electronics

MDPI

Article

Iterative Decoding of LDPC-Based Product Codes and FPGA-Based Performance Evaluation

Weigang Chen *, Wenting Zhao, Hui Li, Suolei Dai, Changcai Han and Jinsheng Yang

School of Microelectronics, Tianjin University, Tianjin 300072, China; zhaowt@tju.edu.cn (W.Z.); hui_li@tju.edu.cn (H.L.); daisuolei@tju.edu.cn (S.D.); cchan@tju.edu.cn (C.H.); jsyang@tju.edu.cn (J.Y.)

* Correspondence: chenwg@tju.edu.cn; Tel.: +86-186-2264-0107

Received: 31 October 2019; Accepted: 5 January 2020; Published: 8 January 2020

Abstract: Low-density parity-check (LDPC) codes have the potential for applications in future high throughput optical communications due to their significant error correction capability and the parallel decoding. However, they are not able to satisfy the very low bit error rate (BER) requirement due to the error floor phenomenon. In this paper, we propose a low-complexity iterative decoding scheme for product codes consisting of very high rate outer codes and LDPC codes. The outer codes aim at eliminating the residual error floor of LDPC codes with quite low implementation costs. Furthermore, considering the long simulation time of computer simulation for evaluating very low BER, the hardware platform is built to accelerate the evaluation of the proposed iterative decoding methods. Simultaneously, the fixed-point effects of the decoding algorithms are also be evaluated. The experimental results show that the iterative decoding of the product codes can achieve a quite low bit error rate. The evaluation using field programmable gate array (FPGA) also proves that product codes with LDPC codes and high-rate algebraic codes can achieve a good trade-off between complexity and throughput.

Keywords: low-density parity-check codes; error floor; product codes; field programmable gate array

1. Introduction

Due to the rapid increase of data volume, the next-generation very high-throughput optical communication has attracted wide interest [1–5]. It also requires the system to have a quite low bit error rate (BER) to ensure the high reliability of communication, and thus advanced channel codes must be used [6–8]. Similarly, the very low BER is also required in the next-generation high-density storage systems [9]. The low-density parity-check (LDPC) code is well known to approach Shannon limit when the code length becomes very large [10–12]. It has the potential to meet the requirement of the high throughput for very high speed communication systems using parallel iterative decoding. However, due to the suboptimal property of the iterative decoding algorithm and the intrinsic error-prone substructures in the LDPC structures, the error floor phenomenon exists when LDPC codes operate in the high signal-to-noise ratio (SNR) region [13–15]. This phenomenon limits the application of LDPC codes for these high-throughput optical communications [13].

In order to satisfy the very low BER requirements, the approaches for reducing the error floor can be divided into two categories. One is to optimize the structure of LDPC codes or to improve the LDPC decoding algorithm, but the additional complexity brought by these two approaches is higher [16–18]. The other is to construct concatenated codes or product codes [15,19–21]. Constructing concatenated codes requires that the outer code has great error correction capability due to the presence of abundant residual error bits in LDPC codewords, which causes the code rate loss of concatenated codes and further affects the overall performance. The product code has the potential to match the error characteristic of LDPC codes for high throughput and good error correction performance [20].

According to the analysis of the residual error characteristic of LDPC codes, product codes which use the LDPC code as the inner code and use the algebraic code whose code rate is close to 1 as the outer code are designed to achieve extremely low BER in [21]. Therefore, the overall code rate of the product code is very close to the code rate of the inner code.

The performance of the product code with various component codes can be further improved using the iterative decoding algorithm [22,23]. In this paper, an iterative decoding scheme between the inner and outer codes is proposed to further improve the performance of the product code proposed in [21]. The scheme updates the prior probability information of LDPC codes in the following iteration process according to the decoding results of the outer code, reducing the residual errors after LDPC decoding, and further makes the outer code correct the residual errors better. We demonstrate that the performance of the proposed iterative decoding scheme obviously surpasses the performance of decoding scheme without iteration. Furthermore, considering software simulation is limited by time and equipment due to the requirements of longer code length and lower error floor, and the quantization method is an important factor for the performance of iterative decoding [19,24], the corresponding hardware decoder and the overall evaluation platform of the product code are implemented to further verify the proposed iterative decoding methods. Hardware experiment results show that the designed codec structure achieves a good trade-off between hardware complexity and throughput and thus effectively accelerates the performance evaluation. The implemented iterative codec verifies that the product codes under finite precision implementation can achieve significant performance gain and presents the good trade-off of complexity and performance.

The rest of this paper is organized as follows. Section 2 presents a type of product codes and the analysis of the residual error characteristics of LDPC codes. Section 3 details the iterative decoding scheme of LDPC-based product codes and presents the performance verification. In Section 4, the performance of the proposed iterative decoding method is analyzed via experimental results using the hardware evaluation platform. Section 5 concludes the present work.

2. Residual Error Characteristics of Multiple Consecutive LDPC Codes

In this section, the construction method of product codes using very high rate outer codes in [23] is reviewed. Furthermore, the residual error characteristics of LDPC codes are illustrated to show the feasibility of constructing product codes.

2.1. The Product Codes with LDPC Codes

In this paper, we consider a type of product codes using very high rate block code $C_1(n_1, k_1)$ and LDPC code $C_2(n_2, k_2)$ [23]. Different from traditional product codes, which usually use short codes as component codes, the inner and outer codes of this type of product codes are designed with longer code lengths, and the rate of the outer code is very high with limited error correction capability. This type of product codes can be decoded with low complexity.

