High-Rate Quasi-Cyclic LDPC Codes for Magnetic Recording Channel with Low Error Floor

Hao Zhong*, Tong Zhang*, and Erich F. Haratsch†
*ECSE Department, Rensselaer Polytechnic Institute, USA
† Agere Systems, Allentown, PA, USA

***Abstract*— By implementing an FPGA-based simulator, we investigate the performance of high-rate quasi-cyclic (QC) LDPC codes for the magnetic recording channel at very low sector error rates. Results show that error-floor-free performance can be realized by randomly constructed high-rate regular QC-LDPC codes with column weight 4 for sector error rates as low as 10^{-9}. We also conjecture several rules for designing randomly constructed high-rate regular QC-LDPC codes with low error floor. We also present a decoder architecture that is well suited to achieving high decoding throughput for these high-rate QC-LDPC codes with low error floor.**

I. Introduction

Recently, there has been a great interest in replacing Reed-Solomon codes with low-density parity-check (LDPC) codes in the magnetic recording channel [1]. Hard disk drive storage systems require powerful error correction codes that achieve sector error rates[1] of about 10^{-12} (and better) with high code rate. However, due to the lack of accurate analytical methods, it remains a challenge to accurately predict the error-correcting performance of LDPC codes at very low sector error rate. In the past, LDPC codes have been evaluated for the magnetic recording channel mainly based on computer simulations, with which sector error rates of only about 10^{-4} can currently be reached. In this regard, a high-speed dedicated hardware simulator is necessary to empirically investigate the performance of LDPC codes at very low sector error rates. Only recently, hardware simulators based on FPGA (field programmable gate array) chips [2], [3] have been implemented to investigate the performance of LDPC codes over the AWGN (additive white Gaussian noise) channel.

To be a promising candidate for the magnetic recording channel, LDPC codes must not only achieve very low sector error rate with a high code rate, but also be suitable for high-speed VLSI implementation to meet the high data rate requirements of hard disk drives. Prior work [4]–[8] has demonstrated that quasi-cyclic (QC) LDPC codes are one family of such implementation-oriented LDPC codes. The parity check matrix of a QC-LDPC consists of arrays of circulants. A circulant is a square matrix in which each row is the cyclic shift of the row above it, and the first row is the cyclic shift of the last row. However, discussion on how the structural parameters of the QC-LDPC code may affect the performance at very low error rate is largely missing in the open literature.

The contribution of this paper is three-fold: (1) By implementing iterative detection and decoding on an FPGA simulator for the magnetic recording channel, we demonstrate that randomly constructed high-rate regular QC-LDPC codes with the column weight 4 can be free of error floors at sector error rates of about 10^{-9}. This provides empirical evidence that LDPC codes deserve serious consideration for hard disk drives. (2) Based on extensive simulations using our FPGA platform, we observe that the size and weight of the circulants may largely affect the performance of randomly constructed high-rate regular QC-LDPC codes. We therefore speculate empirical rules for designing randomly constructed high-rate regular QC-LDPC codes with low error floor. (3) We improve our previously developed QC-LDPC decoder VLSI architecture [7] thereby supporting more flexible trade-offs between decoding throughput and silicon area. This new architecture allows to implement high-rate QC-LDPC codes with low error floor for very high decoding throughput.

II. Read Channel FPGA Simulator

Fig. 1 shows the diagram of the simulator that consists of two Altera Stratix-II 180 FPGA devices. The first FPGA

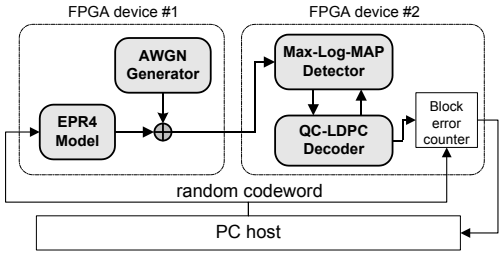

Fig. 1. System Diagram.

device models the magnetic recording channel as extended partial response class 4 (EPR4) signal in the presence of AWGN. The AWGN generator is designed based on the quantized version of the Box-Muller method [9]. It generates a random sample x with Gaussian distribution (zero mean and standard deviation $\sigma = 1$) using two random samples x_1 and x_2 uniformly distributed between [0, 1] as follows:

$$x = f(x_1) \cdot g(x_2), \quad \text{where}$$
$$f(x_1) = \sqrt{-ln(x_1)} \quad \text{and} \quad g(x_2) = \sqrt{2}\cos(2\pi x_2).$$

[1]Notice that we use the term of *sector error rate* following the convention in the magnetic storage community. Although in current hard disk drives one sector equals 512 user bytes, throughout the paper, we always assume that the length of one sector equals to the length of the considered LDPC code.

