Multi-column parallel QC-LDPC decoder architecture for NAND flash memory

Wei Shen[1], **Cheng Chen**[2], **and Jin Sha**[3a)]
[1] *School of Electrical Science and Engineering, Nanjing University, Nanjing 210046, People's Republic of China*
[2] *Microelectronics R&D Institute, ZTE Corporation, Nanjing 210046, People's Republic of China*
[3] *Shenzhen Research Institute, Nanjing University, Nanjing 210046, People's Republic of China*
a) *shajin@nju.edu.cn*

Abstract: Quasi-cyclic (QC) low-density parity-check (LDPC) codes are famous for their excellent error correction performance and hardware friendly structure in NAND flash memory application. Array LDPC code is a type of highly structured QC-LDPC code that provides a good balance between performance and complexity. In this paper, a method is proposed for the construction of (18900, 17010) LDPC code that is based on the Latin square and an improved array dispersion strategy to achieve multi-column alignment of the structure. Compared with traditional design, the parallel hardware architecture reduces the number of barrel shifters by 32%. The corresponding ASIC implementation results show that the throughput of the proposed QC-LDPC code was up to 3.49 Gb/s and the throughput-to-area (TAR) of the proposed codes was significantly improved.

Keywords: QC-LDPC, NAND flash memory, column-based shuffle decoding, multiple-columns
Classification: Integrated circuits

References

[1] G. Dong, *et al.*: "Using data postcompensation and predistortion to tolerate cell-to-cell interference in MLC NAND flash memory," IEEE Trans. Circuits Syst. I, Reg. Papers **57** (2010) 2718 (DOI: 10.1109/TCSI.2010.2046966).
[2] K. Kim: "Technology for sub-50 nm DRAM and NAND flash manufacturing," IEEE International Electron Devices Meeting, 2005. IEDM Technical Digest (2005) 323 (DOI: 10.1109/IEDM.2005.1609340).
[3] J. Li, *et al.*: "Realizing unequal error correction for NAND flash memory at minimal read latency overhead," IIEEE Trans. Circuits Syst. II, Exp. Briefs **61** (2014) 354 (DOI: 10.1109/TCSII.2014.2312640).
[4] R. Gallager: "Low-density parity-check codes," IRE Trans. Inf. Theory **8** (1962) 21 (DOI: 10.1109/TIT.1962.1057683).
[5] M. P. C. Fossorier, *et al.*: "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," IEEE Trans. Commun. **47** (1999) 673 (DOI: 10.1109/26.768759).

© IEICE 2018
DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018

[6] K. C. Ho, *et al.*: "A 520k $(18900, 17010)$ array dispersion LDPC decoder architectures for NAND flash memory," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **24** (2016) 1293 (DOI: 10.1109/TVLSI.2015.2464092).
[7] H. C. Lee, *et al.*: "Optimization techniques for the efficient implementation of high-rate layered QC-LDPC decoders," IEEE Trans. Circuits Syst. I, Reg. Papers **64** (2017) 457 (DOI: 10.1109/TCSI.2016.2612655).
[8] J. Li, *et al.*: "Algebraic quasi-cyclic LDPC codes: Construction, low error-floor, large girth and a reduced complexity decoding scheme," IEEE Trans. Commun. **62** (2014) 2626 (DOI: 10.1109/TCOMM.2014.2339329).
[9] M. R. Li, *et al.*: "A low-complexity LDPC decoder for NAND flash applications," 2014 IEEE International Symposium on Circuits and Systems (ISCAS) (2014) 213 (DOI: 10.1109/ISCAS.2014.6865103).
[10] E. Eleftheriou and S. Olcer: "Low-density parity-check codes for digital subscriber lines," IEEE Communication Letters (2002) 1752 (DOI: 10.1109/ICC.2002.997149).
[11] W. Shao, *et al.*: "Dispersed array LDPC codes and decoder architecture for NAND flash memory," IEEE Trans. Circuits Syst. II, Exp. Brief (DOI: 10.1109/TCSII.2017.2783438).