The structure of the product code is illustrated in Figure 1. First, the information bits are arranged into k_1 rows and k_2 columns. Second, the k_1 bits of each column are encoded into an outer codeword, and R_2 outer codewords form the matrix of n_1 rows and k_2 columns. Third, k_2 bits in each row are encoded into an LDPC codeword, and all the n_1 LDPC codewords form the final data block with the dimension of $n_1 \times n_2$. Therefore, the code rate of the product code is $R = R_1 R_2 = (k_1 k_2)/(n_1 n_2)$, where R_1 is the code rate of the outer code, namely $R_1 = k_1/n_1$ close to 1, and R_2 is the code rate of the LDPC code, namely $R_2 = k_2/n_2$. In this paper, we choose Hamming codes, which can only correct a single error, in order to achieve the highest code rate for outer codes. Therefore, the overall code rate R of the product code is very close to the code rate R_2 of the LDPC code.

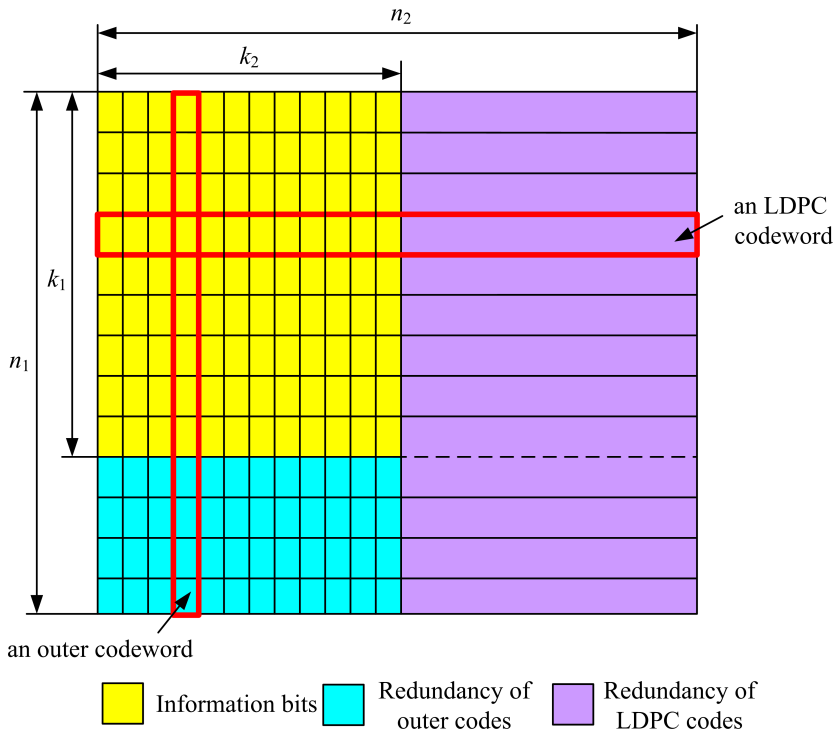

Figure 1. The product code structure using the LDPC code and high rate algebraic code.

2.2. Analysis of Residual Error Characteristics for Product Codes

In order to show the feasibility of the proposed product codes, the residual error characteristics of LDPC codes are analyzed. A case study of the product code using the LDPC code under IEEE 802.16e as the inner code is considered. The simulation is performed under an additive white Gaussian noise (AWGN) channel with binary phase shift keying (BPSK) constellation.

First, at different SNRs, the residual error distribution of the LDPC code with the code rate of 0.5 and code length of 576 bits is illustrated in Figure 2, where k denotes the number of residual error bits occurring in the information bit part in the LDPC codeword after decoding, and $P(k)$ denotes the occurrence probability of the codeword in which k information bits are erroneous. It can be observed that, though the number of residual error bits of the LDPC code is much less than the code length, it is still large compared with some special LDPC codes [19,20]. Therefore, the product code scheme is proposed to disperse the residual errors of LDPC codewords into various algebraic codewords. It is a general scheme for constructing codes to achieve very low BER.

Then, the probability distribution of the number of erroneous information bits, which are at the same entry in multiple successive different LDPC codewords, is simulated. Figure 3 presents an example, in which there are two erroneous LDPC codewords and they cause many outer codewords to have one or two error bits. The distribution simulation result is shown in Figure 4, where n denotes the number of erroneous information bits at the same position across consecutive N codewords, $P(n)$ denotes the probability of the number of error bits equal to n, and N denotes the number of consecutive codewords for analysis in a product construction form. It can be observed that n is quite small, and the corresponding probability $P(n)$ decreases very sharply with the increasing of n. As E_b/N_0 increases, the tail of the distribution vanishes quickly. It is quite different with the distribution of the residual errors in a single codeword. Thus, high rate algebraic codes can be used to construct product codes for correcting residual errors and compensating for rate loss.

From the analysis above, the residual errors of LDPC codes in product codes are dispersed into multiple different outer codewords, at most k_2 different outer codewords. Most outer codewords do not contain residual error bits or only contain one or two error bits due to two facts. One fact is the low

LDPC codeword error rate in the reasonable SNR region. When the codeword error rate is low, the case that two or more error LDPC codewords occur in one product codeword is quite rare. The other is the sparsity of the error bits compared with the length of the whole information bits, as shown in Figure 2. Therefore, we adopt the very high rate algebraic code with limited error correction capability as the outer code, aiming at reducing the additional complexity, for the algebraic codes can be decoded using very concise structure, for example, the error-trapping decoding.