An array of 64-bit linear feedback shift registers is used to generate the random samples x_1 and x_2. The functions $f(x_1)$ and $g(x_2)$ are implemented using look-up tables. The calculated sample x is scaled according to the signal-to-noise ratio (SNR) at the output of the AWGN generator.

The second FPGA device implements an iterative datapath consisting of a Max-Log-MAP detector and a QC-LDPC decoder. The well-known sliding window method [10] is used to implement the Max-Log-MAP detector. The QC-LDPC decoder is implemented using the architecture presented in [7]. The detector and decoder operate on the received data iteratively. As shown in Fig. 1, the PC host provides randomly generated codewords for simulation. Due to the speed mismatch between the FPGA simulation and the data transfer between the FPGA and PC host, each random codeword is used for several rounds of FPGA simulations.

The finite precision parameters of the simulator are outlined as follows: (a) The output of the AWGN generator is 6 bits with 3 fractional bits; (b) In the Max-Log-MAP detector, the branch and path metrics are 9 bits with 3 fractional bits, and the detector soft-output is 6 bits with 3 fractional bits; (c) In the QC-LDPC decoder, the internal decoding messages and soft outputs are 6 bits with 3 fractional bits. For the simulation results presented below, we configured the simulator as follows: (i) The maximum number of detector-decoder iterations is 4, i.e., the decoder sends its soft output back to the detector at most 4 times, and (ii) each time, the QC-LDPC decoder carries out 4 internal decoding iterations.

III. Simulation Results and Discussion

The parity check matrix $\mathbf{H}$ of a QC-LDPC code can be written as

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{1,1} & \mathbf{H}_{1,2} & \cdots & \mathbf{H}_{1,n} \\ \mathbf{H}_{2,1} & \mathbf{H}_{2,2} & \cdots & \mathbf{H}_{2,n} \\ \cdots & \cdots & \ddots & \cdots \\ \mathbf{H}_{m,1} & \mathbf{H}_{m,2} & \cdots & \mathbf{H}_{m,n} \end{bmatrix},$$

where each sub-matrix $\mathbf{H}_{i,j}$ is a $p \times p$ circulant over GF(2). Notice that the permutation matrix and zero matrix are special cases of circulants with the weights 1 and 0, respectively. In this work, we focus on randomly constructed high-rate regular QC-LDPC codes, where all the non-zero circulants have the same weight. Given the structural parameters, including m, n, p, column weight, and circulant weight, we construct the parity check matrix randomly subject to the constraint that there are no any cycles of degree 4 or less. In all the parity check matrices we have ever constructed, at most there are only one or two redundant rows. Hence the code rate can be approximated as $(n-m)/n$. Leveraging the FPGA simulator, we investigated how the code parameters affect the performance of high-rate QC-LDPC codes, as discussed below. All the simulated sector error rates presented in the following were obtained under the condition that at least 10 erroneous sectors are captured.

First, we present how the weight of non-zero circulants, denoted as w, may affect the performance. We simulated two QC-LDPC codes with the same column weight 6 and different values for w, i.e., 2 and 3. Fig. 2 shows the simulated results. Although the code with $w = 3$ has a longer code length (N=9216), its performance curve is slightly worse in the water-fall region and shows an error floor at the sector error rate of 10^{-8}. We conjecture that $w \geq 3$ is not a good choice and may render the randomly constructed regular QC-LDPC code more subject to error floors.

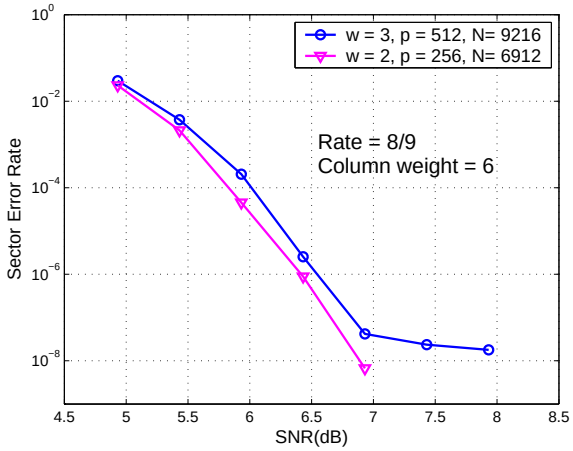

Fig. 2. Simulation results that show the effect of circulant weight on the performance.