1 Introduction

NAND flash memories are widely used for storage in mobile devices. Owing to the growing demand for high-density storage capacity and throughput, multi-level cell (MLC) and trinary-level cell (TLC) [1, 2] techniques are used in flash memory. However, data reliability is reduced due to higher raw bit error rates (RBER) introduced by the increasing bit density [3].

Error correction codes (ECC), such as BCH codes and LDPC codes are efficient approaches to guarantee the data reliability. Compared with BCH codes, LDPC codes yield superior error correction performance with parity bits of the same size [4, 5]. QC-LDPC code is known for its hardware-friendly structure and excellent error correction performance. Several studies contributed on the implementation of the QC-LDPC codes [6, 7, 8, 9], including improving the decoding algorithm and optimizing hardware architecture. Few studies, however, have focused on increasing the throughput of the decoding architecture. In this paper, we aim at developing a parallel decoding hardware architecture to meet the throughput requirements of both the Toggle DDR 2.0 and the ONFI 3.0 NAND interfaces [10].

The remainder of this paper is organized as follows. Section 2 introduces the column-based shuffle decoding algorithm, along with the challenges to parallel implementation. We also propose here the construction of a $(18900, 17010)$ QC-LDPC code. In Section 3, the overall parallel architecture and the results of its implementation are detailed. Finally, we offer our conclusions in Section 4.

2 Code construction and decoding algorithm

2.1 Column-based shuffle decoding (CBSD) algorithm

An $(N, N-M)$ LDPC code represents the parity-check matrix H of size $M \times N$. At the wth iteration, the check-to-variable (C2V) message from check node (CN) c to variable node (VN) v and the variable-to-check (V2C) message from VN v to CN c

DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018

are denoted by L^w_{cv} and L^w_{vc} respectively, where $c = 0, 1, \ldots, M-1$, and $v = 0, 1, \ldots, N-1$. Note that $N(c)$ is the set of VNs connected to CN c and $M(v)$ the set of CNs connected to VN v. The VNs are divided equally into G groups and each group N_g contains N/G VNs, where $g = 0, 1, \ldots, G-1$. The following shows the message update between the CNs and VNs in the gth group in the wth iteration.

1. *VNU operation*: The V2C message L^w_{vc} is updated as in Eq. (1).

$$L^w_{vc} = L_{init,v} + \sum_{c' \in M(v) \backslash c} L^{w-1}_{c'v} \tag{1}$$

$L_{init,v}$ is the initial channel value of VN v.

2. *CNU operation*: The C2V message L^w_{cv} is updated as in Eq. (2), and a represents the scaling factor. N_g is the set of VNs located in the gth groups for $g = 0, 1, \ldots, G-1$. N_L is the union of $N_0, N_1, \ldots, N_{g-1}$, and N_R is the union of $N_g, N_{g+1}, \ldots, N_{G-1}$.

$$\begin{aligned} L^w_{cv} = a \times \prod_{v' \in N(c) \cap N_L} sgn(L^w_{v'c}) \prod_{v' \in N(c) \cap N_R \backslash v} sgn(L^{w-1}_{v'c}) \\ \times \min\Big(\min_{v' \in N(c) \cap N_L} |L^w_{v'c}|, \min_{v' \in N(c) \cap N_R \backslash v} |L^{w-1}_{v'c}| \Big) \end{aligned} \tag{2}$$

The posteriori probability $L_{app,v}$ is computed as in Eq. (3) and used for hard decision.

$$L_{app,v} = L_{init,v} + \sum_{c' \in M(v)} L^{w-1}_{c'v} \tag{3}$$

2.2 Parallelism analysis

Ho *et al.* proposed a top-down design that encapsulates optimization as well as hardware implementation. The architecture can achieve a high throughput with the degree-of-parallelism (DOP) at one. The matrix H^{QC} consists of four column groups as shown in Fig. 1. Note that each element of $\{\alpha^A, \alpha^B, \alpha^C, \alpha^D\}$ represents the submatrix and each element of $\{A, B, C, D\}$ represents the shifting value of the corresponding submatrix cycle shifted by an identity matrix of size z. The submatrix is an all-zero matrix when the element is zero. Barrel shifters are widely used to align the C2V messages of column groups. The C2V and V2C messages in

