JOURNAL OF SEMICONDUCTOR TECHNOLOGY AND SCIENCE, VOL.18, NO.4, AUGUST, 2018
https://doi.org/10.5573/JSTS.2018.18.4.518
ISSN(Print) 1598-1657
ISSN(Online) 2233-4866

Energy-efficient Key-equation Solving Algorithm for BCH Decoding

Hoyoung Yoo[1] and Youngjoo Lee[2]

***Abstract*—This paper presents an energy-efficient method to solve the key equation in BCH decoding. The key-equation solving block is so complicated that it consumes lots of energy because of multiple registers being dynamically updated every cycle. The block dominates the overall energy dissipation of strong BCH decoding and induces unwanted hot-spots. In achieving a high-performance BCH decoder, an energy-efficient algorithm should be developed for solving the key equation. This paper proposes a novel method to detect the case of single error by exploiting the relation among syndromes. If a single-error case is detected, the modified error-locator polynomial is obtained without solving the key-equation. For a (16383, 15543, 60) decoder implemented in a 130nm CMOS process, the proposed method saves 99% and 91% of energy compared to the conventional algorithm and the previous method that detects the error-free case, respectively.**

***Index Terms*—BCH decoding, Key equation solver, Low-power design**

I. INTRODUCTION

The Bose-Chaudhuri-Hocquenghem (BCH) code has been widely applied to a variety of systems ranging from optical communication [1] to storage signal processing [2]. Recently, strong BCH decoders that recover more than 10-bit random errors are reported for erroneous applications [3, 4]. However, the hardware complexity of a decoder is exponentially proportional to the error-correcting capability, and the energy consumed in the decoding process increases enough to cause unwanted hot-spots.

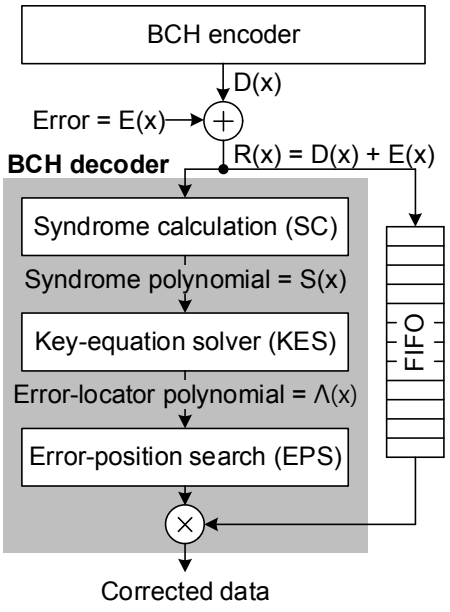

Fig. 1. A typical BCH decoding procedure with 3 pipeline stages.

The 3-stage architecture shown in Fig. 1 has been widely used to decode the (n, k, t) BCH code in hardware, where n is the code length, k is the data length, t is the number of correctable bit-errors [5]. Given a received BCH code $R(x)$, the syndrome calculation (SC) block generates a syndrome polynomial whose coefficients represent $2t$ syndromes. Using the syndrome polynomial $S(x) = s_1 + s_2x + \ldots + s_{2t}x^{2t-1}$, the key-equation solver

Manuscript received Feb. 7, 2018; accepted May. 14, 2018
[1] Department of Electronics Engineering, Chungnam National University, Deajeon 34134, Korea
[2] Department of Electrical Engineering, Pohang University of Science and Technology (POSTECH), Pohang 37673, Korea
E-mail : youngjoo.lee@postech.ac.kr

