2014년 10월 전자공학회 논문지 제 51권 제 10 호
Journal of The Institute of Electronics and Information Engineers Vol. 51, NO. 10, October 2014 http://dx.doi.org/10.5573/ieie.2014.51.10.072

논문 2014-51-10-11

Pipeline-Aware QC-IRA-LDPC 부호 및 효율적인 복호기 구조

(Pipeline-Aware QC-IRA-LDPC Code and Efficient Decoder Architecture)

사 부 흐*, 이 한 호**

(Sabooh Ajaz and Hanho Lee©)

요 약

본 논문은 PIPELINE-AWARE QC-IRA-LDPC (PA-QC-IRA-LDPC) 코드 생성 방법과 Rate-1/2 (2016,1008) PA-QC-IRA-LDPC 코드에 대한 효율적인 고속 복호기 구조를 제안한다. 제안한 방법은 비트 오류율 (BER) 성능 저하 없이 파이프라인 기법을 사용하여 임계경로를 나눌 수 있다. 또한 제안한 복호기 구조는 데이터 처리량, 하드웨어 효율 및 에너지 효율을 크게 향상시킬 수 있다. 제안한 복호기 구조는 90-nm CMOS 기술을 사용하여 합성 및 레이아웃이 수행되었으며, 이전에 보고된 복호기 구조들에 비해서 하드웨어 효율성이 53%이상 향상되었고, 훨씬 좋은 에너지 효율성을 보여준다.

Abstract

This paper presents a method for constructing a pipeline-aware quasi-cyclic irregular repeat accumulate low-density parity-check (PA-QC-IRA-LDPC) codes and efficient rate-1/2 (2016, 1008) PA-QC-IRA-LDPC decoder architecture. A novel pipeline scheduling method is proposed. The proposed methods efficiently reduce the critical path using pipeline without any bit error rate (BER) degradation. The proposed pipeline-aware LDPC decoder provides a significant improvement in terms of throughput, hardware efficiency, and energy efficiency. Synthesis and layout of the proposed architecture is performed using 90-nm CMOS standard cell technology. The proposed architecture shows more than 53% improvement of area efficiency and much better energy efficiency compared to the previously reported architectures.

Keywords : Low-density-parity-check (LDPC), min-sum decoder, pipeline-aware, forward error correction (FEC).

Ⅰ. Introduction

Low-density parity-check (LDPC) codes are widely used in modern communication systems. LDPC codes are one of the most important forward error correction (FEC) codes because of their remarkable performance that can approach Shannon limit. Recently, several techniques are proposed that

* 학생회원, ** 평생회원, 인하대학교 정보통신공학과
(Department of Information and Communication Engineering, Inha University)
© Corresponding Author(E-mail: hhlee@inha.ac.kr)
※ 이 논문은 2013년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(2013R1A2A2A01068628).
※ 이 논문은 인하대학교의 지원에 의하여 연구되었음.
접수일자: 2014년05월16일, 수정일자: 2014년08월29일
게재확정: 2014년10월06일

offer wide variety of tradeoffs between code performance (BER), decoder complexity and decoder throughput. These techniques can be classified into three major categories. 1) Construction of LDPC code for low complexity decoder architecture with minimal bite error rate (BER) degradation[1, 2]. These codes are known as "block structured LDPC codes" or "architecture-aware LDPC codes". 2) Improvement in decoding algorithm/schedule to achieve faster convergence and low complexity decoding. Min-Sum algorithm[2] offers low complexity compared to log belief propagation (log-BP) algorithm. Layered schedule[3] offers faster convergence compared to the conventional BP schedule. Variable node units (VNUs) are updated more often compared to BP. Hence, faster convergence is achieved and the similar BER is achieved with almost 50% of the iterations. 3) Efficient hardware design techniques such as non-uniform quantization[4], differential shifting[5], pipelining[2], the partially parallel decoder designs[6], multi-mode QC-LDPC decoder architecture[7]. The focus of this work is on the first and third category. A code construction method of pipeline-aware Quasi-Cyclic Irregular Repeat Accumulate (QC-IRA) LPDC code is described. An efficient decoder design is proposed to achieve low-latency and high-throughput without any BER degradation.