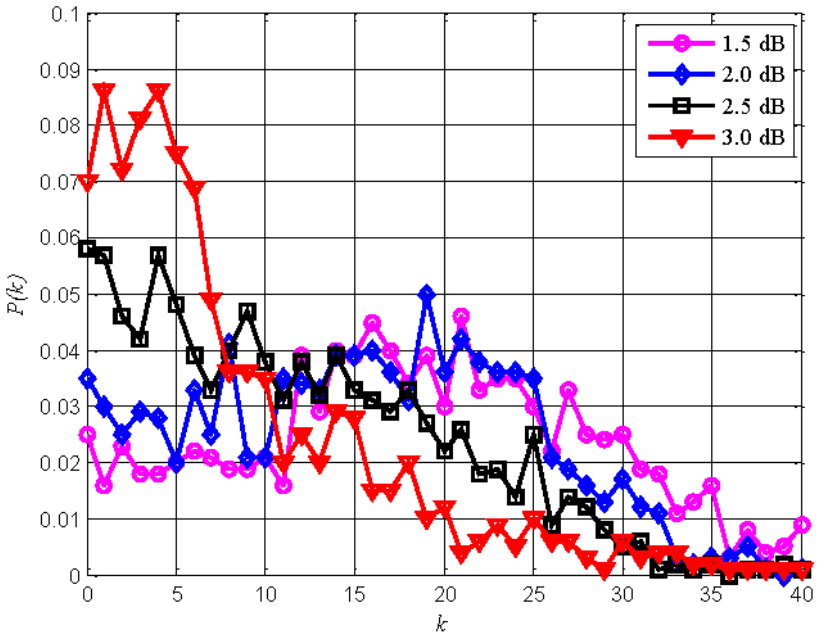

Figure 2. The residual error distribution of LDPC codewords [23].

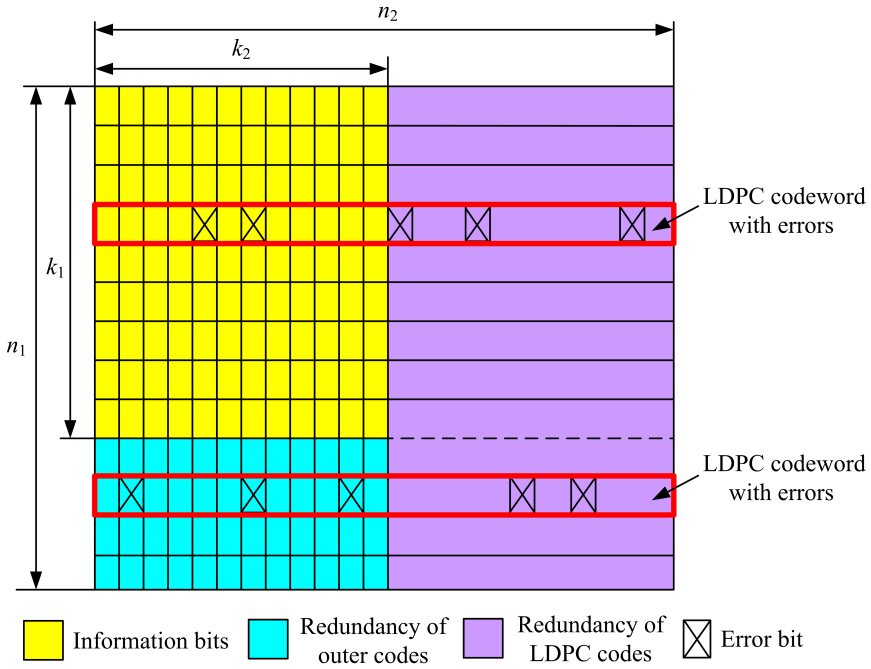

Figure 3. Errors in the LDPC codewords dispersed into multiple outer codewords.

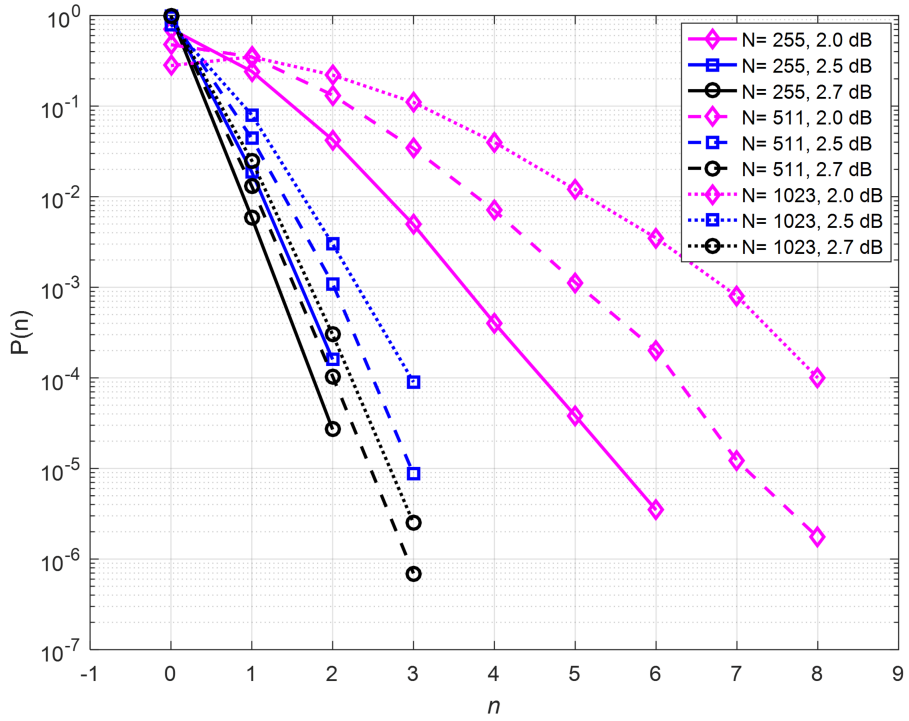

Figure 4. The probability distribution of the number of erroneous information bits.