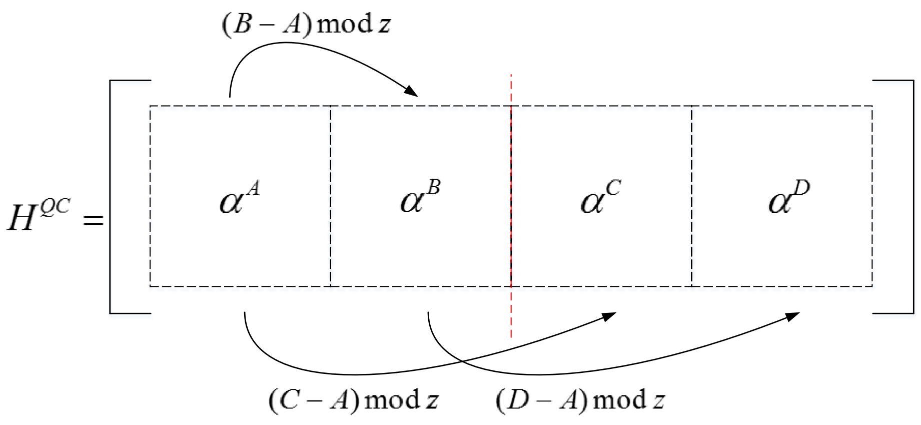

Fig. 1. Shifting parameter of barrel shifters

© IEICE 2018
DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018

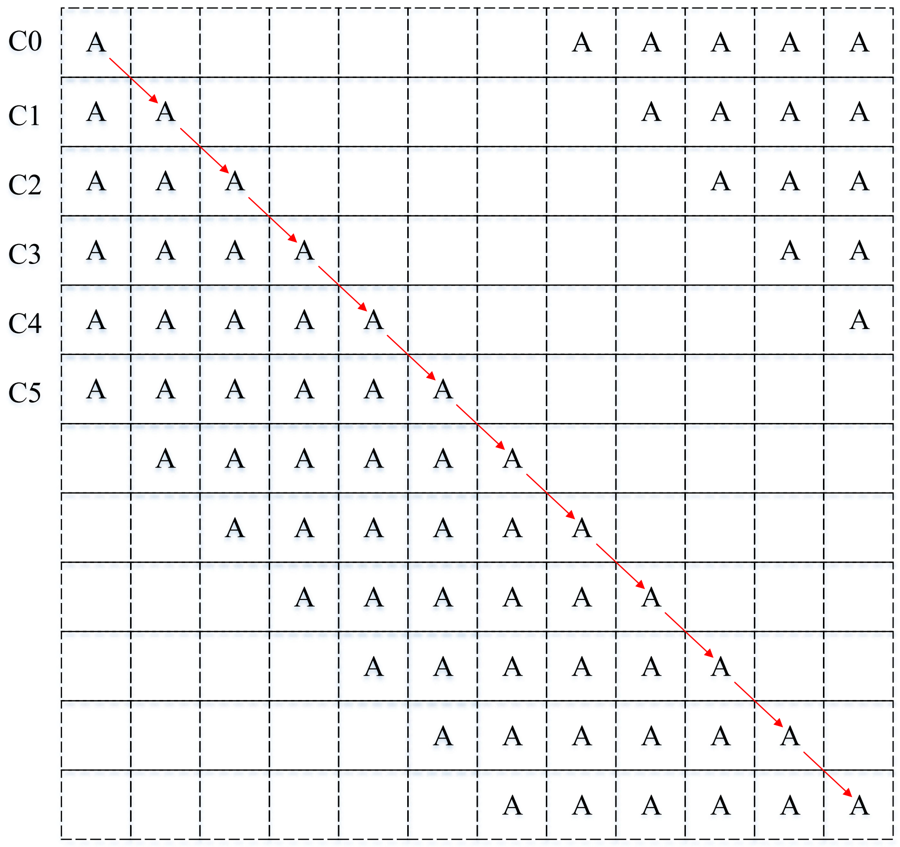

Fig. 2. Behavior of CNU block processing units

these four column groups are updated one by one. To align α^A with α^B, the shifting parameter of the barrel shifter should be set to $(B - A) mode\, z$. When the DOP is increased to two, the barrel shifter should not only align α^A and α^B, but also align α^C, α^D and α^A with the shifting parameters of $(C - A) mode\, z$ and $(D - A) mode\, z$ in once iteration. The traditional design faces challenges to the implementation of high-DOP architectures.

