http://dx.doi.org/10.5392/IJoC.2013.9.1.033

Optimization of a Systolic Array BCH encoder with Tree-Type Structure

Duk-Gyu Lim, Sharad Shakya and Je-Hoon Lee
Division of Electronic and Information Communication Engineering.
Kangwon National University, Samcheok Campus,
1 Joongang-ro, Samcheok, Gangwon-do, 245-711, South Korea

ABSTRACT

BCH code is one of the most widely used error correcting code for the detection and correction of random errors in the modern digital communication systems. The conventional BCH encoder that is operated in bit-serial manner cannot adequate with the recent high speed appliances. Therefore, parallel encoding algorithms are always a necessity. In this paper, we introduced a new systolic array type BCH parallel encoder. To study the area and speed, several parallel factors of the systolic array encoder is compared. Furthermore, to prove the efficiency of the proposed algorithm using tree-type structure, the throughput and the area overhead was compared with its counterparts also. The proposed BCH encoder has a great flexibility in parallelization and the speed was increased by 40% than the original one. The results were implemented on synthesis and simulation on FPGA using VHDL.

***Keywords**: BCH code, encoder, parallel processing, tree-type systolic arrays.*

1. INTRODUCTION

The theory of error detection and correction codes deals with the reliable transmission and storage of data over unreliable communication channels. Error correction coding is the encoding process of adding parity bits to the message bits, making it longer in size than the original bits which are mostly called "code-word". When this code-word is received at destination, it is decoded to retrieve the original message bits. The BCH (Bose-Chaudhuri-Hochquenghem) code is one of the most powerful algebraic codes. It is extensively used for the modern digital communication system owing to its efficient error correction ability with high speed hardware implementation. Compared to the RS (Reed-Solomon) codes, BCH codes can achieve around additional 0.6dB coding gain over the AWGN noise [1].

The conventional BCH encoder is implemented by LFSR (linear feedback shift register) architecture. Since this architecture is based on a single feedback loop, it can be operated at high frequency. However, the major drawback of this LFSR based BCH encoder is that it is operated with bit-serial manner, thereby it operates only one message bit in a single clock cycle. Thus, this fact becomes a barrier for a high throughput. Owing to the ever increasing demands, where high throughput is usually desired, the clock frequency of such LFSR based encoders cannot keep up with data transmission rate and thus parallel processing must be employed [2-6].

Several parallel BCH encoding methods have been introduced earlier such as matrix multiplication, unfolding method, and CRT based encoding [2-4]. In matrix multiplication, the BCH encoder becomes very complex as the parallel factor increases than the number of registers in the circuit. The generator polynomial has to be modified in the unfolding method for greater output making the area overhead. The complexity of the CRT method is the flexibility in parallelization. In a parallel BCH encoder circuit, p bits of data are processed at a time, where the total number of clock cycles can be reduced by p times. But it doesn't increase the throughput by p times because the critical path becomes longer as the parallel factor p increases. Therefore, the parallel encoder with high speed and small area overhead are essential. Further the flexibility to increase and decrease the parallel factor p is also enhanced for the desired compatible throughput.

In this paper, we present a new systolic array for BCH encoder with several p-parallel factors. In addition, the proposed BCH encoder introduces a tree-type structure so as to reduce the delay time for the critical path. The proposed systolic array encoder has a great flexibility in parallelization without any complexity with high throughput. It can improve the performance compared to its counterparts without any significant area increase. In addition, the optimized tree-type structure significantly increases the throughput in the same parallel factor. We have implemented a (31,16) triple error correcting binary BCH code, which is similar to error detecting capability of CRC-16, as an example. The synthesis and simulation results for the several p factors original encoder and optimized systolic array are presented using VHDL on FPGA.

The structure of the paper is as follows. Sect. 2 gives a brief explanation of the BCH code and generator polynomial with the conventional serial and parallel BCH encoders. Sect. 3

* *Corresponding author, Email : jehoon.lee@kangwon.ac.kr*
Manuscript received Oct. 15, 2012; revised Dec 21, 2012; accepted Dec 31, 2012

contains the original and proposed tree-type systolic array encoder with the subsequent description and architecture and Sect. 4 contains the synthesis and simulation results. A short conclusion is given in Sect. 5.