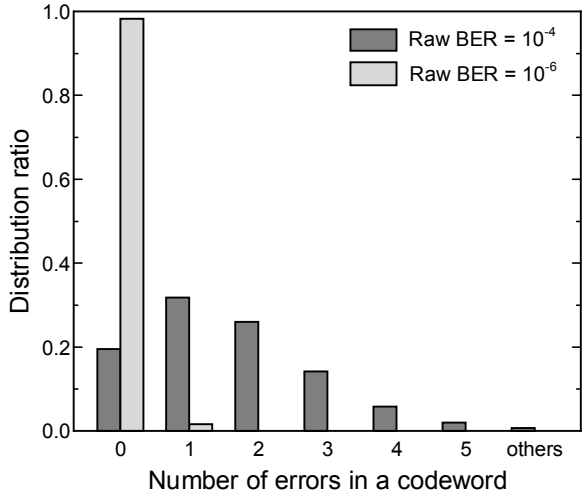

Fig. 2. The distribution of the number of errors in the received codewords.

(KES) derives an error-locator polynomial. In the last stage, the error-position search (EPS) block finds error positions by examining $\Lambda(x)$ for n times. Among the three stages, the KES is the most complicated and energy consuming stage especially in strong BCH decoding. Thus, several researches have been conducted to develop an energy-efficient scheme and structure for the KES.

One of the most popular methods to reduce the energy consumption of the KES is to detect error-free cases reported [6] and [10]. In the previous works, the KES and EPS stages are skipped when there is no error in the received BCH code. This is simply detected by examining whether the $2t$ syndromes are all zeros or not. However, the previous works are only effective when the raw bit-error rate (BER) is quite small. In the (16383, 15543, 60) BCH code, for example, 98% of received codes have no errors at a raw-BER of 10^{-6}, whereas the error-free case does not dominate when the raw BER is increased to 10^{-4} as illustrated in Fig. 2. For recent storage applications suffering from a raw-BER of more than 10^{-5}, it is needed to develop a new energy-efficient scheme that can enlarge the skipping cases [7].

To expand the cases to be skipped, this paper proposes a novel algorithm that detects whether the received code has only an error. As the error-locator polynomial for the single-error code can be easily generated without performing the KES process, energy can be saved remarkably even in strong BCH decoders aiming at erroneous applications. In addition to the single-error detecting scheme, we also present an energy-efficient KES algorithm and its hardware structure based on the Berlekamp-Massey (BM) architecture in [6] and [8].

The rest of this paper is organized as follows. Section II describes the proposed detection method. The energy-efficient KES algorithm and its hardware architecture are explained in Section III. Experimental results are analyzed in Section IV and concluding remarks are made in Section V.

II. SINGLE ERROR DETECTION

Given a syndrome polynomial $S(x)$, the error-locator polynomial, $\Lambda(x) = \lambda_0 + \lambda_1 x + \ldots + \lambda_t x^t$, is obtained by solving the key equation expressed as $\Lambda(x) \times S(x) = \Omega(x) \bmod x^{2t}$, where $\Omega(x)$ is the error-evaluator polynomial that is not necessary in BCH decoding. To solve the key equation, two algorithms known as Euclidian and BM have been used. Both of the algorithms are iterative in nature, and multiple intermediate values are stored into registers every cycle [2-4, 6, 9]. As all the values are updated every cycle, the KES block consumes a significant energy. To make the block energy-efficient, the error-free codes are detected early [2, 6, 9, 10]. In the previous algorithm, the received codeword is regarded as error-free when all the $2t$ syndromes are zero. If the error-free case is detected, the decoding process does not necessitate the remaining stages, eliminating the energy dissipated in the KES and EPS stages. We can further reduce the energy dissipation by expanding the detection range beyond the error-free case.

