A Novel Data Packing Technique for QC-LDPC Decoder Architecture applied to NAND flash controller

Longyu Ma
Department of Computer Science
The University of Auckland
Auckland, New Zealand
lma792@aucklanduni.ac.nz

Hong-fu Chou
Department of Computer Science
The University of Auckland
Auckland, New Zealand
exhan100chou@gmail.com

Chiu-Wing Sham
Department of Computer Science
The University of Auckland
Auckland, New Zealand
b.sham@auckland.ac.nz

***Abstract*—This paper presented a data packing technique for Quasi-Cyclic LDPC codes decoder applied to NAND flash controller which has the great challenge of implementing long code length to cause high complexity and routing congestion while doing floor planning. Firstly, we introduce a proposed shift network to reorder the channel data sequence. This leads to a generic data packing architecture for the LDPC decoder. Secondly, the proposed LDPC architecture can overcome the complicated routing problem caused by random accessing of massage passing within the LDPC decoder. Based on the proposed architecture, the hardware description has been synthesized on Design Compiler using a TSMC 0.18um model and provide FPGA-based floor planning with the implementing result. Synthesis results also show that the area and throughput of the proposed decoder has desirable performance for the application of NAND flash controller.**

***Index Terms*—NAND flash, LDPC code, Solid State Disc**

I. Introduction

The market demand for non-volatile NAND flash memory has been increasing for a long time, since the development of new applications that are not depend on heavy hard disks and has been rapidly growing. Flash memory [1] serves the main non-volatile storage device, and the flash interface unit is applied for system on chip (SoC) products. Flash memory also provides a low power solution for storage systems and, it is worth mentioning the small size and the light form factor are the essential properties for this type of storage. The basic flash commands are provided by which can be used by the main central processing unit(CPU) to access data from the flash memory. The flash memory is also used to initiate the boot process from the firmware and plays an important role for the storage device to execute the tasks which be performed by the main CPU. The flash interface unit has mainly provided a reliable component for graphics and multimedia processors and have been applied to digital televisions, car navigation systems, and mobile applications. In order to support multimedia applications, flash controller units have been optimized for large block read and write, as presented in [2] which provide the minimization of the main CPU interaction and supports direct memory access (DMA) [3] when transferring from the flash memory to the system DRAM memory. In the SoC applications, all of the boot information is generally stored in the flash memory. The flash memory includes a number of partitions for the boot loader code and the flash file system is created in the flash memory. All of the boot information is generally stored in the flash memory. As discussed in [4], the DMA collaborate with the error control coding (ECC) block. The ECC engine is a critical issue regarding system performance which leads to area and cost is dominated by the ECC decoder and comprising a high percentage of the flash controller. Quasi-Cyclic (QC) LDPC code [5] reduce the hardware implementation and achieve a desirable decoding performance comparing to computer generated random codes, since QC-LDPC code still has the random property within based matrix using cyclic shifted ordering to construct the code. To consider a more efficient approach, packing multiple passing messages into a single memory word benefit to the configurable datawidth and depth of embedded memory in an FPGA which is able to maximize the decoding throughput. Authors in [6] to improve the implementation of LDPC decoder has been proposed, they introduces some key challenges such as memory access provides multiple messages and duplication of the functional units to process the messages concurrently. Moreover, data alignment hardware needs to route the messages along with the appropriate functional unit which is required. However, it is difficult to manage well in randomized LDPC messages passing network. A generic methodology of message packing technique need to address of effective data process. Based on the configuration of block RAM, we follow the thread in [6] and develop a different aspect on viewing the randomized property which reorder the input message sequence of decoder by the order of wordlength of block RAM. This approach can alleviate the disorder of randomized shifting property while decoding the message passing algorithm. This is able to reduce routing overhead and redundant processing time for check node computation.