The rest of this paper is organized as follows. Section II describes a brief introduction of QC-IRA LDPC codes and decoding algorithm. Section III described the code construction and efficient decoder architecture for pipeline-aware QC-IRA-LDPC code. Performance comparison and results are presented in Section IV. Finally, conclusions are drawn in Section V.

Ⅱ. QC-IRA-LDPC CODE AND DECODING ALGORITHM

1. IRA-LDPC Code

Repeat accumulate (RA) or irregular repeat accumulate (IRA) codes can be considered to be a concatenation of a set of repetition codes with an accumulator, through an interleaver[8~9]. Despite their simple structure, they were shown to provide good performance[9] and, more importantly, they paved a path toward the design of efficiently encode-able and decode-able LDPC codes.

An RA code consists of a serial concatenation of a single rate-$1/q$ repetition code through an interleaver with an accumulator having transfer function $1/(1 \oplus D)$[8~9]. RA codes can be either non-systematic or systematic. In the first case, the accumulator output, p, is the codeword and the code rate is $1/q$. For systematic RA codes, the information word, u, is combined with p to yield the codeword $c = [u\ p]$ and so that the code rate is $1/(1 + q)$. The systematic irregular repeat accumulate (IRA) codes generalize the systematic RA codes and the repetition rate may differ across the k information bits. Irregular repeat-accumulate codes provide several advantages over RA codes. They allow both flexibility in the choice of the repetition rate for each information bit so that high rate codes may be designed and capacity is more easily approached. The parity check matrix with k information bits and m parity bits ($m = n - k$) for systematic RA and IRA codes has a following form.

$$H = [\ H_u \qquad H_p] \tag{1}$$

where H_p is a m×m "dual-diagonal" square matrix.

2. QC-IRA-LDPC Code

Given the code rate, length, and degree distributions, an IRA code is defined entirely by the matrix H_u. While a random-like H_u would generally give good performance, it is problematic for both encoder and decoder implementations. An IRA-LDPC code with structured (quasi cyclic) H_u, developed in [8, 9] actually showed a good performance compared to random like codes. QC-IRA-LDPC codes can be developed by choosing H_u to be array of circulant

permutation matrices. Let P be an L×J array of Q×Q circulant permutation matrices then H_{QC} is defined as (2).

$$HQC = [\ P \quad \Pi \quad HP\Pi T] \tag{2}$$

$$P = \begin{bmatrix} b_{0,0} & b_{0,1} & \cdots & b_{0,J-1} \\ b_{1,0} & b_{1,1} & \cdots & b_{1,J-1} \\ \vdots & \vdots & \ddots & \vdots \\ b_{L-1,0} & b_{L-1,1} & \cdots & b_{L-1,J-1} \end{bmatrix}_{L\times J} \tag{3}$$

$$\prod H_p \prod^T = \begin{bmatrix} I_0 & & & (I_1) \\ I_0 & I_0 & & \\ & \ddots & \ddots & \\ & & I_0 & I_0 \end{bmatrix}_{L\times J} \tag{4}$$

where, parity part of H_{QC}, *i.e.,* $\Pi\ \mathbf{H}_p\ \Pi^T$ is a quasi-cyclic dual diagonal matrix, with upper right entry may be included for tail-biting accumulator[8]. I_0 is a Q×Q identity matrix and I_1 is a cyclic shifted version of I_0. Generally, inverse of the both parts of H_{QC} is required to obtain generator matrix. The inverse only exists only when tail-biting is absent[8].

3. Layered Decoding Algorithm

A Min-Sum based layered decoding algorithm[2-4] is briefly described below.

$$L^l[k] = P^{l-1}[k] - R^l[k-1] \tag{5}$$

$$R^l[k] = \alpha \times \prod_{n\in N(c)/v} sign\left(L^l[k]\right) \times \min_{n\in N(c)/v} \left|L^l[k]\right| \tag{6}$$

$$P^l[k] = L^l[k] + R^l[k] \tag{7}$$

Here, l denotes the layer and k is the iteration number. R denotes the check to variable message (c to v). L represents the variable to check message (v to c). In the k-th iteration, the LLR message from layer l to next layer for variable node v is represented by P. In (6), N(c) / v denotes the set of variable nodes connected to check node c excluding variable node v. The layered decoder performs all operations from (5) to (7) in a sequence.