For the sake of simplicity, the n-bit transmitted codeword is represented as $D(x) = d_0 + d_1 x + \ldots + d_{n-1} x^{n-1}$, and the n-bit error vector as $E(x) = e_0 + e_1 x + \ldots + e_{n-1} x^{n-1}$. Here, we consider the single-error case that the received codeword, $R(x) = D(x) + E(x) = r_0 + r_1 x + \ldots + r_{n-1} x^{n-1}$, is corrupted by an error. In the syndrome polynomial $S(x)$, the i-th syndrome s_i is calculated as follows:

$$s_i = R(\alpha^i) = r_0 + r_1 \alpha^i + \ldots + r_{n-1} \alpha^{(n-1)i}. \quad (1)$$

Let $E(x)$ be x^ω, that is, there is an error at the ω-th position. For the case, the i-th syndrome is evaluated as $s_i = R(\alpha^i) = \alpha^{\omega i}$, since $D(\alpha^i)$ is always zero by definition. If the received codeword has a single error, therefore, the syndromes have the following relation:

Proposed KES algorithm

Initialize: $\lambda_0(0) = b_0(0) = 1$, $k(0) = 0$, $\gamma(0) = 1$.
Input: $S(x) = s_1 + s_2x + \ldots + s_{2t}x^{2t-1}$.

if $s_{2i+1} = s_{2i-1} \times s_2$, ($i$ = 1, 2, …, t–1) **begin**
  if $s_1 == 0$
    Error-free case → Decoding success.
  else
    Single-error case → $\Lambda(x) = s_1x + 1$.
  end
else begin
  for $u = 0$ **step** 1 **until** t–1 **begin**
    $\delta(u) = s_{2u+1} \times \lambda_0(u) + s_{2u} \times \lambda_1(u) + \ldots + s_{2u-t+1} \times \lambda_t(u)$.
    $ctrl = (\delta(u) \neq 0$ **and** $k(u) \geq 0)$.
    $\lambda_i(u+1) = \gamma(u) \times \lambda_i(u) + \delta(u) \times b_i(u)$ for $0 \leq i \leq t$.
    $b_i(u+1) = (ctrl)\ ?\ \lambda_{i-1}(u) : b_{i-2}(u)$ for $0 \leq i \leq t$..
    $\gamma(u+1) = (ctrl)\ ?\ \delta(u) : \gamma(u)$.
    $k(u+1) = (ctrl)\ ?\ -k(u) : k(u) + 2$.
  end
  $\Lambda(x) = \lambda_0(t-1) + \lambda_1(t-1)x + \ldots + \lambda_t(t-1)x^t$
end

Output: $\Lambda(x)$

Fig. 3. The proposed KES algorithm.

$$s_i = s_{i-1} \times s_1 \text{ for } 2 \leq i \leq 2t. \tag{2}$$

When all the syndromes satisfy (2) and s_1 is not zero, the codeword has only one error. According to (2), the single-error case is detected by $2t$–1 Galois-field (GF) multiplications and comparisons. The signal indicating the single-error case can be derived by ANDing ($2t$–1) cases.

The proposed algorithm can be extended to include the error-free case by conducting one additional comparison.

If s_1 is zero and all the rest syndromes satisfy (2), all the syndromes are zero, which means that the codeword is error-free. The proposed detection uses $2t$–1 GF multipliers and $2t$ comparators to find both the error-free and single-error cases. Note that the previous error-free detection requires $2t$ comparisons to find all-zero syndromes. Considering the much wider coverage of detection, we can accept the additional operations required in the proposed detection. Furthermore, the additional complexity can be significantly reduced by reusing the hardware units of the KES stage. The hardware structure supporting the proposed single-error detection will be illuminated in Section III.

As the proposed detection algorithm requires multiple comparisons and general GF multiplications, it is needed to relieve the computational complexity of the proposed work in order to suppress the addition of hardware components. Based on the well-known properties of GF operations, we present here a way to reduce the computation complexity. As shown in [4] and [10], squaring a syndrome expressed in (1), we can derive a special property as follows:

$$\begin{aligned} s_i^2 = R^2(\alpha^i) &= (r_0 + r_1\alpha^i + \ldots + r_{n-1}\alpha^{(n-1)i})^2 \\ &= (r_0)^2 + (r_1\alpha^i)^2 + \ldots + (r_{n-1}\alpha^{(n-1)i})^2 \\ &= r_0 + r_1\alpha^{2i} + \ldots + r_{n-1}\alpha^{2(n-1)i} \\ &= R(\alpha^{2i}) = s_{2i} \end{aligned} \tag{3}$$

