4μW Soft-Decision BCH Decoder for Wireless Body Area Networks," *IEEE Trans. Circuits and Systems-I*, vol. 61, no. 9, pp. 2721-2729, Sep. 2014.

[10] G. T. Chen, L. Cao, "Test-Pattern-Reduced Decoding for Turbo Product Codes with Multi-Error-Correcting eBCH Codes," *IEEE Trans. Communication*, vol. 57, no. 2, pp. 307-310, Feb. 2009.

BoSeok Jung received the B.S and M.S degrees, both in Information & Communication Engineering, from Inha University, Incheon, Korea, in 2013 and 2015, respectively. His research interests VLSI and SOC architecture design for digital signal processing and communication systems.

Taesung Kim received the B.S degree in Information & Communication Engineering in 2015, from Anyang University, Anyang, Korea. Now, he is currently working toward the M.S degree in Inha University, Incheon, Korea. His research interests VLSI and SOC architecture design for digital signal processing and communication systems.

Hanho Lee received Ph.D. and M.S. degrees, both in Electrical & Computer Engineering, from the University of Minnesota, Minneapolis, 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. From August 2002 to August 2004, he was an Assistant Professor at the Department of Electrical and Computer Engineering, University of Connecticut, 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. His research interest includes VLSI architecture design for digital signal processing, forward error correction architectures, cryptographic systems, and communications.