Ⅲ. PIPELINE-AWARE QC-IRA-LDPC CODE CONSTRUCTION

1. Pipelining Issues for LDPC Decoders

Efficient pipelining is an essential part of high throughput LDPC decoders. But in case of layered decoder, pipelining is not straight forward. A pipeline method with approximated layered decoding approach [2] causes BER degradation. Hence this technique is not much efficient in terms of BER performance. On the other hand, [10] used a multi-frame pipeline technique, which effectively doubles the memory requirement.

The main issue is the data dependency between consecutive rows in the layered decoding algorithm, because a posterior probability messages from layer "j-1" are required before starting the process for layer "j". Quasi-cyclic **H** matrix with four layers or block rows and a one stage pipeline schedule is shown in the Fig. 1. Block layer # 0 and 1 can be processed independently, because these layers have no conflicting entry. Similarly block rows # 3 and 4 have no conflicting entry. But still pipeline insertion is not possible, because column five and seven have a conflicting entry. If these two conflicting entries

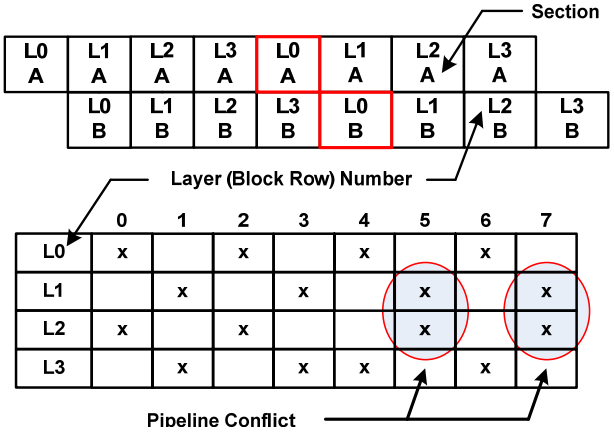


그림 1. LDPC 복호기 H-matrix 값들의 파이프라인 충돌
Fig. 1. Pipeline conflict in entries of H-matrix for LDPC decoders.

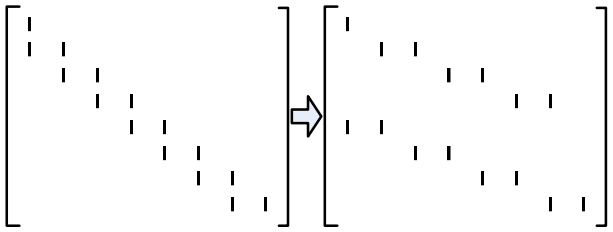
are removed by any mean, then it is possible to insert a pipeline that follows a schedule shown in Fig. 1. Hence, any LDPC code that follows the following lemma (with $i \geq 2$) is considered as a pipeline-aware LDPC code.

"If all 'i' consecutive block rows (layers) have no 1-entry in common then, 'i-1' pipeline stages can be inserted without any pipeline hazard"

2. Construction of Pipeline-Aware QC-IRA-LDPC Codes

An example of dual-diagonal 8×8 quasi-cyclic matrix is shown in Fig. 2(a). It is quite clear that it doesn't depict a pipeline-aware property. A row permutation is performed on this matrix. First all the even rows i.e. rows # 0, 2, 4, 6 are permuted to row # 0, 1, 2, 3. Then all odd rows *i.e.* rows # 1, 3, 5, 7 are permuted to row # 4, 5, 6, 7. The resulting matrix shown in Fig. 2(b) is a pipeline-aware with "$i > 2$".

The QC-IRA-LDPC code construction method described in [8, 9] has been imrpoved and tailored to achieve the pipeline-aware property described above. Let H_{QC} is defined by (2) then, systematic and parity parts are given in (3) and (4), respectively. A modified construction method for QC-IRA-LDPC code with "$i = 2$" pipeline-aware feature is described as follows.



(a) (b)

그림 2. (a) 기존의 dual-diagonal matrix
(b) 제안한 permuted pipeline-aware matrix
Fig. 2. (a) Conventional dual diagonal matrix.
(b) Proposed permuted pipeline-aware matrix.

1. Generate L×J quasi cyclic matrix H_p:
a) Initlize H_p as quasi cyclic dual diagonalmatrix..
b) Permute H_p according to the odd/ even splitting method desribed above.