3. Proposed Low-Complexity Iterative Decoding for LDPC-Based Product Codes

In this section, the proposed iterative decoding scheme is presented. Meanwhile, the performance of the LDPC-based product code with iterative decoding is verified through software simulation.

3.1. Iterative Decoding of Product Codes

The product code using high rate block code and LDPC code designed in [23] achieves superior error correction performance in the high SNR region aiming at eliminating the error floors. Although most outer codewords with error bits can be corrected, it can be seen from Figure 4 that there may still be a large number of residual error bits in some outer codewords. For these codewords, the outer codes may fail to correct the residual error bits, thus affecting the overall BER performance of product codes.

In this paper, a low complexity iterative decoding scheme is designed for this type of product codes. The overall flow diagram of the iterative decoding scheme is shown in Figure 5 and the detailed steps are presented by Algorithm 1. The scheme introduces iterations between the LDPC decoding and the outer decoding via exchanging hard-decision results. The outer code can use very simple decoding algorithms based on hard-decision results, for example, error-trapping decoding. Error-trapping decoding is a classical decoding method used to decode cyclic codes and this decoding scheme only needs cyclic shift operation. Therefore, its hardware implementation is simple, thus having low complexity [25]. The hard-decision output results from the outer decoder are fed back to the LDPC decoder which retries decoding on the failed codewords in the earlier rounds. The iterations of decoding for the inner and outer codes continue in this fashion until either the LDPC decoder generates valid codewords or the number of iterations reaches the preset number. The scheme can reduce the number of residual error bits after LDPC decoding, thereby reducing the number of erroneous outer codewords due to limited error correction capability of outer codes.

The increased computation complexity of the proposed decoding algorithms is quite small. The increased complexity can be decomposed into two parts. One is brought by the outer decoding algorithms. In this paper, we use the error-trapping decoding for Hamming codes and the complexity is small. It will be further verified in Section 4 using hardware resources. Actually, the trapping

decoding algorithms have the similar complexity as the encoder for Hamming codes. The other is brought by the increased additional LDPC decoding rounds. Fortunately, only very rare failed LDPC codewords need to execute the additional Belief–Propagation (BP) iterative decoding. The proportion of failed LDPC codewords in previous iteration is directly related to the codeword error rate, which is quite low for our several design examples.

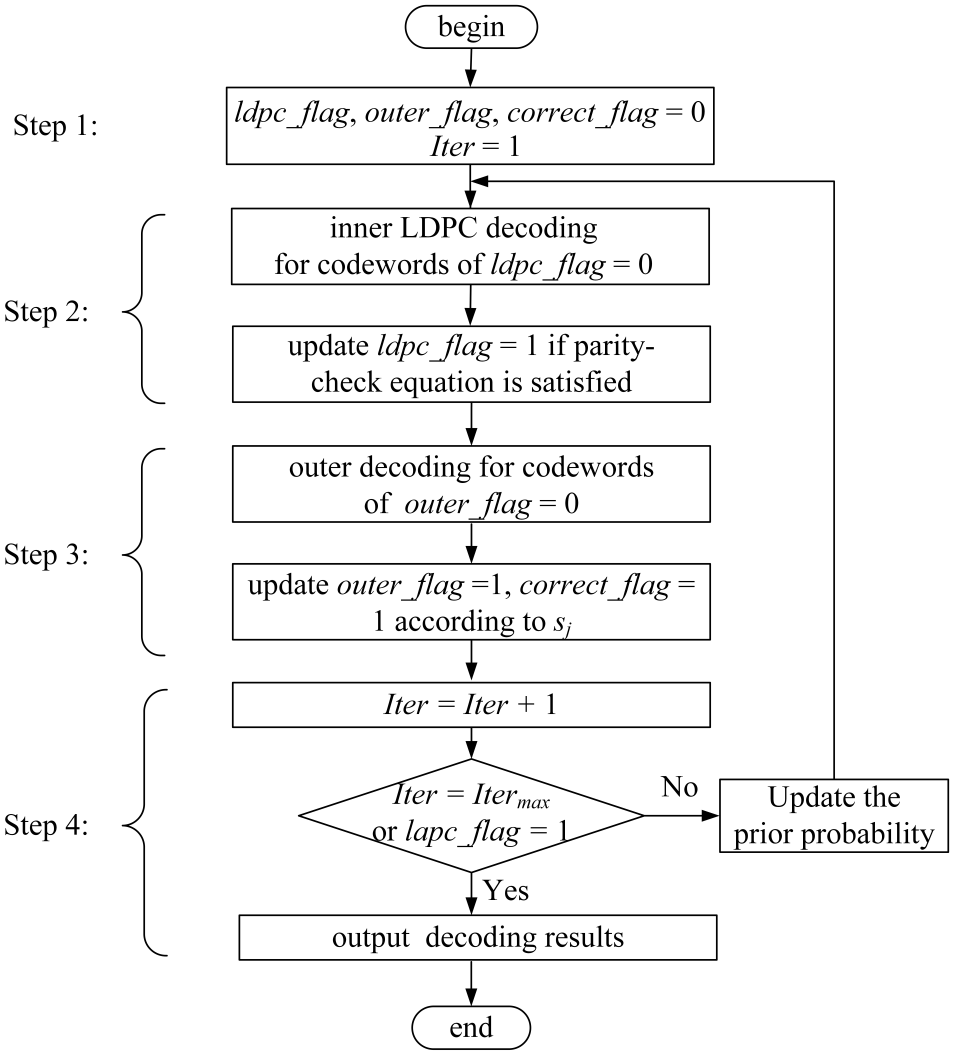

Figure 5. The overall flow diagram of the iterative decoding process of the product code.