Next, we show the effect of the circulant size (i.e., the value of p) on the performance. We simulated three QC-LDPC codes with the same code length of 4068, code rate of 8/9, and column weight of 4. Three different values for p were considered, including 64, 128, and 256. The QC-LDPC code with $p = 64$ has the worst performance and badly suffers from an error floor, while the other two do not show an error floor in the region that can be observed using the FPGA simulator. This may suggest that the value of p should be relatively large in order to achieve a low error floor for randomly constructed high-rate QC-LDPC codes.

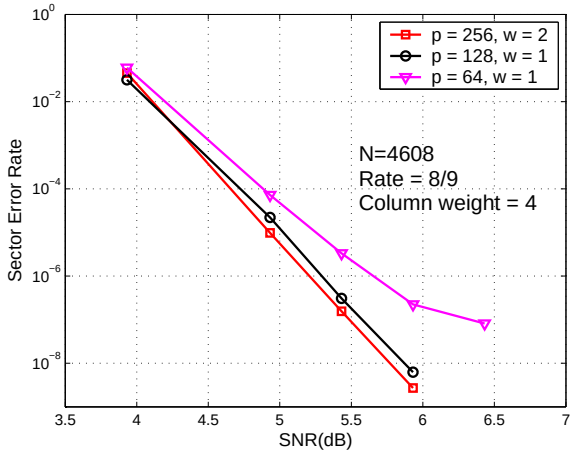

Fig. 3. Simulation results that show the effect of circulant size on the performance.

From the above, we conjecture the following rules for designing randomly constructed high-rate QC-LDPC codes with low error floors for the magnetic recording channel: (i) keep the circulant weight less than 3, and (ii) make the circulant size relatively large. Moreover, our simulation results suggest that a column weight of 4 may be the best choice for high-rate QC-LDPC codes, as very good performance can be achieved with relatively low computational complexity. Finally, three high-rate QC-LDPC codes with circulant weight 2 and column weight 4 are compared in Fig. 4, where the highest code rate is 15/16 (i.e., 0.9375). Although the performance curve slopes vary for the different code lengths and code rates, none of the simulated codes shows an error floor.

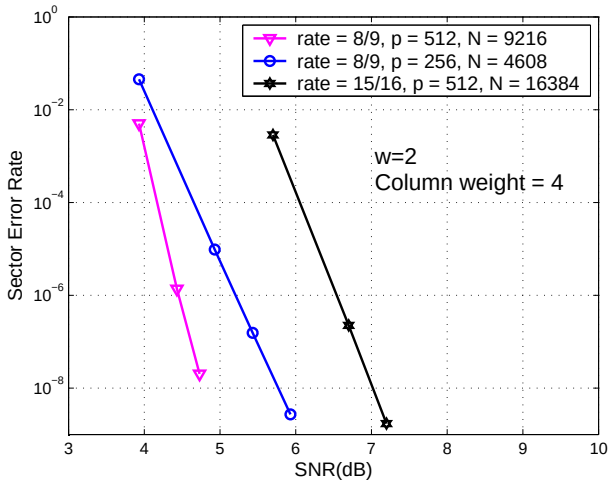

Fig. 4. Comparison of three randomly constructed high-rate regular QC-LDPC codes.

IV. QC-LDPC Decoder Design for High Throughput

As discussed above, in order to achieve low error floors, the circulant size (i.e., the value for p) of randomly constructed high-rate QC-LDPC codes should be relatively large. The QC-LDPC decoder architectures presented in [6], [7] are suitable for high-speed decoding due to their simple datapath and fixed interconnect structure. However, the decoding parallelism in these decoders is reversely proportional to p, i.e., the computations of each group of p variable or check nodes are mapped onto a single hardware processing unit in a time-division multiplexed mode. Large values for p will directly reduce the achievable throughput of such decoders.