Fig. 2 shows the behavior of check-node-unit processing (CNU). The traditional code and the corresponding architecture with only six CNU blocks are not suitable to simultaneously update two columns.

2.3 Construction of parallel dispersed array QC-LDPC

We adopt the Latin square algorithm to construct a base matrix W^a of size $z \times z$, where z is $2^q = 64$. The entries $A_{i,j}$ of W^a are the elements of $GF(2^q)$. For $0 \leq i \leq 2^q - 1$ and $0 \leq j \leq 2^q - 1$, $A_{i,j}$ represents a submatrix that is either an all-zero matrix or a cyclic-shift of an identity matrix of size $2^q - 1$. We can illustrate the construction of the code as follows.

- *Step 1:* Select the upper-left corner of matrix W^a to construct a base matrix H^b as in Eq. (4), of size $r \times s$, where $r \leq 2^q$, $s \leq 2^q$, and set $r = 6$ and $s = 60$.

$$H^b = \begin{bmatrix} A_{0,0} & A_{0,1} & \cdots & A_{0,s-1} \\ A_{1,0} & A_{1,1} & \cdots & A_{1,s-1} \\ \vdots & \vdots & \ddots & \vdots \\ A_{r-1,0} & A_{r-1,1} & \cdots & A_{r-1,s-1} \end{bmatrix} \tag{4}$$

© IEICE 2018
DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018

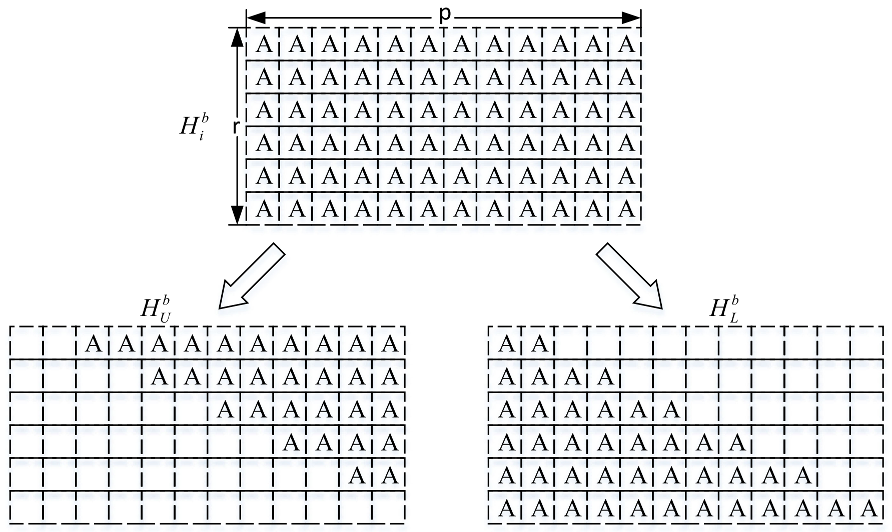

Fig. 3. Separation of matrix H_i^b

- *Step 2:* Divide the matrix into five submatrices by column, $H^b = [H_0^b, \ldots, H_4^b]$. Each submatrix as in Eq. (5) is a matrix of size (r, p), and $p = 2r$.

$$H_i^b = \begin{bmatrix} A_{0,0} & A_{0,1} & \cdots & A_{0,p-1} \\ A_{1,0} & A_{1,1} & \cdots & A_{1,p-1} \\ \vdots & \vdots & \ddots & \vdots \\ A_{r-1,0} & A_{r-1,1} & \cdots & A_{r-1,p-1} \end{bmatrix} \tag{5}$$