In the discussion of the rest of the paper, we use the following notations and define the variables in our scheme:

- $ldpc\_flag$ of length n_1 denotes the decoding status of LDPC codewords.
- $outer\_flag$ of length k_2 denotes the decoding status of outer codewords.
- $correct\_flag$ whose dimension is $n_1 \times k_2$ denotes the correcting status of the bits.
- $Iter$ is the iteration number.
- $Iter_{max}$ is the maximum number of iterations.
- y_{ldpc} is the prior probability information of LDPC codewords.
- $\hat{y}_{ldpc}$ is the posterior probability information of LDPC codewords.
- c_i is the check result of LDPC codewords using the parity–check matrix.
- s_j is the check result of outer codewords using the syndrome.
- y_{outer} is the decoding result of outer codewords.

Algorithm 1: The iterative decoding for product codes

Step1: Parameter initialization.

1.1. Initialize three flag bits to false and the iteration number $Iter$.

$$\begin{cases} Iter = 1, \\ ldpc\_flag(i) = 0, & i = 1, 2, \cdots, n_1 \\ outer\_flag(j) = 0, & j = 1, 2, \cdots, k_2 \\ correct\_flag(i, j) = 0, \end{cases}$$

Step2: Inner LDPC decoding.

2.1. Before decoding, compute the prior probability information of LDPC codewords y_{ldpc}.

2.2. Decode the LDPC codewords with $ldpc\_flag = 0$ using a generalized decoding algorithm.

2.3. For $0 < i \leq n_1$, update the $ldpc\_flag(i)$ bit of the LDPC codewords with the c_i of 0 is set to true.

$$ldpc\_flag(i) = \begin{cases} 1, & \text{if } \ c_i = 0, \\ 0, & \text{if } \ c_i \neq 0, \end{cases}$$

2.4. Compute the posterior probability information $\hat{y}_{ldpc}$.

Step3: Outer decoding.

3.1. Decode the outer codewords with $outer\_flag = 0$ using the error-trapping algorithm;

3.2. For $0 < j \leq k_2$, update the $outer\_flag(j)$ bit of the outer codewords with the s_j of 0 is set to true.

$$outer\_flag(j) = \begin{cases} 1, & \text{if } \ s_j = 0, \\ 0, & \text{if } \ s_j \neq 0, \end{cases}$$

3.3. After decoding, if the decoder corrects the error bits in the outer codewords, $correct\_flag(i, j) = 1$.

3.4. Compute $Iter = Iter + 1$, if $Iter = Iter_{max}$ or $ldpc\_flag = \mathbf{1}$, output the decoding result, otherwise, go to execute Step4;

Step4: Update the prior probability information

4.1. For $0 < i \leq n_1, 0 < j \leq k_2$, if the $correct\_flag(i, j) = 1$ and $outer\_flag(j) = 1$, go to step 4.2;

4.2. Update the prior probability information according to the decoding result of outer codewords y_{outer}. Then, go to execute step 2;

$$\begin{aligned} y_{ldpc} &= sign(y_{outer}) \left| \hat{y}_{ldpc} \right| \\ &= \begin{cases} \hat{y}_{ldpc} & \text{if } \ y_{outer} = 0, \\ (-1) \times \hat{y}_{ldpc} & \text{if } \ y_{outer} = 1. \end{cases} \end{aligned}$$

3.2. Performance of Product Codes Using Iterative Decoding

This part presents the iterative decoding performance of product codes consisting of LDPC codes and Hamming codes with different code lengths. For the LDPC code, we choose the LDPC code with the code length of 576 bits and the code rate of 0.5 in IEEE 802.16e as an example. The BP decoding algorithm is used for LDPC codes and the maximum number of iterations is set to 30, for it is enough for the convergence of the decoding algorithm. The simulation is performed under the AWGN channel with BPSK constellation. It should be noted that, in order to verify the performance of the product codes, an ideal decoding algorithm is used for outer decoding. During the decoding process, the number of errors occurring among the outer codewords can be determined. If the number of errors occurring is within the error correction capability of the outer code, the decoder will correct all the errors; otherwise, the decoder will not perform any operation on this codeword. In this way, some

LDPC codewords, which fail in the previous iteration, may succeed in the new round of decoding due to the fact that some bits have been corrected by outer codes.

First, the simulation result of the product code with Hamming (255, 247) using the proposed iterative decoding is shown in Figure 6. It can be observed that the product code with two iterations has about 0.65 dB gain over the product code without iteration. In addition, the product code with five iterations has about 1.0 dB gain compared with the product code without iteration.

Then, the performance of product codes using Hamming (511, 501) and Hamming (1023, 1012) as outer codes is verified, as shown in Figures 7 and 8. The simulation results verify that the iterative decoding method is effective for outer codes with different code lengths. It can be observed that the overall performance can be improved significantly using only several iterations, for example, five iterations.