Hence, an even-indexed syndrome s_{2i} is equal to the square of s_i. Let the received codeword have an error at the ω-th position. The proposed single-error detection checks every syndrome except s_1. As every even-indexed syndrome is dependent on an odd-indexed syndrome, the syndrome relation of (2) can be rewritten as

$$s_{2i+1} = s_{2i-1} \times s_2 \text{ for } 1 \leq i \leq t-1. \tag{4}$$

This relation leads to a new computing structure that compares only the odd-indexed syndromes. The new detecting equations can be implemented with a (t–1)-input AND operation and t–1 GF multiplications and comparisons. In other words, the hardware complexity required in (2) is reduced to almost a half by exploiting (3).

III. Proposed KES Architecture

1. Energy-efficient KES Algorithm

Among various KES algorithms, the BM algorithm has been widely used in BCH decoding due to its area-efficient architecture [3]. Based on the inversionless-BM (iBM) method that takes only t cycles [6], the KES algorithm adopting the proposed detection is presented in Fig. 3. In the proposed algorithm, t–1 equations between adjacent odd-indexed syndromes are checked first according to (4). When all the relations are satisfied, the proposed algorithm examines s_1 to decide whether the

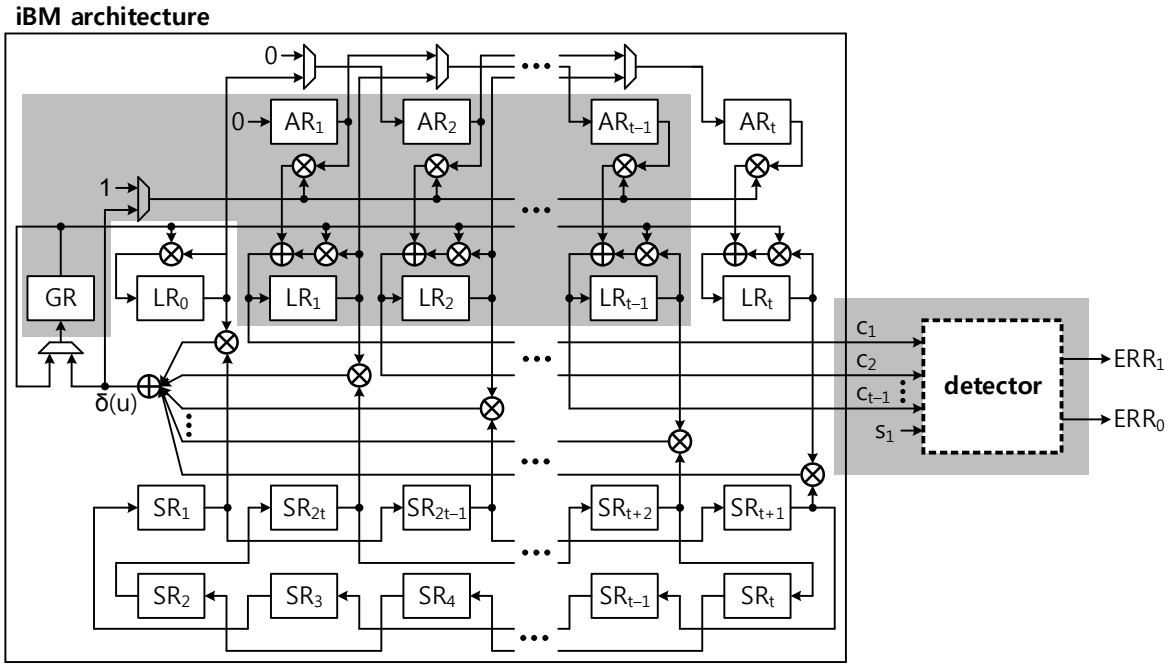

Fig. 4. The entire KES architecture that combines the *t*-cycle iBM algorithm and the proposed single-error detection.

codeword is error-free or has an error.