2. Generate L×J quasi cyclic matrix P:
a) Randomly select j^{th} column and l^{th} row (Do not select a [j, l]-block which already has a cyclic permutation registered).
b) Check pipeline-aware condition shown in (8) and (9):

$$b_{(i+1),j} = 0_Q \tag{8}$$

$$b_{(i-1),j} = 0_Q \tag{9}$$

Here, 0_Q = Q×Q zero matrix or empty matrix.
c) If condition '2b' is true then go to step # 3, else go to step # 2.

3. Assign a cyclic permutation value to $b_{l,j}$
a) Randomly select an integer x $\in$ [0, Q), different from most recently selected integer and set $b_{l,j}$ = x.

4. Check condition 1 and 2 described below [8]
a) Condition 1: Every entry in each column of P should have a distinct cyclic permutation value.
b) Condition 2: Check all 4-cycles with in P by applying (10) and check all length-4 cycles across Pand H_p by applying (11).

$$\left(b_{l1,j1} - b_{l1,j2}\right) \bmod Q \neq \left(b_{l2,j1} - b_{l2,j2}\right) \bmod Q \tag{10}$$

$$b_{l,j} \neq b_{l+1,j} \;\&\; \left(b_{L-1,j} - b_{0,j}\right) \bmod Q \neq 1 \tag{11}$$

where,

$$l_1, l_2 \in [0, L), l_1 \neq l_2 \;\&\; j_1, j_2 \in [0, J), j_1 \neq j_2$$

and

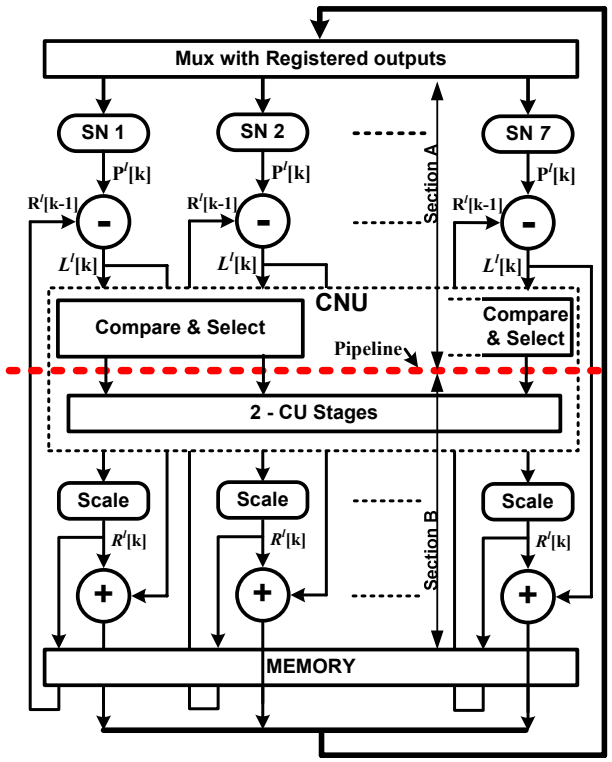
$$l \in [0, L-1) \;\&\; j \in [0, J)$$

c) If both conditions '4a' or '4b' are true, then register $b_{l,j}$ at j^{th} column and l^{th} row and go to step # 2, else go to step # 3.

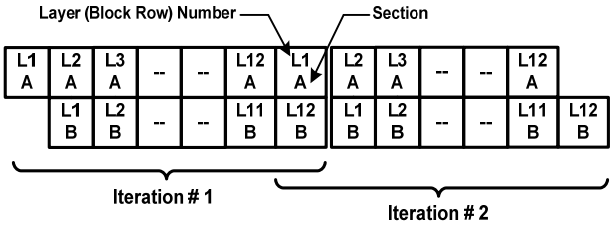
The algorithm described above avoids all 4-cycles. Hence the girth of the code is at least 6. Further improvement is possible by applying the 6-cycle condition given in [9].