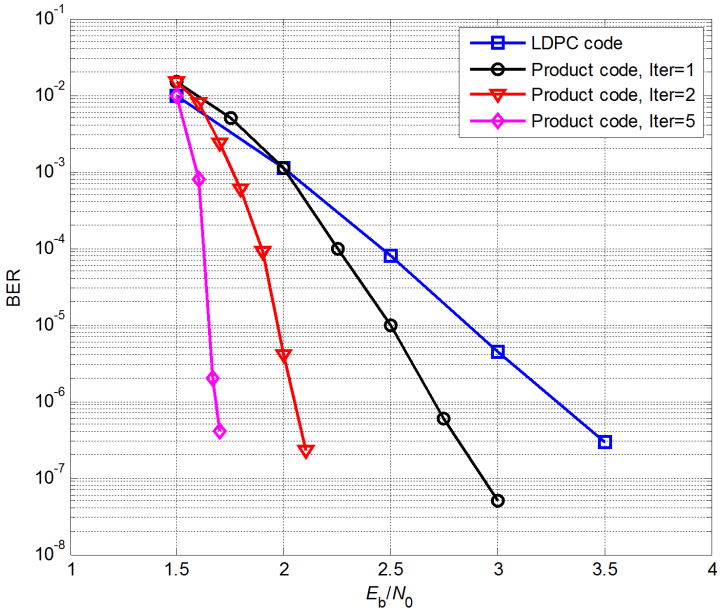

Figure 6. Iterative decoding performance of the product code with LDPC (576, 288) and Hamming (255, 247).

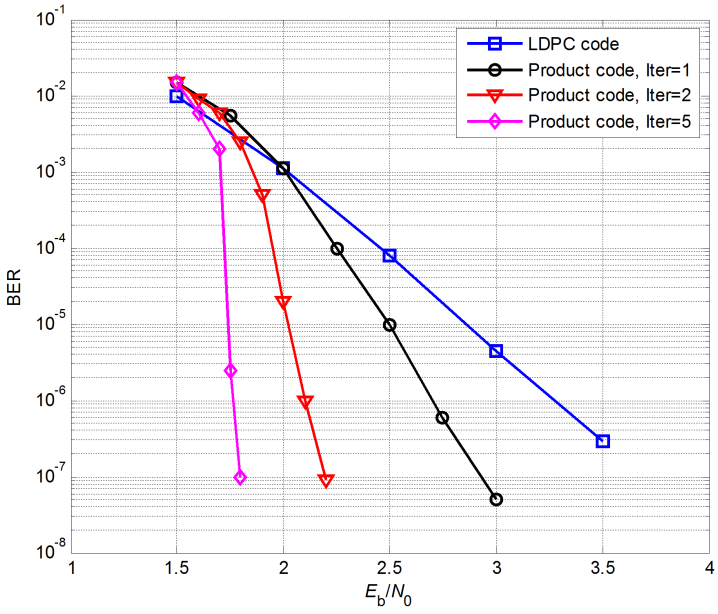

Figure 7. Iterative decoding performance of the product code with LDPC (576, 288) and Hamming (511, 502).

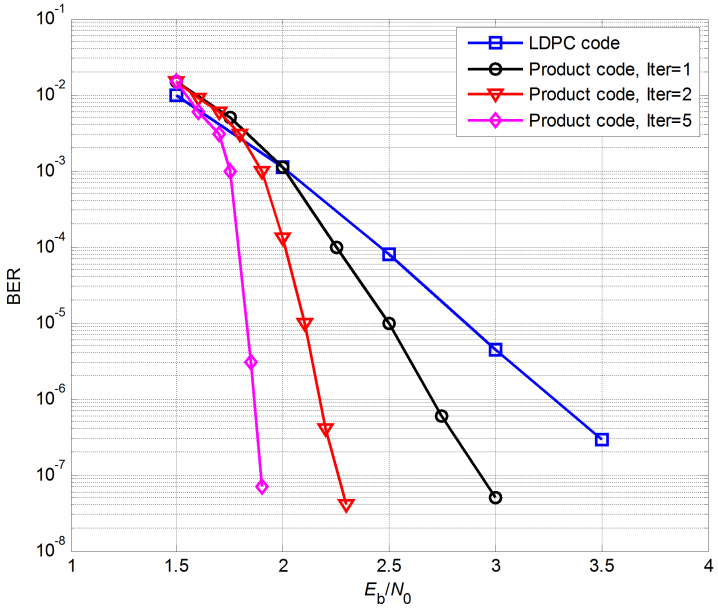

Figure 8. Iterative decoding performance of the product code with LDPC (576, 288) and Hamming (1023, 1013).

4. Hardware Evaluation for Iterative Decoding of Product Codes

The performance of the LDPC codes is affected by decoding algorithms and quantization methods [19,24]. Furthermore, in the case of extremely low error rate, software simulation is not practical for it will need a long period of simulation time [26]. Therefore, the hardware platform is built in this paper. On one hand, the hardware platform can accelerate the performance evaluation of the proposed iterative decoding methods. On the other hand, the quantization methods for specific decoding algorithms and the feasible implementation architecture are also evaluated [5,13,14]. In this section, the encoder and decoder of the product code are implemented, and the corresponding performance evaluation results using the hardware platform are presented.

4.1. Hardware Evaluation Platform

In this subsection, the structure of the hardware evaluation platform is illustrated in Figure 9. All the source bits, AWGN, and error recorder module are implemented in field programmable gate array (FPGA). In the following, the encoder and decoder architecture on the hardware platform are reported, respectively. For the LDPC decoder, the offset min-sum (OMS) algorithm is used and the fixed-point quantization bit width is six for prior information and extrinsic information. Using this hardware evaluation platform, on one hand, we can accelerate the performance simulation. On the other hand, we can evaluate the effects of fixed-point implementation of the decoding algorithms. Therefore, we can quickly obtain more practical results for applications.