Unlike the error-free case that can be completed immediately, it is required for the single-error case to find the error location. We present a direct method to calculate the error-locator polynomial of the single-error case without solving the key-equation. In the EPS stage shown in Fig. 1, error positions are determined by examining whether $\Lambda(\alpha^i) = 0$, where i ranges from 0 to n–1. When α^i is a root of $\Lambda(x)$, there is an error at r_{n-i-1} of $R(x)$. In other words, if there is an error at the ω-th bit of $R(x)$, $1/\alpha^\omega$ should be a root of $\Lambda(x)$. Therefore, $\Lambda(x)$ of the single-error case is a first-order polynomial that has a root of $1/\alpha^\omega$. After t cycles in the previous iBM architectures [3, 6], the error-locator polynomial for the single-error codeword is calculated as $\Lambda_{prev}(x) = \alpha^{\omega t} \cdot x + \alpha^{\omega(t-1)}$, where $\Lambda_{prev}(x)$ denotes the error-locator polynomial of the previous iBM algorithms. Including [10], there have been many iBM modifications presented to improve the critical delay associated with the regular structures, however they need to calculate $\Lambda_{prev}(x)$ by updating some registers during t cycles as those variations are theoretically identical to the original iBM algorithm.

In our algorithm, the error-locator polynomial of the single-error codeword is generated in a different way. Preserving the same root of $\Lambda_{prev}(x)$ and eliminating redundant operations, the output of KES is set to $\Lambda_{pro}(x) = \alpha^\omega \cdot x + 1 = s_1 \cdot x + 1$ as described in Fig. 3. The proposed error-locator polynomial can be derived directly from s_1 while having the same root as the previous schemes. Note that the proposed algorithm takes only one cycle for both the error-free and single-error cases, whereas the previous methods issue the complicated t-cycle operations for the single-error codeword. Therefore, the energy required in the proposed KES architecture can be significantly saved as a result of the extended skipping cases. In addition, the proposed scheme can be applied to any variation, leading to the low-power KES architecture.

If one of the t–1 relations in (4) fails, the iBM algorithm should be performed to generate $\Lambda(x)$ by solving the key equation. At the u-th cycle of the iBM algorithm in Fig. 3, the coefficients of the error-locator polynomial are represented as $\lambda_x(u)$, and those of the assisted polynomial as $b_x(u)$, where subscript x denotes the index of a coefficient. In each iteration, the discrepancy value $\delta(u)$ is calculated first by using t+1 syndromes and $\lambda_x(u)$. Considering $\delta(u)$ and other control values such as $\gamma(u)$ and $k(u)$, $\lambda_x(u+1)$ and $b_x(u+1)$ are calculated by performing GF and shift operations. Finally, the control values are updated for the next iteration. Note that the iBM algorithm updates all the related values in every iteration, and thus consumes a significant energy in

Table 1. Synthesis results of iBM architectures for (16383, 15543, 60) BCH code

Architecture	Conventional	Error-free detection [6]	This work
Gate count	148 k	149 k	149 k
Average cycles*	60	10.06	1.72
Average latency*	300 ns	50.3 ns	8.6 ns
Energy consumption*	1.2×10^{-8} J	1.81×10^{-9} J	1.59×10^{-10} J
Decoder energy consumption*	1.7×10^{-8} J	4.8×10^{-9} J	3.2×10^{-9} J

* Estimated at a raw-BER of 10^{-5}.

general. The proposed scheme does not activate the energy-intensive BM processing for the dominant cases associated with no error or a single error, which plays a critical role in reducing the energy consumption.