- *Step 3:* Instead of dividing H_i^b diagonally [6], we separate the lower left corner and upper right corner of each H_i^b by two-column-aligned strategy as Fig. 3 to generate the H_U^b and H_L^b with the same dimension of H_i^b. Except the element from the H_i^b, other elements of H_U^b and H_L^b are filled with zeros. The zeros represent a $(2^q - 1, 2^q - 1)$ all-zero submatrix.
- *Step 4:* Apply array dispersion to expand H_U^b and H_L^b to a large matrix. Repeat and combine H_U^b and H_L^b t times to generate the diagonal-like structure matrix H_i^t of size $(t \times r, t \times p)$ as in Eq. (6), where $t = 5$. As in *Step3*, the extra positions are filled with zeros, and each represents an all-zero matrix of the same size as H_U^b and H_L^b.

$$H_i^t = \begin{bmatrix} H_L^b & 0 & 0 & 0 & H_U^b \\ H_U^b & H_L^b & 0 & 0 & 0 \\ 0 & H_U^b & H_L^b & 0 & 0 \\ 0 & 0 & H_U^b & H_L^b & 0 \\ 0 & 0 & 0 & H_U^b & H_L^b \end{bmatrix} \tag{6}$$

- *Step 5:* Connect the five diagonal-like matrices and obtain the final parity check matrix H of size $(18900, 1890)$ as in Fig. 4.

The Latin matrix W^a satisfies row-column-constraint that no two rows and columns have 1 at the same place [11]. H_U^b and H_L^b decomposed from H^b as in Fig. 3 do not violate the constraint. Similar to [6], there is only one H_U^b and H_L^b

© IEICE 2018
DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018

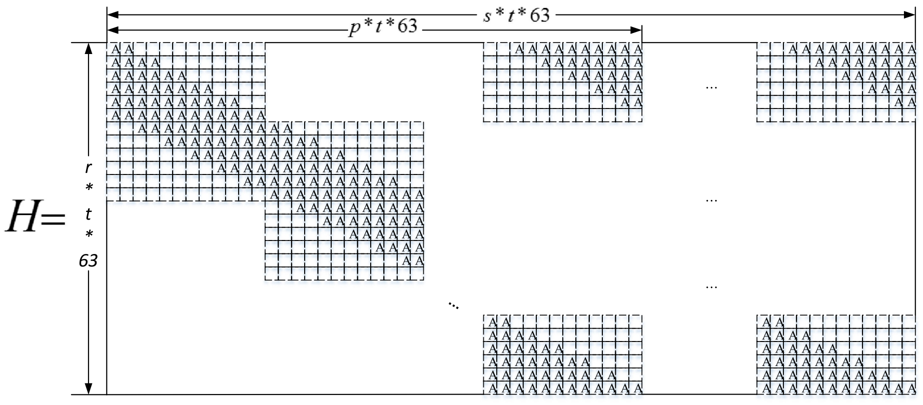

Fig. 4. H

in each column or row in expanded matrix. Therefore, matrix H also contains no 4-cycle after array dispersion.

3 Proposed architecture and the results of emulation

3.1 Overall parallel architecture

Fig. 5 shows the architecture of the multi-column parallel decoder, and Fig. 6 shows a CNU and a data selector unit (DSU) in detail. There are only two VNU blocks, six CNU blocks, and 10 DSU blocks in total. Each CNU block contains 63 sorters units, 63 sign update units, and four barrel shifters for message and sign bits. A DSU contains a barrel shifter and 63 minimum selectors. Data flow starts from