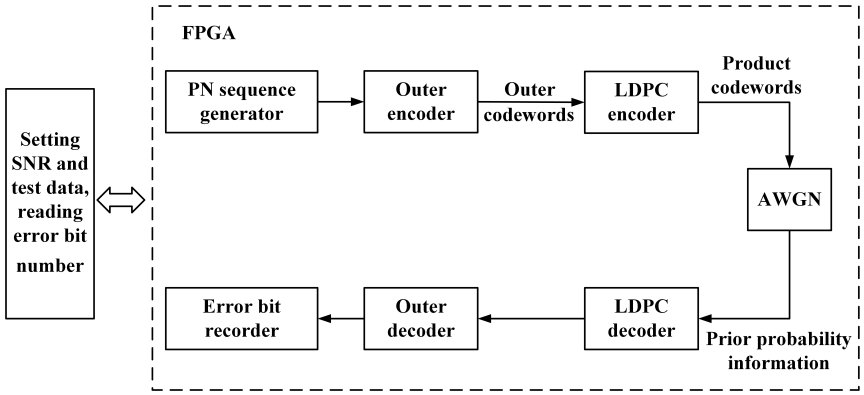

Figure 9. The hardware evaluation platform.

4.1.1. The Encoder Architecture

The encoder architecture of product codes is depicted in Figure 10. In our implementation, multiple parallel encoders composed of nine parallel outer encoders and six parallel LDPC encoders are designed for achieving the trade-off between complexity and throughput. Furthermore, the RAM matrix consisting of 54 independent block RAMs is used to simplify the outer codeword storage.

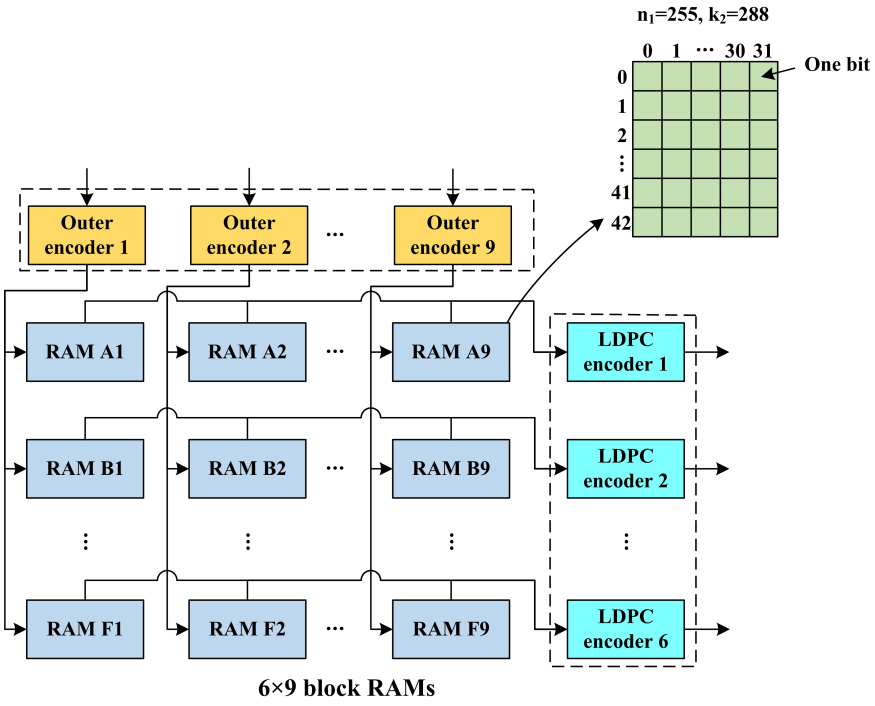

Figure 10. The encoder architecture of product codes.

During the outer encoding process, k_2 columns of information bits are divided into nine groups and input into nine outer encoders respectively. The information bits in the same group are encoded by the same encoder. Due to the parallel design, nine outer codewords can be obtained in parallel and further stored in block RAM matrix. For RAM, the read/write bit width is 1 and the depth is $n_1/6 \times k_2/9$. The block memory is logically divided into $n_1/6$ rows and $k_2/9$ columns according to the read/write address generating module.

For convenience and scalability, the LDPC encoding is performed after all the outer codewords are stored in the RAM matrix. The data in the block RAM matrix is read and sent into the six parallel LDPC encoders. Thus, the virtual columns in the same position of six block RAMs store one single outer

codeword. Specifically, the virtual rows in the same position of nine block RAMs store the information vector for one LDPC codeword. For each LDPC encoder, it encodes $n_1/6$ LDPC codewords sequentially.

4.1.2. The Decoder Architecture

For our proposed iterative scheme, the prior information and the intermediate information bits after LDPC decoding should be updated. Thus, we use two groups of block RAM for storing these information for global iterations.

Specifically, as shown in Figure 11, the decoder is composed of 162 block RAMs, six parallel LDPC decoders and nine parallel outer decoders. These block RAMs are used to store the referred two parts of information. Some RAMs are for the prior probability imformation storage. We use 108 block RAMs to store the prior probability imformation, each of which has the word width of Q bits and the depth of $n_1/9 \times k_2/6$. Other 54 block RAMs store the intermediate decoding results of LDPC codes during the global iterations. These block RAMs are organized in the same way as those in the encoder.

The specific iterative decoding process is presented as follows.