To solve this problem, we propose an improved decoder architecture that can map the computations for each group of v (where p is divisible by v and h is defined as the ratio p/v) variable or check nodes onto a single hardware processing unit, leading to an h times improvement of the decoding parallelism. Fig. 5(a) shows the decoder architecture for a QC-LDPC code with a $(m \cdot p) \times (n \cdot p)$ parity check matrix. It contains m groups of check node computation units (CNUs) and n groups of variable node computation units (VNUs), where each group contains h CNUs or VNUs. Each CNU (VUN) performs the computations associated with consecutive v rows (columns) in the parity check matrix in a time-division multiplexed mode. All the decoding messages and channel messages are stored in a memory fabric, as shown in Fig. 5(a). Each decoding iteration takes $2v$ clock cycles:

1) During the first v clock cycles, the decoder works in check node processing mode, i.e., carrying out the computations associated with all the $m \cdot p$ check nodes;
2) During the second v clock cycles, the decoder works in variable node processing mode, i.e., carrying out the computations associated with all the $n \cdot p$ variable nodes.

The real *challenge* in the decoder design is how to design the memory fabric and interconnect between the memory fabric and CNU/VNU array in such a way that *all* the messages required for the *same* variable or check node computation are sent to the *same* VNU or CNU at the *same* clock cycle. In the following, we present our solution to tackle this issue.

The memory fabric mainly contains arrays of decoding message memory blocks (DMMBs) and channel message memory blocks (CMMBs). Recall that w represents the weight of non-zero circulants. All the $w \cdot p$ decoding messages associated with one non-zero circulant are stored in w DMMBs. Notice that each non-zero circulant can be considered as a sum of w permutation matrices. Each DMMB stores the p decoding messages associated with the p 1's in each permutation matrix. The address space of each DMMB is $0 \sim v-1$, and each address location stores h decoding messages. Let $x_0, x_1, \cdots, x_{p-1}$ denote the p decoding messages sorted in ascendent order by the column index of the corresponding 1's in the permutation matrix. At the i-th address location, DMMB stores the h decoding messages $\{x_i, x_{i+v}, \cdots, x_{i+(h-1)v}\}$. For non-zero circulant $\mathbf{H}_{i,j}$, the corresponding DMMB group has the architecture as shown in Fig. 5(b), which is explained below.

Each DMMB is a dual-port memory and has one port always configured for read and another one always configured for write. The read address of each DMMB is generated by a binary counter. Let $t_1, \cdots, t_w$ represent the column indices of the w non-zero entries in the first row of the circulant $\mathbf{H}_{i,j}$. In each decoding iteration, the state of the binary counter associated with DMMB$_k$ $(1 < k < w)$ is initialized as $t_k \ mod \ v$ at the beginning of the check node processing mode and initialized as 0 at the beginning of the variable node processing mode. The write address is simply a delayed version of the read address depending on how many pipeline stages are inserted in the datapath between the DMMB memory data output and input ports. The barrel shifter is a combinational circuit that can rotate the input by any number of bits in a single operation. The barrel shifters associated with DMMB$_k$ are configured to rotate $\lfloor t_k/v \rfloor$ decoding messages. As illustrated in Fig. 5(b), the DMMB group associated with circulant $\mathbf{H}_{i,j}$ connects with the i-th group of CNUs and j-th group of VNUs.

The memory fabric contains n CMMB blocks, each of which stores the channel messages for each group of p consecutive variable nodes. Each CMMB is a single-port