3. Pipelined-Aware QC-RA-LDPC Decoder Architecture

The proposed pipeline-aware QC-IRA-LDPC decoder architecture with row weight of seven is shown in Fig. 3(a). First the switch network rotates all the input values and old check to variable values are retrieved from memory simultaneously. Then subtraction is performed according to Eq. (5). Then CNU operation is carried out. CNU performs the minimum sorting and sign operation shown in (6). Efficient CNU architecture is adopted from [11]. First stage of CNU is compared and selected, and then two CU stages are added. The pipeline stage can be freely adjusted inside CNU without any pipeline hazard because the proposed code is pipeline-aware. We have inserted the pipeline just before the CU stages to reduce the critical path. The output of CNU (first and second minimum values) or new check to variable node messages are scaled with factor α=0.75. Finally, addition operation is performed using (7), to generate new log-likelihood ratio (LLR) values. for next layer. The process continues until last layer in final iteration is reached. Fig. 3(b) shows the pipeline schedule for 12 layers. Each block in pipeline schedule is represented by a layer number and a pipeline section. The pipeline section is shown as "Section A" and "Section B" in Fig. 3(a). It is clear form pipeline schedule that two different layers are processed by "Section A" and "Section B" simultaneously. There is no pipeline hazard because pipeline-aware property made all successive layers conflict free.



(a)



(b)

그림 3. (a) 제안한 pipeline-aware LDPC 복호기 구조
(b) 제안한 LDPC 구조의 Pipeline 스케줄

Fig. 3. (a) Proposed pipeline-aware LDPC decoder architecture. (b) Pipeline schedule for the proposed LDPC decoder.

Ⅳ. RESULTS AND COMPARISONS

1. BER Performance

The proposed method was used to design rate-1/2 (2016, 1008) PA-QC-IRA-LDPC code with 12-layers or block rows, pipeline "i= 2" and "Q = 84". Fig. 4 shows the floating-point BER performance of rate-1/2 (2016, 1008) QC-IRA-LDPC code without pipeline condition and proposed pipeline-aware QC-IRA LDPC code with pipeline condition (using 10

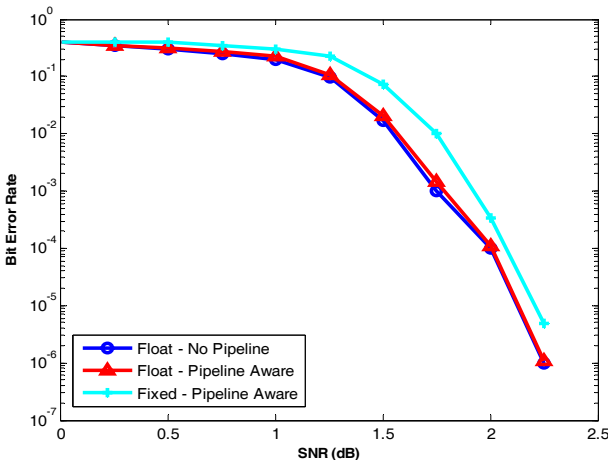


그림 4. 제안한 rate-1/2 (2016, 1008) LDPC 코드의 BER 성능
Fig. 4. BER performance of the proposed rate-1/2 (2016, 1008) LDPC code.

iterations). Fig. 4 also shows the BER performance for a proposed layered decoder with 6-bit quantization, which shows no BER degradation due to pipelining condition.

2. Implementation Results

The proposed pipeline-aware layered decoder was modeled in Verilog HDL and simulated to verify its functionality. After complete verification of the design functionality, it was then synthesized using appropriate time and area constraints. Both synthesis and layout steps were carried out using the SYNOPSYS design tool and TSMC 90-nm CMOS technology.

Table I shows the performance comparisons between the proposed pipeline-aware layered decoder architecture with various other decoders of comparable code length[12~15]. The proposed code length is 2016-bit. The nearest code lengths available in literature are 1944-bit[14], 2304-bit[12, 14] and 2048-bit[15]. LDPC decoders given in [12 - 15] implement various techniques for efficient decoding. To provide a fair comparison, the factors of area and energy efficiency are introduced. Area efficiency is measured in throughput provided for each mm^2 of area. The proposed decoder shows much better area efficiency compared to all other works provides in [12 ~15]. The factor of power efficiency shows how much joules of energy is consumed for each bit in each iteration. The factor of pj/bit/iteration is much better for comparison with various code lengths because this factor not only encompasses power but also the time required to produce a decoded output.