2. SERIAL AND PARALLEL BCH ENCODER

In a binary BCH (*n,k*) code, a *k* bit message is encoded in *n*-bit code-word [7]. It consists of *k* bit message and *n-k* parity bits. The *n*- bit code-word is defined as $(c_{n-1},c_{n-2},\ldots\ldots,c_0)$, where $c_i \in GF(2)$, $(0 \le i \le n-1)$ as the co-efficient of a degree *n-1* of polynomial *c(x)* and *k* bit message is defined as $(m_{k-1}, m_{k-2},\ldots,m_0)$, where $m_i \in GF(2)$, $(0 \le i \le k-1)$ as the coefficient of a degree *k-1* of polynomial *m(x)*. The encoding of a BCH code can be expressed as $c(x) = m(x)g(x)$, where *g(x)* is the generator polynomial of a degree *n-k*.

Consistently, BCH encoding is constructed with three steps. Multiplying the message *m(x)* by x^{n-k} and dividing it by generator polynomial *g(x)*, where the remainder $Rem(m(x).x^{n-k})_{g(x)}$ is obtained. The remainder is now added to the message to form a code-word as shown in Eq. (1). In this paper, the proposed BCH encoder is implemented with (31,16, 3) BCH code [8]. Let α be the primitive elements of $GF(2^5)$ such that the primitive polynomial is x^5+x^2+1 and $\theta_i(x)$ be the minimal polynomial of α^i. The first three odd powers of α minimal polynomial are

$$\alpha : \theta_1(x) = 1 + x^2 + x^5$$
$$\alpha^3 : \theta_3(x) = 1 + x^2 + x^3 + x^4 + x^5$$
$$\alpha^5 : \theta_5(x) = 1 + x + x^2 + x^4 + x^5$$

$$c(x) = Rem(m(x).x^{n-k})_{g(x)} + m(x).x^{n-k} \quad (1)$$

Thus, we get the generator polynomial, *g(x)* as shown in Eq. (2).

$$g(x) = 1 + x + x^2 + x^3 + x^5 + x^7 + x^8 + x^9 + x^{10} + x^{11} + x^{15} \quad (2)$$

For the conventional bit-serial BCH encoding, the *k* message bits are input to the LFSR with bit-serial manner. At the k^{th} cycle, the registers contain $Rem(m(x).x^{n-k})_{g(x)}$, which is also called the parity bits. Fig. 1 illustrates the circuit connection of a conventional serial BCH encoder. The critical path of this bit-serial architecture consists of two XOR gates as shown in Fig. 1. This architecture is quite straight forward, but it cannot run in a high speed as the application requirement.

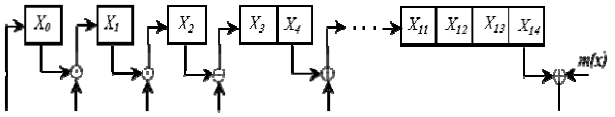

Fig. 1. The architecture of conventional bit-serial BCH Encoder

The parallelization of the circuit is the method of sending the *p* number of message bits at a time *t*. When *p* message bits arrive in each clock cycle, only *k/p* clock cycles are required to compute the remainder in the registers. Unfolding is a method of parallelization of a circuit and has a high throughput [3]. In the *J*-unfolded architecture, there are *J* copies of each node with the same function as in the original architecture. It is assumed that there is a path from node *U* to node *V* in the original architecture with *W* delay elements. Therefore, node U_i is connected to $V_{[(i+w)\%J]}$ with $[(i+w)/J]$ delay elements ,where, U_i, V_j $(0 \le i,j \le J)$ are the copies of nodes *U* and *V* respectively. Fan-out problem also exist in the unfolding method, but retiming is not accessible when the *J* factor is larger than the degree difference between the highest and the second highest order of *g(x)*.