2. Hardware Architecture

Fig. 4 shows an example of hardware realization supporting the proposed KES algorithm described in Fig. 3. Note that only the detector, which is depicted with a dotted box, is added to the iBM structure. In the iBM hardware, four types of registers are defined to do the KES processing for t cycles: SR_i is used to store $2t$ syndrome values computed in the SC block in Fig. 1, LR_i stores t+1 coefficients of error-locator polynomial, and the assisted polynomial and $\gamma(u)$ used in the iBM algorithm correspond to AR_i and GR, respectively. Note that processing units and registers in the iBM architecture are shared to calculate t–1 checking equations in (4).

The hardware components shown in the shaded region of Fig. 4 are activated to detect the single-error case. The registers in the activated region should be initialized to proper values so as to detect error-free and single-error codewords. More precisely, the multiplexor in the shaded region selects 1 instead of $\delta(u)$, and GR is set to s_2 at the detecting phase which is done before the iBM processing. At the same time, the syndromes corresponding to s_{2i-1} of (4) are placed into LR_1 through LR_{t-1}, and the first t–1 AR_x registers are initialized to s_{2i+1} of (4). Note that the other registers in Fig. 4 are all set to 0 during the detection, and there is no need to activate them for the detection. Since the proposed error detection scheme can start when all t odd syndromes are available, a syndrome calculator is assumed to be implemented in a parallel form.

After the register initialization, the detection is achieved in only one cycle, as described in Fig. 3. Note that c_x in Fig. 4 is $s_{2x+1} + s_{2x-1}\times s_2$. If all the c_x values are zero, or $s_{2x+1} = s_{2x-1}\times s_2$, we check s_1 to clarify if the codeword is error-free. As shown in Fig. 4, these comparisons necessitate simple logic circuitry that decides whether the input values are zero or not. Thus, the additional hardware complexity required to implement the proposed detection method is negligible compared to the complexity of the iBM architecture.

When the relation of (4) is met, the detector activates one of ERR_0 and ERR_1 signals depending on s_1, where ERR_0 and ERR_1 represent the detection of error-free and single-error cases, respectively. Activation of ERR_0 leads to the completion of the decoding process, since there is no bit-error to be corrected. If ERR_1 is activated, the error-location polynomial is immediately set to $\Lambda(x) = s_1x + 1$ by setting LR_0 to 1, LR_1 to s_1, and the rest of LR_n registers to 0. For the single-error case, therefore, the proposed KES architecture takes only one cycle and does not process the complicated iBM algorithm, leading to significant reduction in energy consumption as well as in decoding latency.

IV. Experimental Results

To verify the energy efficiency of the proposed detection method, we have implemented three different KES blocks: the conventional block that does not have any detection methods, a block based on the previous detection algorithm that detects the error-free case at the iBM structure reported in [6], and the proposed architecture that detects the single-error case as well as the error-free case. For a fair comparison, all the decoders adopt the iBM algorithm to calculate the error-locator polynomial, and they are all designed under an operating frequency of 200 MHz in 130 nm CMOS technology. The synthesis results targeting the (16383, 15543, 60) BCH code are summarized in Table 1. Among the three structures, the proposed architecture uses the smallest energy at a raw-BER of 10^{-5}, and the average number of processing cycles

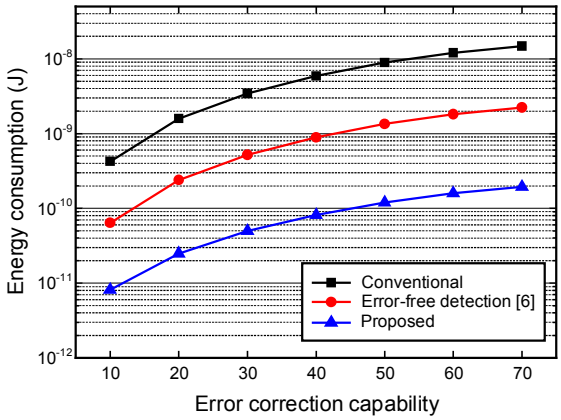

Fig. 5. Average energy consumptions of various KES blocks for (16383, 16383–14t, t) BCH codes, where t denotes the error-correction capability.