The proposed decoder developed with pipeline-aware technique shows a major advantage in terms of throughput with reduced area. The K. Zhang decoder[12] shows an impressive throughput of 2.2 Gb/s with 2.9 mm^2 of area. Whereas the proposed decoder shows about 40% throughput improvement

표 1. 구현 결과 및 성능 비교
Table 1. Result and performance comparison.

Design	Proposed	K. Zhang [12]	Y. Sun [13]	C. Roth [14]	M. Mansour [15]
Max. Code length	2016	2304	2304	1944	2048
CMOS tech.	90-nm	90-nm	90-nm	90-nm	180-nm
Schedule	Layered	Layered	Layered	Layered	TDMP
Clock freq. (MHz)	185	950	450	346	125
Max iterations	10	10	10	10	10
Max. throughput (Gb/s)	3.08	2.2	1.0	0.68	1.28[b]
Area (mm^2)	2.63	2.9	3.5	1.77	3.575[a]
Avg. power (mW)	510	870	410	107	787
Area eff. ($Gb/s/mm^2$)	1.17	0.76	0.286	0.38	0.358
Energy eff. (Pj/bit/itr)	16.5	39.2	-	15.8	30.75[c]

[a] Area conversion factor for 180-nm to 90-nm CMOS Technology: (180nm-Area)÷((1.414)^2)^2
[b] Throughput conversion factor for 180-nm to 90-nm CMOS Technology: (180nm-Throughput)×(1.414)^2
[c] Energy efficiency conversion factor for 180-nm to 90-nm CMOS Technology: (180nm-Energy eff.)÷((1.414)^2)^2

compared to [12]. It outperforms the decoder given in [12] by more than 53% in terms of area efficiency. The proposed decoder shows much better power efficiency primarily because it consumes less power and it requires much less time to give a decoded output. The proposed decoder only needs (layers×Iter)+i=121 clock cycles, whereas the K. Zhang decoder needs 990 clock cycles[12]. Hence it consumes much more pico joules over a period of time.

The C. Roth decoder[14] achieves less than a 1.0 Gb/s with the area lesser than the proposed decoder. But in terms of area efficiency the proposed decoder shows much better results. In terms of power efficiency both the decoders depict almost similar results, but the proposed decoder shows improved power without considering register transfer level optimizations utilized in [14]. The reduced power in the proposed decoder is due to efficient pipeline-aware technique discussed in section III.

The Y. Sun decoder [13] shows 1 Gb/s of throughput with 3.5 mm^2 of area. The proposed decoder shows a better area efficiency compared to [13]. The power efficiency depends upon the power and time required to get the decoded output. We cannot find enough information in [13] to accurately calculate the power efficiency.

The 2048-bit decoder shown in [15] is synthesized using 180-nm CMOS technology. Hence some conversion method is required to provide fair comparison between decoder [15] and the proposed decoder. We have adopted the conversion factors form in [16]. A factor of $((1.414)^2)^2 = 4$ is used for area conversion and a factor of $(1.414)^2 = 2$ is used for throughput conversion. A factor of $((1.414)^2)^2 = 4$ is used to convert power efficiency. The proposed decoder shows much better area and energy efficiency compared to [15].

V. CONCLUSION

The paper introduces a novel construction method for PA-QC-IRA-LDPC code. A pipeline stage can be inserted inside the decoder architecture without BER degradation. Moreover, the proposed decoder offers an acceptable increase in throughput with reduced area, compared to the other decoders with comparable code length. The proposed decoder shows much better area and energy efficiency compared to other decoders with similar code lengths available in literature. The proposed techniques are expected to be incorporated in next-generation high efficiency QC-LDPC decoder design.

REFERENCES

[1] M. Mansour and N. Shanbhag, “High-throughput LDPC decoders,” IEEE Trans. VLSI Systems, vol. 11, no. 6, pp. 976-996, Dec. 2003.

[2] Z. Cui, Z. Wang, Y. Liu, “High-throughput layered LDPC decoding architecture,” IEEE Trans. VLSI Systems, vol. 17, no. 4, pp. 582-587, Apr. 2009.

[3] J. Zhang, M Fossorier, “Shuffled Iterative Decoding”, IEEE Trans. on Comm., vol. 53, no. 2, pp. 209-213, Feb. 2005.