In this paper, we proposed a generic data packing technique which is well-fitted to our proposed architecture and take advantage of the configurable datawidth and depth of embedded

memory to provide flexible partial parallel decoder for solving the complexity of routing cost. In Section II, we provide our proposed data packing technique. In Section III, the proposed LDPC architecture is presented. In Section IV, the Conclusion is presented.

II. The Proposed Data Packing Technique

The $M \times N$ parity matrix, denoted as H, of a QC-LDPC code is set as an $m \times n$ array of $p \times p$ sub matrices which each sub matrix is either a $p \times p$ zero matrix or a cyclic-shifted identity matrix. The parity matrix is divided into n block columns and m block rows as follows.

$$H = \begin{bmatrix} \mathbf{A}_{0,0} & \mathbf{A}_{0,1} & \dots & \mathbf{A}_{0,n-1} \\ \mathbf{A}_{1,0} & \mathbf{A}_{1,1} & \dots & \mathbf{A}_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{m-1,0} & \mathbf{A}_{m-1,1} & \dots & \mathbf{A}_{m-1,n-1} \end{bmatrix} \quad (1)$$

A binary n-tuple $\mathbf{v} = (v_0, v_1, \dots, v_{N-1})$ is a codeword of QC-LDPC code if and only if $vH^T = 0$ and in an extension form as

$$v_0 h_{i,0} + v_0 h_{i,1} + \dots + v_{N-1} h_{i,N-1} = 0 \quad (2)$$

For each ith row of H, the operation of $\mathbf{v}H^T = 0$ provide a modulo-2 check-sum. Regarding to the check-sum operation, this is the essential computation for check node. As shown in the row of parity matrix H, the ith check node is required to compute the message on the jth variable node while $h_{ij} = 1$. Regarding to the check node processing, the all of the jth variable node related to ith check node should be accessed by the given memory address. However, the partial parallel architecture is limited to the wordlength configuration of a block RAM. Without carefully design to the data alignment, randomized QC-LDPC shifted property may cause inferior processing organization. Therefore, the nature order of message is required to be modified in order to fit the randomized QC-LDPC shifted property. Our proposed approach is shown that the circular shifted operation can effectively benefit to the randomized message passing. After shifting with the order of the depth of RAM, each bundle of message stored within individual memory address can easily be matched to the identical check node. As a result, the wider word configuration of block RAM can provider better throughput of QC-LDPC decoder under a higher parallelism architecture.

While the receiving messages are stored into LLR RAM, the sequence of messages needs to reorder for the data packing technique. The reordering is related to the depth of RAM for each address which store a package of message. For example, the depth of RAM is l and each address in the RAM can have p/l massage in a bundle. We should note that l should be a factor of p. As a result, the block interleaver [7] for the reordering is presented as $\pi(i)$, where $0 <= i < N$.

$$\pi(i) = (p/l) * (i \ mod \ l) + \lfloor (i/l) \rfloor \quad (3)$$

where the operation $\lfloor \ \rfloor$ present the nearest integer less than i/l . In Fig. 1, we illustrate a simple case of $p = 16$ and $l = 4$ and than we pack the messages from Address 0 to 3. As shown in Fig. 1 (A), the natural order of packing the messages into the memory. After interleaving the messages sequence, the messages can be packed into memory as shown in Fig. 1 (B). The proposed data packing technique rely on the reordering data sequence to pack the messages to the identical address of BRAM and pop out while the message passing in order to ensure the messages are all sorted to the identical parity check node. As shown in Fig. 2, the natural order messages pack into the example of memory which result in the mismatching messages of the address $Addr0$ to the identical check node. This lead to redundant time of accessing the address $Addr1$ to get the message 12. If the order of our design is getting larder, this kind of disorder will deteriorate the performance of LDPC decoder. As a result, this approach alleviate redundant time waiting and buffer the messages to register which allow us to compute the check node processing instantly while using the partial parallel architecture. As an example in Fig. 3, check node process can take the calculation to the bundle of messages after circular shifting. Since the message sequence is reordered according to the wordlength of BRAM, the randomized cyclic shifted property is unable to affect our proposed architecture which leads to effectively computing and less routing overhead.