$$g(x) = 1 + g_1 + g_2x^2 + \cdots + g_{n-k-1}x^{n-k-1} + x^{n-k} \quad (3)$$

In the case, if a *J* unfolded BCH encoder is acquired, the generator polynomial needs to be modified and the remainder $Rem(m(x).x^{n-k})_{g(x)}$ in the BCH encoding can be obtained by the following steps :

Step1: Multiply the input message *m(x)* by *p(x)*.
Step2: Divide $m(x)p(x)x^{n-k}$ by *g'(x)*.
Step3: The remainder of step2 is divided by *p(x)* again as $Rem(m(x).x^{n-k})_{g(x)}$

Additional hardware will increase dramatically when large unfold factor *J* is used [4].

3. THE PROPOSED TREE-TYPE SYSTOLIC ARRAY BCH ENCODER

Before implementing a *p*-parallel encoder, we need to know the state of the LFSR at time *t+1* from the state *t*. Let $X(t)=[X_0,X_1,\ldots,X_{n-1}]$ denotes the state of the registers at time *t* and z_t denotes the input bit to be entered at time *t*. *T* is the associate matrix of the generator polynomial *g(x)*. The modulo-2 matrix in Eq. (4) performs the shifting operation in the serial BCH encoder registers

$$X_{t+1} = X_tT \oplus z_tG \quad (4)$$

Where,
$$G = [g_0, g_1, \ldots\ldots\ldots g_{n-1}] \quad (5)$$

$$T = \begin{bmatrix} 0 & 1 & 0 \cdots & 0 \\ 0 & 0 & 1 \ldots & 0 \\ . & . & . \ldots & . \\ . & . & . \ldots & . \\ 0 & 0 & 0 \ldots & 1 \\ g_0 & g_1 & g_2 \cdots & g_{n-1} \end{bmatrix} \quad (6)$$

Let $z_t, z_{t+1}\ldots\ldots z_{t+p-1}$ are the *p* message bits to be entered into a serial conventional encoder during the *p* shifting operation [9]. The contents of the encoder at the *p* shifts are derived by the vector X_{t+p}. From the Eq. (4), a recursive equation for X_{t+p} can be derived as:

$$X_{t+p} = X_tT^p \oplus Z_tD_p \quad (7)$$

where, Z_t be the input message bits in sequence.

$$Z_t = [z_{t+p-1}, z_{t+p-2}, \ldots\ldots z_{t+1}, z_t]$$

$$D_p = \begin{bmatrix} G \\ GT \\ GT^2 \\ . \\ . \\ GT^{p-1} \end{bmatrix} \quad (8)$$

From the Eq. (7), the circuit has the ability to compute the *p*-bits in parallel, which changes the state from X_t to X_{t+p} in a single clock cycle. The generator polynomial for the eight-parallel encoder for (31,16, 3) BCH code is given in Eq. (2). For this generator polynomial, the Eq. (7) gives the state transition D_8 with vectors G to GT^7 as shown in equation (10).

$$G = [111101011111000] \quad (9)$$

$$D_8 = \begin{bmatrix} 111101011111000 \\ 011110101111100 \\ 001111010111110 \\ 000111101011111 \\ 111110101010111 \\ 100010001010011 \\ 101100011010001 \\ 101011010010000 \end{bmatrix} \quad (10)$$

The systolic array BCH encoder is almost the same with the conventional serial BCH encoder. In the serial BCH encoder the output of the rightmost XOR gate is the input to the rest of the XOR gates as well as the first register. Whereas, in the systolic array BCH encoder, the output of the XOR gates are the input to the next stage. The position of the XOR gates in each stage is a replica of the first stage. This process can be concluded as a shift operation of the generator polynomial. The stages are the number of parallel factor *p*. After the XOR operation the output of the last or *p-1* stage is the input to the first stage and is repeated consecutively. The vectors *G, GT, GT^2, ... GT^{p-1}* represents the contents of the serial conventional encoder as the vector *G* is shifted *p-1* times.