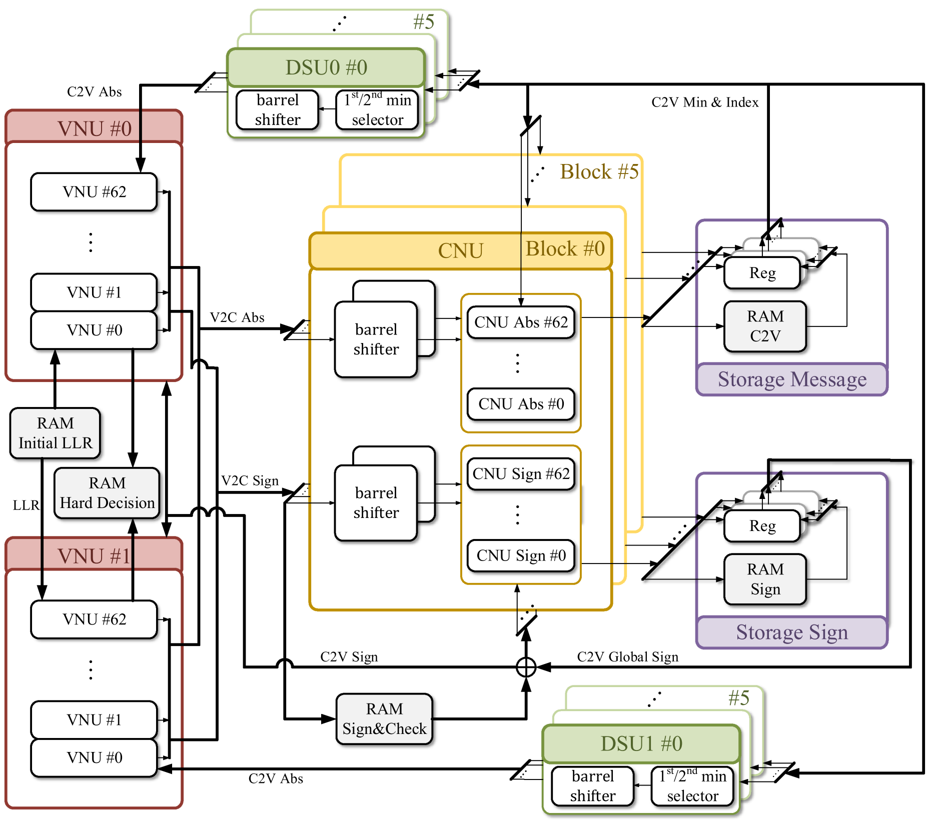

Fig. 5. Parallel decoder architecture for QC-LDPC

© IEICE 2018
DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018

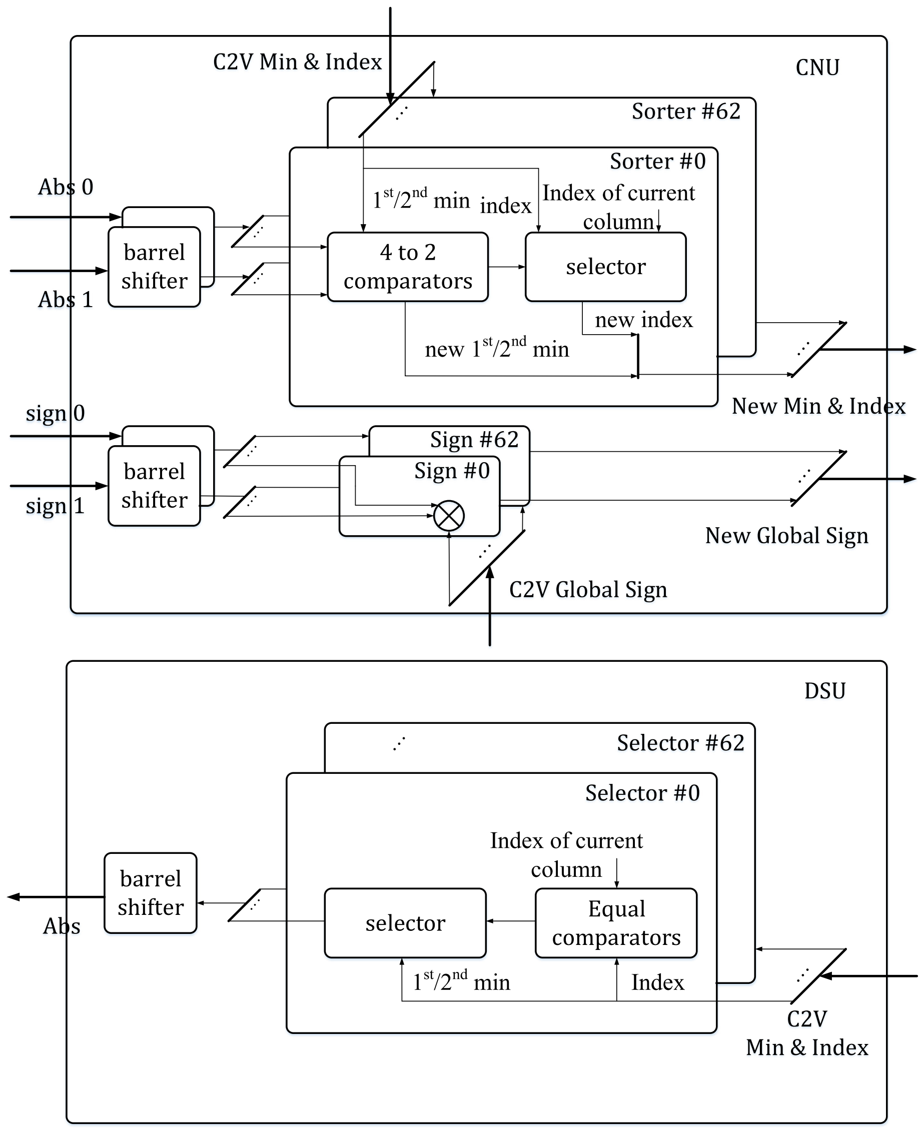

Fig. 6. Architecture of a CNU and a DSU block in detail

the storage component and passes through the DSU block, the VNU block, the CNU block, and is finally written back to memory.

As is shown in Fig. 6, the DSU updates the minimum value of the submatrix according to the entered C2V message and the index of the given column. Prior to being transferred to the VNU, the selected minimum values are aligned in the barrel shifter the shifting value of which is equal to $A_{i,j}$. The CNU updates the C2V data according to the V2C messages from the VNUs and the prior C2V messages. The shifting value of each barrel shifter in the CNU is $63 - A_{i,j}$ to align the different V2C messages. Each CNU sorter contains one four-to-two comparator to generate the first and second minimum values. Each sign update unit gets a new global sign. Compared with [6], the proposed architecture costs twice selector resources and VNUs.

3.2 Barrel shifters and storage memory

In traditional architectures [6], barrel shifters are present only in CNUs, and need to process 19 bits, consisting of two C2V messages (three bits per value), two address

© IEICE 2018
DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018

Table I. Comparison of barrel shifters

		This work	TVLSI'16 [6]
CNU	width	4	19
	number	12×63	6×63
	total	3024	7182
DSU	width	3	N/A
	number	10×63	N/A
	total	1890	N/A
total		4914	7182