III. The Proposed LDPC decoder architecture and the Synthesis Result

The proposed LDPC decoder is presented in Fig. 4 which take an example of code length 10240 and (40,4) QC-LDPC code. The QC-LDPC code is randomly generated by computer seed for the low error floor orientation. The parity length is 1024 with code rate 0.9. We use layer min-sum decoding for 4 layers and maximum iteration is set to 8. We proposed a 5 groups of combining variable node process(VNP) and check node process(CNP). In each group, there are 32 VNP and 16 CNP for computing messages passing operation. After that, messages are stored into RAM and pop out accordingly to the VNC. For our generic proposed architecture, the CNP only need to rely on simple circular shifter to match a bundle of messages data on the corresponding party check ensemble. After CNP computing, the messages are passing to CMPtop to make a decision on selecting the minimum value. The controller provide the control signal to maintain the execution for the involving state. In Fig. 5, a computing schedule of the proposed architecture is presented. Regarding as a partial parallel architecture, decoding a block of block RAM which store the messages of 8 $\mathbf{A}_{i,j}$ blocks is divided into 4 sections to access the message and computing CNP and VNP. Due to the wide word length of SRAM configuration, 2 message bundle are accessed from individual address corresponding to 2 $\mathbf{A}_{i,j}$ blocks. The CNP and VNP are sequentially processed as we are able to insert some filp flops to retiming the shortest path along the combinational circuit. As a result, we provide the synthesis result for the proposed LDPC decoder architecture with 636.74K gate counts using a TSMC 0.18um technology. The latency of decoding an QC-LDPC code is 6144 clock

cycle. With a clock frequency 320MHz, the throughput of decoder can reach 633Mbps.

IV. Conclusion

The proposed QC-LDPC decoder alleviate the complexity of random accessing messages passing scenario which affects the floor planning and complicated messages management to store back into block RAM. We utilize a simple block interleaver to reorder the messages sequence and circular shifter can simply solve the routing congestion during the place and route process. For the future research, a higher parallel processing unit can increase the throughput without the cost of routing difficulty. This generic architecture provide a insight of long QC-LDPC code for the application of storage system.

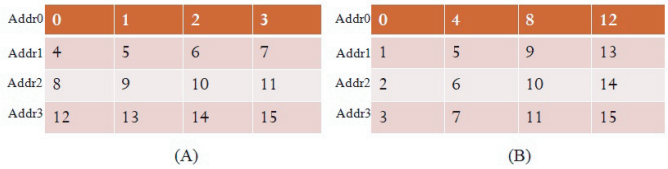

Fig. 1. Example of a packing memory.

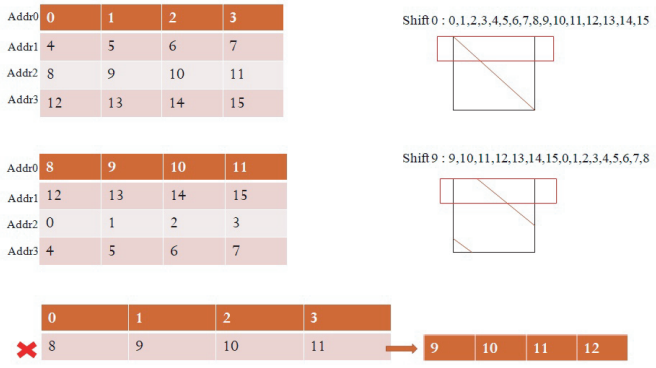

Fig. 2. Example of a natural packing approach.