[4] D. Oh, K. Parhi, “Min-Sum decoder architecture with reduced word length for LDPC Codes”, IEEE Trans. Circuits Syst. I, vol. 57, no. 1, pp. 105-115, Jan. 2010.

[5] S. Kim, G. E. Sobelman, H. Lee “A Reduced-Complexity Architecture for LDPC Layered Decoding Schemes,” IEEE Trans. on VLSI Systems, vol. 19, no. 6, pp. 1099-1103, Jun. 2011.

[6] L. Liu and C.-J. R. Shi, “Sliced message passing: High throughput over- lapped decoding of high-rate low-density parity-check codes,” IEEE Trans. Circuits Syst. I, vol. 55, no. 11, pp. 3697-3710, Nov. 2008.

[7] H. Lee, S. Ajaz, "A High-Throughput QC-LDPC Decoder Architecture for Multi-Gigabit WPAN Systems," Journal of The Institute of Electronics Engineers of Korea Vol. 50, No. 2, pp. 105-113, Feb. 2013.

[8] L. Gianluigi, S. Song, L. Lan, Y. Zhang, S. Lin and W.E Ryan "Design of LDPC Codes: A Survey and New Results", J. of Communication Software and Systems, vol. 2, pp. 191-211, Sep. 2006.
[9] Y. Zhang, W.E. Ryan, "Structured IRA Codes: Performance Analysis and Construction", IEEE Trans. on Comm., vol. 55, no. 5, pp. 837-844, May 2007.
[10] A. Darabiha,, A. C. Carusone,, F. R. Kschischang, "Block-Interlaced LDPC Decoders with Reduced Interconnect Complexity", IEEE Trans. Circuits Syst. II, Express Brief, vol.55, no. 1, pp. 74-78, Jan. 2008.
[11] C. Wey, M. Shieh, S. Lin, "Algorithms of Finding the First Two Minimum Values and Their Hardware Implementation", IEEE Trans. Circuits Syst. I, vol.55, no. 11, pp. 3430-3437, Dec. 2008.
[12] K. Zhang, X. Huang, Z. Wang, "High-throughput layered decoder implementation for quasi-cyclic LDPC codes", IEEE J. on Sel. Areas in Comm., vol. 27, no. 6, pp. 985-994, Aug. 2009.
[13] Y. Sun and J. R. Cavallaro, "A low power 1-Gbps reconfigurable LDPC decoder design for multiple 4G wireless standards," IEEE International SOC Conference, pp. 367-370, Sept. 2008.
[14] C. Roth, P. Meinerzhagen, C. Studer. and A. Burg, "A 15.8 pJ/bit/iter Quasi-Cyclic LDPC Decoder for IEEE 802.11n in 90 nm CMOS," IEEE Asian Solid State Circuits Conference, pp. 1-4, Nov. 2010.
[15] M. M. Mansour and N. R. Shanbhag, "A 640-Mb/s 2048-Bit Programmable LDPC Decoder Chip," IEEE Journal of Solid-State Circuits, vol. 41, pp. 684 - 698, Mar. 2006.
[16] J. R. Hauser, "Handbook of Semiconductor Manufacturing Technology," editor R. Doering and Y. Nishi, CRC Press, Bosa Roca, 2007, 2nd ed. 1-21.

저 자 소 개

Sabooh Ajaz(학생회원)
2006년 NED University of Engg. &Tech. Karachi, 전자공학과 학사 졸업
2010년 University of Wollongong, Australia, Electrical Computer and Telecom Engineering 석사 졸업
2011년～현재 인하대학교 정보통신공학과 박사 재학
<주관심분야 : 오류정정 아키텍처 설계>

이 한 호(평생회원)-교신저자
1993년 충북대학교 전자공학과 학사 졸업
1996년 Univ. of Minnesota 전기 컴퓨터공학 석사 졸업
2000년 Univ. of Minnesota 전기 컴퓨터공학 박사 졸업
2000년～2002년 Member of Technical Staff, Lucent Technologies (Bell Labs.), USA.
2002년～2004년 Assistant Prof. Dept. of Electrical and Computer Engineering, Univ. of Connecticut, USA.
2004년～현재 인하대학교 정보통신공학과 교수
<주관심분야 : 디지털신호처리 및 오류정정 아키텍처 설계>