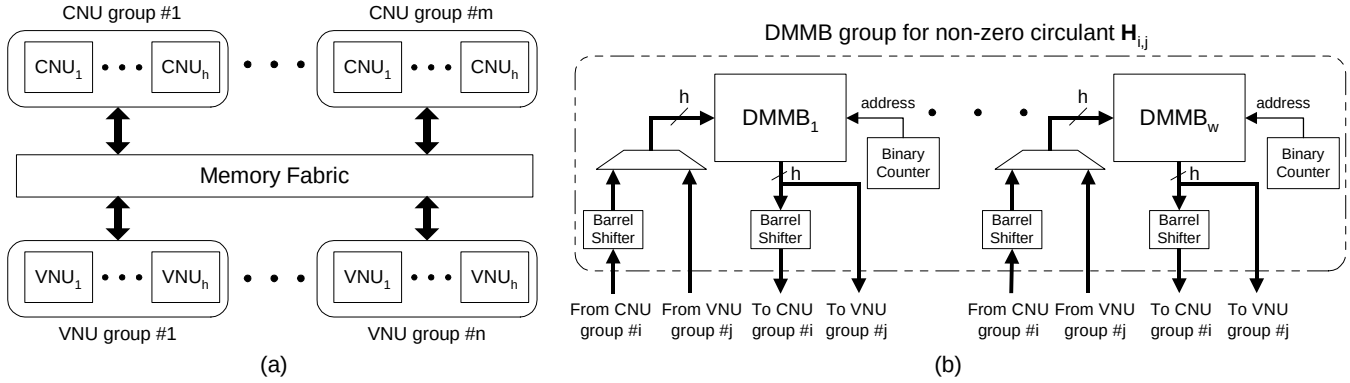

Fig. 5. (a) The general decoder architecture, and (b) storage of the decoding messages associated with non-zero circulant $\mathbf{H}_{i,j}$.

memory with the address space of $0 \sim v-1$. Each memory location stores h channel messages. The storage pattern of the p channel messages in one CMMB is the same as the pattern in DMMB, i.e., let $c_0, c_1, \cdots, c_{p-1}$ denote the p channel messages sorted in ascendent order by the column index, each CMMB stores h channel messages $\{c_i, c_{i+v}, \cdots, c_{i+(h-1)v}\}$ at the i-th address location. Since the channel messages are only used in variable node processing mode, the CMMBs only send the data to the corresponding VNU groups. The read address of each CMMB is generated by a binary counter that is initialized as 0 at the beginning of the variable node processing mode.

V. Conclusion

Using an FPGA-based simulator, we empirically evaluated high-rate QC-LDPC codes for the magnetic recoding channel at low sector error rates. We demonstrated that randomly constructed high-rate regular QC-LDPC codes with column weight 4 can achieve error-floor-free performance for sector error rates as low as 10^{-9}. Based on extensive FPGA simulations, we speculated empirical rules for designing randomly constructed high-rate QC-LDPC codes with low error floors. Moreover, by improving published decoder architectures, we presented a new decoder architecture that is better suited to achieving high-speed decoding for high rate QC-LDPC codes with low error floors. Future work is directed to evaluating the performance of those deterministically constructed QC-LDPC codes and comparing them with their randomly constructed counterparts for the magnetic recording channel.

References

[1] A. Dholakia, E. Eleftheriou, T. Mittelholzer, and M.P.C. Fossorier, "Capacity-approaching codes: can they be applied to the magnetic recording channel?," *IEEE Communications Magazine*, vol. 42, pp. 122–130, Feb. 2004.

[2] D. Divsalar, S. Dolinar, J. Thorpe, and C. Jones, "Constructing ldpc codes from simple loop-free encoding modules," in *Proc. of IEEE International Conference on Communications*, May 2005, pp. 658–662.

[3] L. Sun, H. Song, and B.V.K.V. Kumar, "Error floor investigation and girth optimization for certain types of low-density parity check codes," in *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, March 2005, pp. 1101–1104.

[4] M. M. Mansour and N. R. Shanbhag, "Architecture-aware low-density parity-check codes," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, Bangkok, Thailand, May 2003, pp. 57–60.

[5] D. E. Hocevar, "LDPC code construction with flexible hardware implementation," in *IEEE International Conference on Communications*, 2003, pp. 2708 –2712.

[6] J.K.-S. Lee, B. Lee, J. Thorpe, K. Andrews, S. Dolinar, and J. Hamkins, "A scalable architecture of a structured LDPC decoder," in *International Symposium on Information Theory*, July 2004, p. 293.

[7] H. Zhong and T. Zhang, "Block-LDPC: A practical LDPC coding system design approach," *IEEE Transactions on Circuits and Systems I*, vol. 52, pp. 766–775, April 2005.

[8] Z. Li, L. Chen, S. Lin, W. Fong, and P.-S. Yeh, "Efficient encoding of quasi-cyclic low-density parity-check codes," *IEEE Transactions on Communications*, to appear, 2005.

[9] E. W. Weisstein, *Box-Muller Transformation*, From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/Box-MullerTransformation.html.

[10] A. J. Viterbi, "An intuitive justification and a simplified implementation of the map decoder for convolutional codes," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 260–264, Feb. 1998.