messages (six bits per value), and a global sign bit. The parallel architecture aligns the selected minimum value in the DSU first, and the V2C messages and sign bits in the CNU next. The width of each barrel shifter in the DSU is three bits message, and each barrel shifter in CNU processes three bits message and a sign bit. Table I lists a comparison of barrel shifters between the proposed architecture and that in [6].

Table II shows storage usage in the decoding architecture. Before decoding begins, the initial LLR message and the parity-check matrix are stored into a single-port RAM and ROM, respectively. Compared with [6], a hard-decision RAM and a sign-and-check RAM are twice the width and half the depth in the proposed architecture. The hybrid storage architecture containing a two-port RAM and six registers blocks is also adopted.

Table II. Width and depth of ROMs and RAMs

	Type	Width (bit)	Depth
Matrix ROM	single-port ROM	72	150
Initial LLR RAM	single-port RAM	252	150
C2V RAM	dual-port RAM	1260	24
Sign-&-check RAM	dual-port RAM	882	150
Hard-decision RAM	single-port RAM	126	150

3.3 Results of emulation

We implemented the proposed architecture on the Xilinx Virtex UltraScale XCVU440 FPGA emulation platform. The maximum number of iterations was set to 20 and early termination was adopted.

For comparison, we constructed an $(18900, 17010)$ LDPC code as in [6] without masking. The results of simulation with different DOPs are shown in Fig. 7. The FER and BER of both hard-one-bit and soft-two-bit decisions of the initial LLR were considered. The results show that the parallel architecture did not sacrifice correction performance in terms of BER.