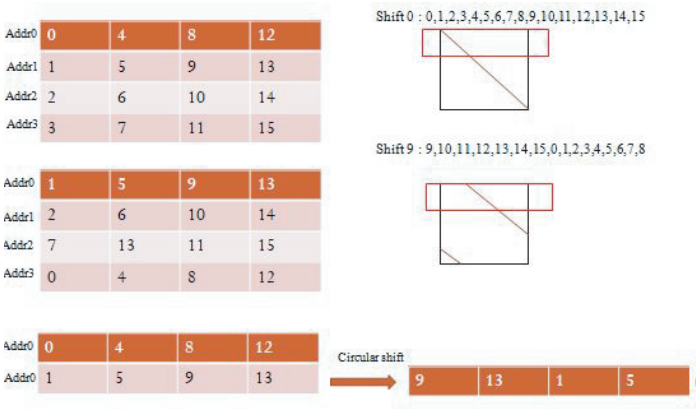

Fig. 3. Example of a proposed packing technique.

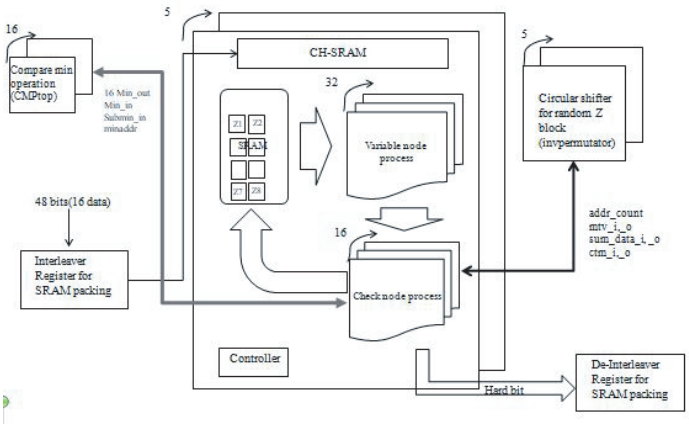

Fig. 4. A proposed LDPC decoder architecture.

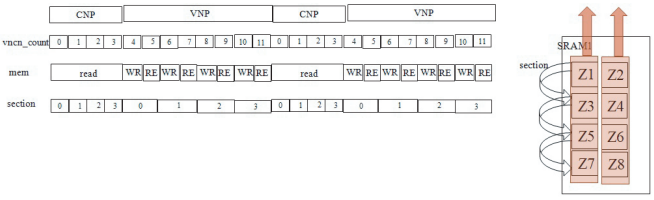

Fig. 5. A computing schedule of the proposed architecture

References

[1] Yoshio Nishi, Advances in Non-volatile Memory and Storage Technology, Electronic and Optical Materials: Woodhead Publishing, 2014.

[2] Yu Liu, Lixin Cheng and Xuguang Wang,"Commands scheduling optimized flash controller for high bandwidth SSD application," Solid-State and Integrated Circuit Technology (ICSICT) IEEE 11th International Conference on, pp.1-3, 2012

[3] L. Rota, M. Caselle, S. Chilingaryan, A. Kopmann, M. Webe, "A PCIe DMA Architecture for Multi-Gigabyte Per Second Data Transmission," Nuclear Science IEEE Transactions on, vol. 62, pp. 972-976, June 2015.

[4] Daesung Kim and Jeongseok Ha, "Serial quasi-primitive BC-BCH codes for NAND flash memories," (ICC) 2016 IEEE International Conference on, pp. 1-6.

[5] L. Lan, L. Zeng, Y. Tai, L. Chen, S. Lin, and K. Abdel-Ghaffar, " Construction of quasi-cyclic LDPC codes for AWGN and binary erasure channels: A finite field approach,"IEEE Transactions on Information Theory, vol. 53, no. 7, Jul. 2007, pp. 2429V2458.

[6] X. H. Chen, J. Y. Kang, S. Lin V. Akella, "Memory System Optimization for FPGA-Based Implementation of Quasi-Cyclic LDPC Codes Decoder,"IEEE Transactions on Circuit and System, vol. 58, Issue: 1, Jan. 2011, pp. 98-111.

[7] A.D. Houghton, The Enigneer's Error Coding Handbook. Springer, Boston, MA, 1997.