For the BCH (31,16,3) code , the vector of the generator polynomial *g(x)* is shown in (9) and the eight shifted vectors as D_p for the eight parallel systolic array encoder is shown in Eq. (10). Now the vectors *G, GT, GT^2, ..., GT^7* represents the stages of the eight parallel systolic array BCH encoder. Let $X_0(t+8)$ to $X_{14}(t+8)$ represents the value of the registers at *(t+8)*. $z_0(t)$ to $z_7(t)$ represents the eight parallel input data at *t* [8]. We can get the value of the registers with eight parallel inputs processed in the systolic array BCH encoder as follows:

$U_{14} = z_0(t) + X_{14}(t)$
$U_{13} = z_1(t) + X_{13}(t)$
$U_{12} = z_2(t) + X_{12}(t)$
$U_{11} = z_3(t) + X_{11}(t)$
$U_{10} = z_4(t) + X_{10}(t)$
$U_9 = z_5(t) + X_9(t)$
$U_8 = z_6(t) + X_8(t)$
$U_7 = z_7(t) + X_7(t)$

$\mathbf{X_{14}(t+8)} = X_6 + 2U_{14} + U_{13} + U_{12} + U_{11} + U_{10}$
$\mathbf{X_{13}(t+8)} = X_5 + 2U_{14} + 2U_{13} + U_{12} + U_{11} + U_{10} + U_9$
$\mathbf{X_{12}(t+8)} = X_4 + 4U_{14} + 2U_{13} + 2U_{12} + U_{11} + U_{10} + U_9 + U_8$
$\mathbf{X_{11}(t+8)} = X_3 + 4U_{14} + 4U_{13} + 2U_{12} + 2U_{11} + U_{10} + U_9 + U_8 + U_7$
$\mathbf{X_{10}(t+8)} = X_2 + 5U_{14} + 3U_{13} + 3U_{12} + U_{11} + U_{10} + U_9 + U_8 + U_7$
$\mathbf{X_9(t+8)} = X_1 + 4U_{14} + 4U_{13} + 2U_{12} + 2U_{11} + U_9 + U_8 + U_7$
$\mathbf{X_8(t+8)} = X_0 + 4U_{14} + 3U_{13} + 3U_{12} + U_{11} + U_{10} + U_9 + U_8 + U_7$
$\mathbf{X_7(t+8)} = 3U_{14} + 3U_{13} + 3U_{12} + 2U_{11} + U_9 + U_7$
$\mathbf{X_6(t+8)} = 2U_{14} + 2U_{13} + 2U_{12} + U_{11} + U_{10} + U_8$
$\mathbf{X_5(t+8)} = 3U_{14} + 2U_{13} + 2U_{12} + 2U_{11} + U_{10} + U_9 + U_7$
$\mathbf{X_4(t+8)} = 3U_{14} + 2U_{13} + U_{12} + U_{11} + U_{10} + U_9 + U_8$
$\mathbf{X_3(t+8)} = 4U_{14} + 3U_{13} + 2U_{12} + U_{11} + U_{10} + U_9 + U_8 + U_7$
$\mathbf{X_2(t+8)} = 3U_{14} + 3U_{13} + 2U_{12} + U_{11} + U_9 + U_8 + U_7$
$\mathbf{X_1(t+8)} = 2U_{14} + 2U_{13} + 2U_{12} + U_{11} + U_8$
$\mathbf{X_0(t+8)} = U_{14} + U_{13} + U_{12} + U_{11} + U_7$

Constructing a systolic array BCH encoder is simple and can be performed from the equation given in Eq. (11). The equation is to trace the path of the node from one register to another through different stages. The node can be mentioned by the degree of the register in a column and the position of the p-factor stages in a row. First, the conventional serial BCH encoder has to be drawn. The positions of the XOR gates are same as in first stage and are repeated until *p-1* stages.

$$X_a[b] = X'_{a+r}[b + r] \quad (11)$$

Where, *a* is the degree of the register
b is the parallel stage
r is the number of registers in between two XOR gates

For example, to find a path from a node $X_{11}[0]$, $a = 11$, $b = 0$ and $r = 3$, we get a node $X'_{14}[3]$ from the Eq. (11). The first node without name, is the feedback of the rightmost XOR gate, is always shifted by one to the next stage. In the result, if $b' > b$, then $b'-b$ is the number of nodes to be shifted. The out-put of the rightmost XOR gate is input to all other XOR gates in a same manner of the serial BCH encoder. As the output of the *p-1* stages is the input to the registers, there is no need to trace the path.