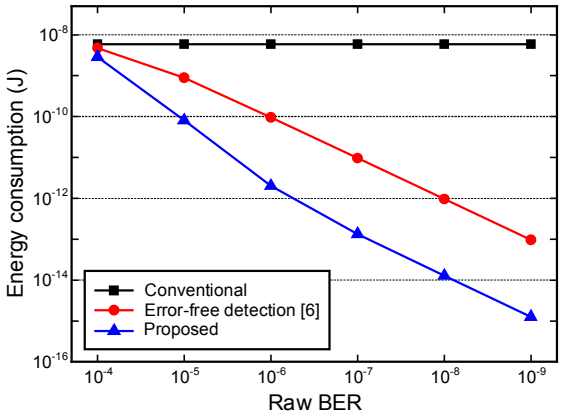

Fig. 6. Average energy consumptions of three KES blocks for (16383, 15923, 40) BCH code when the raw-BER varies.

and latency are remarkably reduced by skipping both the error-free and the single-error codewords. As shown in Table 1, the hardware complexities resulting from the three KES architectures are quite similar to one another, as the logic complexities that are additionally needed to detect the skip cases are much smaller than that of the iBM algorithm.

The energy dissipated in the KES block is estimated based on the following terms. E_{BM} is the energy consumed to process the iBM algorithm for t-cycles, E_{ACT} represents the amount of energy taken to perform the proposed detection with the hardware units shown in the shaded region of Fig. 4, and E_{COMP} is the energy consumed to check if all syndromes are zero. E_0 stands for the average energy consumed for the previous error-free detection method, and $E_{0\&1}$ is the energy consumed in the proposed detection algorithm that finds both the error-free and single-error cases. Given a raw-BER of p_e, E_0 can be defined as

$$E_0 = E_{\mathrm{BM}} \times (1-(1-p_e)^n) + E_{\mathrm{COMP}}. \quad (5)$$

Similarly, E0&1 can be estimated as follows:

$$E_{0\&1} = E_{\mathrm{BM}} \times (1-(1-p_e)^n - n \times p_e \times (1-p_e)^{(n-1)}) + E_{\mathrm{ACT}}. \quad (6)$$

Note that the probabilities of error detection failures that activate the BM block are important since energy consumption in the BM block is dominant. Based on (5) and (6), the average energy consumptions of three KES blocks are compared in Fig. 5. In the comparison, several BCH codes, which are constructed over GF(2^{14}) by varying the error-correction capability t, are considered with assuming a fixed raw-BER of 10^{-5}. The proposed KES algorithm expands the skipping range of the iBM processing. Compared to the conventional KES block and the previous block that detects the error-free case, the proposed block saves 98.7 % and 91.2 % of energy consumption in case of the (16383, 15543, 60) BCH code. Fig. 6 shows how the energy consumption of the KES block is related to the raw-BER. The results are obtained for the (16383, 15923, 40) BCH code. Note that the proposed algorithm always consumes the smallest energy among the three blocks, and significantly reduces energy consumption when the BER is low. Even if the BER increases to 10^{-4}, the proposed work saves 51% of energy consumption, while the previous error-free detection consumes similar energy to the conventional block.

V. CONCLUSION

In this paper, we have presented an energy-efficient method to solve the key equation in BCH decoding. The proposed method is to detect whether the received codeword has only an error or not. To detect the single-error case efficiently, a specific relation that is hold among syndromes is presented. In addition, a direct way to generate the error-locator polynomial having the same root corresponding to the single-error case is described, which eliminates the need to perform the complicated key-equation solving process for the single-error case. The additional hardware complexity for the proposed KES algorithm is minimized by sharing the hardware

units in the conventional processing. The previous method turns off the KES hardware only for error-free cases, while the proposed work saves the energy consumption further by extending the skipping range to the single-error case. In decoding strong BCH codes, consequently, the proposed method considerably reduces energy consumption even compared to the previous error-free detection method.