Table III lists comparisons with several related decoding architectures implemented in the TSMC 90 nm library. The results show that the proposed architecture

© IEICE 2018
DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018

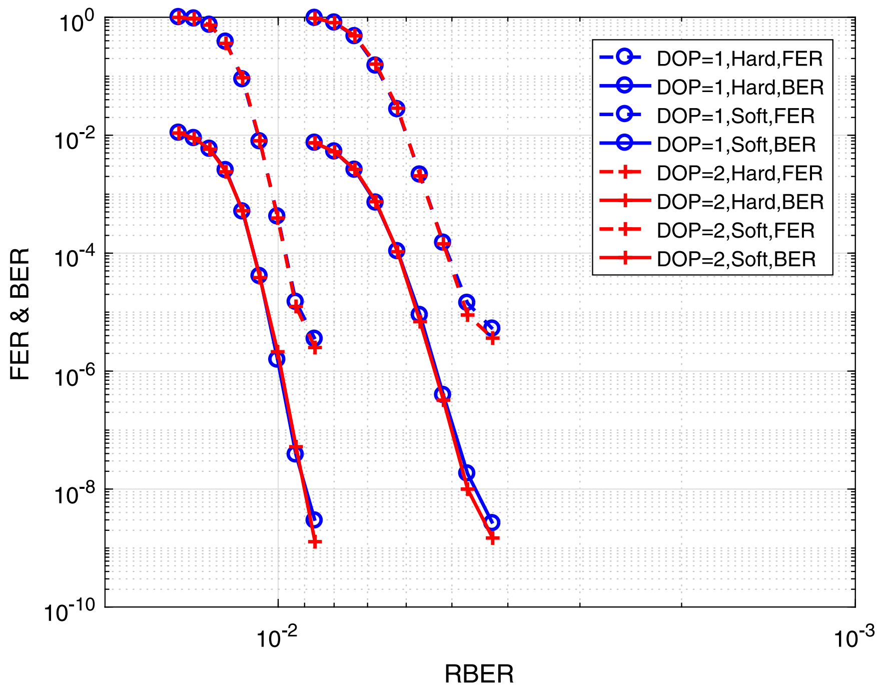

Fig. 7. Performance comparison in terms of FER and BER between DOP = 1 and DOP = 2 of the proposed (18900, 17010) QC-LDPC code over the AWGN channel. The number of quantization bits of the intermediate message was 4 and the scaling factor was 0.75

Table III. Comparision results

	This work [Proposed]	[6] K. Ho TVLSI'16	[11] W. Shao TCAS-II'17	[7] H. Lee TCAS-I'17
Impl.	post-layout	post-layout	post-layout	post-layout
Schedule	Shuffled	Shuffled	Shuffled	Layered
Iterations	6	6	6	8
Quant.	4	4	4	5
Technology	90 nm	90 nm	90 nm	90 nm
Code length	18900	18900	18300	18396
Code rate	0.9	0.9	0.897	0.905
Freq. (MHz)	166	166	200	166
Gate count	732k	520k	410k	926k
Area (mm^2)	2.93	2.56	1.44	4.19
Throughput (Gbps)	3.49	1.58	1.83	4.25
*TAR	1.19	0.62	1.27	1.01

*Throughput-to-area ratio, TAR = (Throughput)/(Area)

achieved a throughput of 3.49 Gb/s under an average number of six iterations and a clock frequency of 166 MHz. For fair comparison, we considered normalized throughput by computing the throughput-to-area (TAR). Compared with related work, the TAR of the proposed architecture was improved from 0.62 to 1.19.

© IEICE 2018
DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018

4 Conclusion

In this paper, a 2 KB-long QC-LDPC was proposed based on the Latin square and a new array dispersion method. With its diagonal-like structure, the parallel architecture reduces the number of barrel shifters compared with the traditional design. The results of simulations show that the TAR of the proposed codes is significantly improved.

Acknowledgments

This work was supported by the National Key *R&D* Program of China (No. 2016-YFA0202102), the National Natural Science Foundation of China under Grant No. 61370040, the project on the Industry Key Technologies of Jiangsu Province, BE2017153, Shenzhen science and technology project JCYJ20170818110714340 and a Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

DOI: 10.1587/elex.15.20180397
Received April 16, 2018
Accepted April 24, 2018
Publicized May 9, 2018
Copyedited May 25, 2018