Instead of one message bit input in the serial BCH encoder, the encoder receives one byte of message bits in parallel in our example. This allows for multiple bits of encoding to be performed simultaneously and at greater speeds than using comparative models. As the stages are the replicas of the first stage, it can be increased or decreased to any factor without increase in hardware complexity. The key reason of its flexibility is that the stages can be moved to any number of parallel factors. However, the value of the *p-1* stage should be the input to the first stage.

In the paper, we focus on the critical path delay of the systolic array BCH encoder. Since the circuit functions as a loop, it faces a critical path delay. In 8-parallel systolic array BCH encoder, the longest critical path is 7T, where T is the delay of an XOR gate. The calculation of the critical path from registers X_2 to X_{10} is shown in Fig.3.

During the XOR operation, the signal *S(0)* is obtained from the value of the register X_2 and the value from Z_0 and X_{14}, which has a critical path of 2T. Similarly the signal *S(0)* is transferred to the another stage to calculate the value from Z_2 and X_{12} and creates a new signal *S(1)*. The critical path is added to 3T. The overall data delay path for the longest critical path for the signal *S(5)* is 7T.

In order to acquire a low delay, tree-type structure has been optimized for the longest critical paths as shown in Fig. 2, which is not shown due to complexity. The critical path of the proposed tree-type BCH encoder has been reduced to 5T. Using tree type structure, the delay caused by the cascaded XOR gates is much less than that caused by large fan-out [2]. Compared to the original systolic BCH encoder, the proposed tree-type encoder is speeded up by 40%.

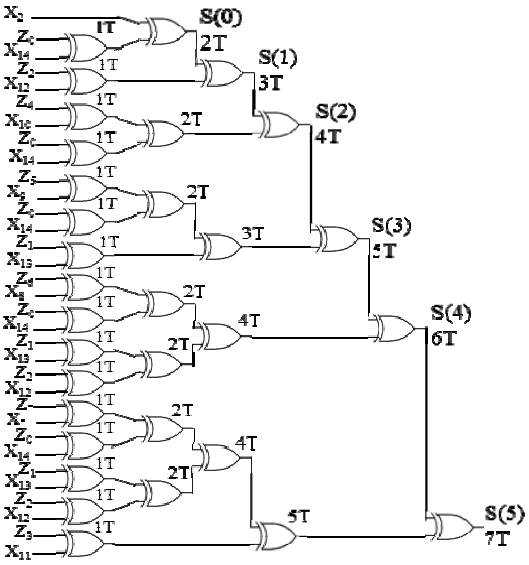

Fig. 2. Data delay path from register X_2 to X_{10}

4. PERFORMANCE RESULTS

The BCH (31,16,3) code encoder is synthesized on the Xilinx ISE 10.1, Virtex II PRO with a device XC2VP30 [11]. The simulation results are implemented with the generator polynomial in Eq. (2) with degree 15 and 9 non-zero terms.