ACKNOWLEDGMENTS

This work was supported by the National Research Foundation (NRF) grants funded by the Korea government (MSIT) (2016R1C1B1007593 and 2018R1A4A10 25679) and by the IC Design Education Center (IDEC).

REFERENCES

[1] Forward Error Correction for Submarine Systems. ITU Telecommunication Standardization Sector, ITU-T Recommendation G.975, 2000.

[2] P. Meku, R. Seva, K. K. Kim, Y.-B. Kim, and M. Choi, “Adaptive multi-path BCH decoder to alleviate hotspot-induced DRAM bit error variation in 3D homogenous processor,” IEIE J. Semicond. Technol. Sci., vol. 17, no. 5, 717–728, Oct. 2017.

[3] S. Hwang, J. Jung, D. Kim, J. Ha, I.-C. Park, and Y. Lee, "An energy-optimized (37840,34320) symmetric BC-BCH decoder for healthy mobile storages," in Proc. IEEE Asian Solid-State Circuits Conf. (A-SSCC), 2017, pp. 169-172.

[4] K. Lee, S. Lim and J. Kim, “Low-Cost, Low-Power and High-Throughput BCH Decoder for NAND Flash Memory,” in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), 2012, pp. 413–416.

[5] S. Lin and D. J. Costello, Error control coding: Fundamentals and Applications, 2nd ed. Englewood Cliff, NJ: Prentice-Hall Inc., 2004.

[6] B. Park, J. Park, and Y. Lee, “Area-optimizaed fully-flexible BCH decoder for multiple GF dimensions,” IEEE Access, vol. 6, pp. 14498–14509, Mar. 2018.

[7] S. Kim, S.-H. Lee, S.-K. Park, Y. Kim, S. Cho, and B.-G. Park, “Investigation of retension characteristics caused by charge loss for charge trap NAND flash memory,” IEIE J. Semicond. Tech. Sci., vol. 17, no. 5, pp. 584–590, Oct. 2017.

[8] J. L. Massey, “Shift-register synthesis and BCH decoding,” IEEE Trans. Inf. Theory, vol. 15, no. 1, pp. 122-127, Jan. 1969.

[9] B. Park, S. An, J. Park, and Y. Lee, “Novel folded-KES architecture for high-speed and area-efficient BCH decoders,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 64, no. 5, pp. 535–539, May 2017.

[10] W. Liu, J. Rho, and W. Sung, “Low-Power, high-throughput BCH error correction VLSI design for multi-level cell NAND flash memories,” in Proc. IEEE Workshop on SiPS, 2006, pp. 303–308.

Hoyoung Yoo received the B.S. degree in Electrical and Electronic Engineering from Yonsei university, Seoul, Korea in 2010, and the M.S. and Ph.D. degrees in Electronics Engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2012 and 2016, respectively. Since September 2016, he has been an assistant professor in the department of Electronics Engineering at Chungnam National University. Special area of interests includes VLSI design for error correction codes and 5G communication systems.

Youngjoo Lee received the B.S., M.S. and Ph.D. degrees in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2008, 2010 and 2014, respectively. Since February 2017, he has been an Assistant Professor in the department of Electrical Engineering, POSTECH, Pohang, Korea. Prior to joining POSTECH, he was with Interuniversity Microelectronics Center (IMEC), Leuven, Belgium, from May 2014 to February 2015, where he researched reconfigurable SoC platforms for software-defined radio systems. From March 2015 to February 2017, he was with the Faculty of the Department of Electronic Engineering, Kwangwoon University, Seoul, Korea. His current research interests include the algorithms and architectures for embedded processors, intelligent mobile systems, advanced error-correction codes, and mixed-signal circuit designs.