First, we simulate the conventional BCH encoder in bit-serial manner and the proposed BCH encoder employing systolic array architecture. Figure 3 and Figure 4 show the comparison results in area and throughputs of bit-serial (p=1), 4-parallel, 8-parallel and 16-parallel systolic array BCH encoders, respectively. As shown in Fig. 3, 8-parallel BCH encoder requires 66% more area compared to the conventional bit-serial BCH encoder. In addition, the hardware complexity of 16-parallel BCH encoder will be increase by 38% compared to 8-parallel BCH encoder. However, as shown in Fig. 4, the throughput of 8-parallel BCH encoder is increased by 80% compared to the conventional bit-serial BCH encoder. Furthermore, the performance of 16-parallel BCH encoder will be enhanced by 52% compared the 8-parallel counterpart. Consequently, the increase in throughput is quite steep as increasing number of parallel factor. When we compared the conventional bit-serial BCH encoder to 16-parallel BCH encoder employing systolic array, the throughput has increases by 91% (i.e. from 648 Mbps to 7.1 Gbps), whereas area by 79%. From the overall result above, it is still worthwhile to implement systolic array typed BCH encoder with high parallel factors.

Table 1 and Table 2 show the comparison results of the eight parallel J unfolding methods, original systolic array and the proposed tree-type systolic array BCH encoders with their device utilization and the critical path delay respectively. The J unfolding method has been implemented without any change in the generator polynomial, i.e. no retiming is applied. Some XOR gates in the proposed tree-type encoder are increased than original systolic encoder. The area is still efficiency (about 0% of the allocated area of the XC2VP30 device). In Table 1, the number of slices is slightly increased, which is negligible for the modern FPGA. The maximum frequency operation of the proposed tree-type encoder is around 657.9 MHz.

From Table 2, the critical path of the original systolic array BCH encoder and unfolding one are almost same with 2.097 ns and 2.078 ns respectively. Using tree-type systolic array encoder, the critical path is reduced to 1.49ns from front-end simulation using Xilinx tool. The throughput of the proposed encoder has speeded up by 40% than the original systolic array encoder.

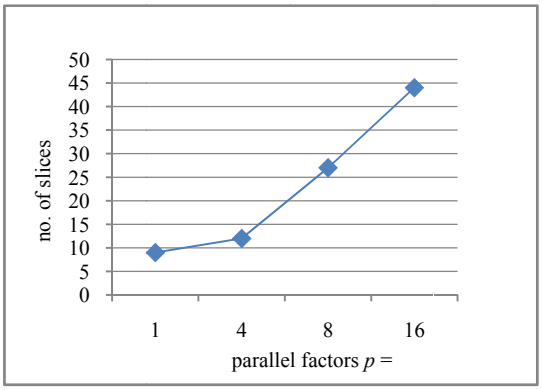

Fig. 3. Area summary of p-parallel original encoders

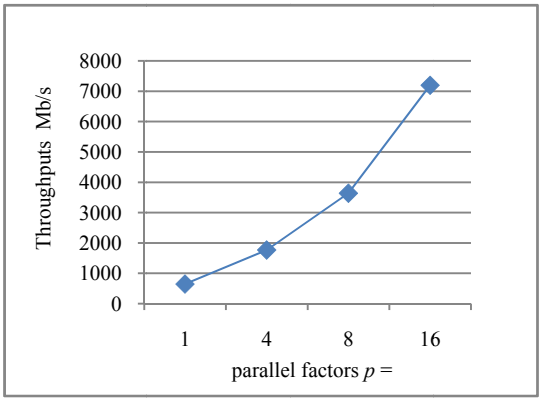

Fig. 4. Throughputs of p-parallel original encoders

Table 1. Area summary of 8-parallel encoders after synthesis

	Unfolding method	Original systolic	Proposed systolic
Slices	27	27	28
Slice flip-flops	15	15	15
4 input LUT	50	50	52
IOBs	24	26	26

Table 2. Speed summary of 8-parallel encoders after synthesis

	Unfolding method	Original systolic	Proposed systolic
Critical path delay	2.078 ns	2.097 ns	1.497 ns
Throughputs (Mbps)	3,670	3,639	5,096

5. CONCLUSION

In this paper, we present the tree-type systolic array BCH encoder to perform in parallel without significant area overhead. Several *p*-factors of the original systolic array BCH encoder has been compared with its area and speed. The systolic array BCH encoder has also been compared with the unfolding method. Using tree-type structure to the original systolic array encoder, the performance is tremendously better with a minor increase in area. Considerably, the proposed BCH encoder has a great flexibility to any number of parallelization factors without any complexity. The fan-out effect has been disregarded in our experiment. Retiming can be applied in the proposed encoder without any modification in the generator polynomial and increasing its hardware. Future work can be directed toward reducing the fan-out effect in the proposed tree-type systolic array BCH encoder to amend its output.

ACKNOWLEDGEMENT

Je-Hoon Lee is the corresponding author.

This research was financially supported by the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea (NRF) through the Human Resource Training Projects for Regional Innovation (2012H1B8A2026055). This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2011-0013219).

REFERENCES

[1] X. Zhang and K. K. Parhi, "High-speed Architectures for Parallel long BCH Encoder," in *Proc. ACM Great Lakes Symp. VLSI*, Apr. 2004, pp.1-6.

[2] T. B. Pei, C Zukowski, "High-speed parallel CRC circuits in VLSI," *IEEE Trans. on Communications*, vol.40, no.4, Apr.1992, pp.653-657.

[3] K. K. Parhi, "Eliminating the fan-out bottleneck in parallel long BCH encoders*," IEEE Trans. on Circuits Syst .1, Reg. Papers,* vol.51, no.3, Mar. 2004, pp.512-516.

[4] Fengbo Liang and Liyang Pan "A CRT-based BCH encoding and FPGA implementation," *In Proc, of ICISA*, 2010, pp.1-8.

[5] K. Lee, H. G. Kang, J. I. Park, and H. Lee, "A high-speed low-complexity concatenated BCH decoder architecture for 100 Gb/s optical communications," J. of Signal Processing Systems, vol. 66, no. 1, Jan. 2012, pp. 43-55.

[6] H. Choi, W. Liu, and W. Sung, "VLSI implementation of BCH error correction for multilevel cell NAND flash memory," *IEEE Trans. on VLSI Systems*, vol.18, no.5, May 2010, pp.843-847.

[7] S.Lin and D.J. Costello Jr. *Error Control Coding*, *Prentice-Hall*, New Jersey, 1983.

[8] Hank Wallace, *Error Detection and Correction Using the BCH Code*, 2001

[9] Z. Jun. W.Z. gong, H. Q. Sheng, X. Jie, "Optimized design for high-speed parallel BCH encoder," In *Proc. of IEEE International Work- shop*, 2005, pp.97-100.

[10] A.M. Patel, "A multichannel CRC register," *in Proc. of AFIPS Conf.*, vol.38, 1971, pp.11- 14.

[11] L.V. Cargini, R.D.R. Fangundes, A.E. Bezerra and G.M. Almeida "Parallel algebraic approach of BCH coding in VHDL," in *Proc. of ICCGI*.2007, p.22.

Duk-Gyu Lim

He received the B.S., M.S in Electronic engineering from Dankook University, Korea in 1978 and 1988 respectively and also received Ph.D in Department Electronic engineering from Dankook University in 1988. Since 1985, he has been faculty of the Kangwon National University at Div. of Electronics, Information, and Communication Engineering. His research interests include digital signal processing, medical electronics.

Sharad Shakya

He received the B.S., M.S in Electronic engineering from kangwon National University, Samcheok Campus, in 2010 and 2013 respectively. His research interest includes communication systems, low power and cost efficient VLSI architectures.

Je-Hoon Lee

He received the B.S., M.S in Computer and Comm. Engineering from Chungbuk Nat'l University, Korea in 1999 and 2001 respectively and also received Ph.D in Computer and Comm. Engineering from Chungbuk Nat'l University in 2005. From 2005 to 2006, he was a visiting scholar at Univ. of Southern California, USA and from 2007 to 2008, he was a visiting scholar at Murdoch University, Australia. From 2006 to 2009, he was an assistant Professor in Chungbuk Nat'l University. He is currently an assistant Professor in Div. of Electronics, Information, and Communication Engineering of Kangwon Nat'l University. His research interests include embedded system applications high-speed and low-power circuit and system design.