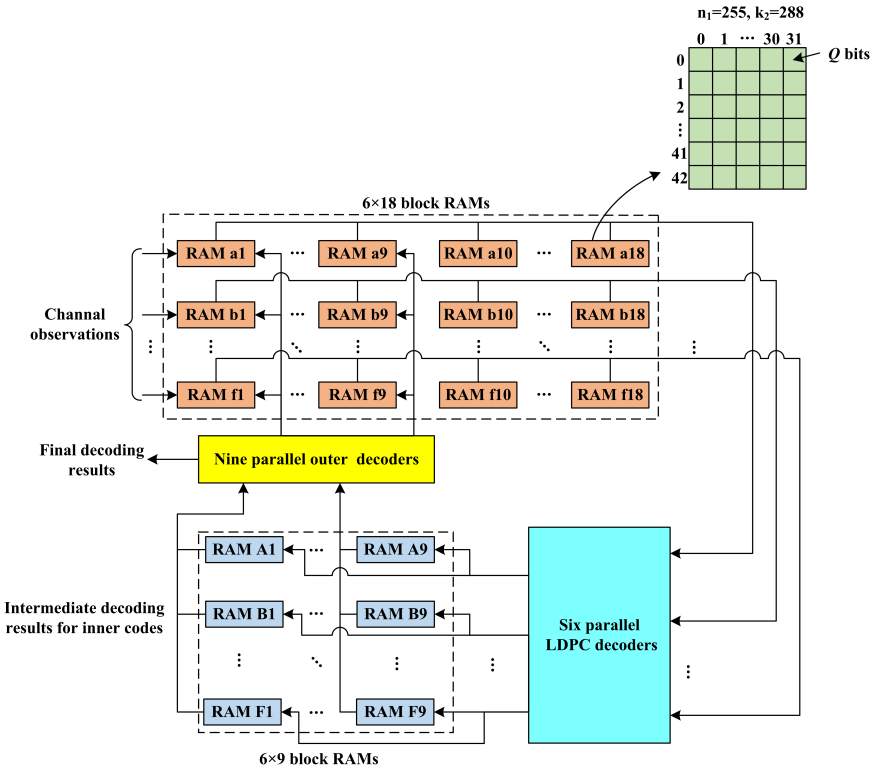

Figure 11. The decoder architecture of product codes.

Step 1: Update n_2 prior probability information for each LDPC codeword according to the received channel observation values.

Step 2: Perform the inner decoding using multiple parallel LDPC decoders and store the intermediate bits in decoding results in 54 block RAMs.

Step 3: Perform error-trapping decoding using nine parallel outer decoders.

Step 4: Update the prior probability information according to the intermediate outer decoding results until the maximum number of iterations has been reached.

Step 5: Output the results of outer decoders in the final iterative round. Otherwise, update the prior probability information and go to step 2.

4.2. Experimental Results Based on Field Programmable Gate Array

The FPGA for our hardware platform is XC6VLX240T FPGA of Virtex-6 series manufactured by Xilinx (San Jose, CA, USA), considering the inner hardware resources needed by our encoders and decoders. The performance of product codes constructed by three Hamming codes and LDPC codes in IEEE 802.16e with code length of 576 bits and code rate of 0.5 is tested using the implemented platform.

For the LDPC decoder, we adopt the OMS algorithm and the number of iterations is 30. For the Hamming decoder, we use the error-trapping decoding algorithm. For the prior probability information and extrinsic information, six bits fixed-point quantization is adopted, in which the integer and decimal parts are both three bits. In our implementation, the circuit can work at the clock frequency of 150 MHz. When the clock frequency is fixed, the throughput is mainly determined by the number of iterations of product codes. The throughput of different product codes calculated according to the data before decoding is shown in Table 1. It can be observed that, for the decoder using different Hamming codes, the maximum throughput without global iteration is similar, though, for longer Hamming codes, the throughput is a little larger. When global iteration between LDPC decoders and Hamming decoders is used, the throughput is linearly scaled by an iteration number.

Table 1. Throughput of the iterative decoders for different product codes.

Outer Code	Iteration Number	Throughput
Hamming (127, 120)	1	151 Mbps
	2	83 Mbps
	5	36 Mbps
Hamming (255, 247)	1	166 Mbps
	2	93 Mbps
	5	41 Mbps
Hamming (511, 502)	1	175 Mbps
	2	98 Mbps
	5	41 Mbps
Hamming (1023, 1013)	1	180 Mbps
	2	101 Mbps
	5	44 Mbps

For the codec of product codes, the outer code length and the number of iterations have little effect on the logical resources occupied. Thus, different codecs for different product codes occupy almost the same logical resource. It can be observed that the increased complexity due to the outer codes is quite limited, especially for the decoder. For example, nine Hamming decoders only use 385 slices, which can be neglected compared with six LDPC decoders. Unfortunately, the memory resource is greatly increased in order to store the intermediate codewords or probability information. The logical resources usage of the hardware evaluation platform of the product code with Hamming code (1023, 1013) is shown in Table 2.

The hardware test result of iterative decoding of the product code with Hamming (127, 120) is shown in Figure 12. It can be observed that, when BER is 10^{-6}, the product code without iteration has 0.45 dB coding gain compared with the LDPC code. In addition, the product code with two iterations has the coding gain of 0.45 dB over the product code without iteration. In addition, the product code with five iterations has the coding gain of 0.2 dB compared with the product code with two iterations.

Table 2. The resource usage of hardware evaluation platform.

	Module	Block RAMs	Slices
Encoder	Hamming encoder	9	356
	LDPC encoder	42	2687
	Memory for outer codewords	54	\
Decoder	Hamming decoder	18	385
	LDPC decoder	200	18,932
	Memory for prior information	216	\
	Memory for intermediate dec results	54	\
XC6VLX240T usage (percent)		832 (71.27%)	22,360 (59.34%)

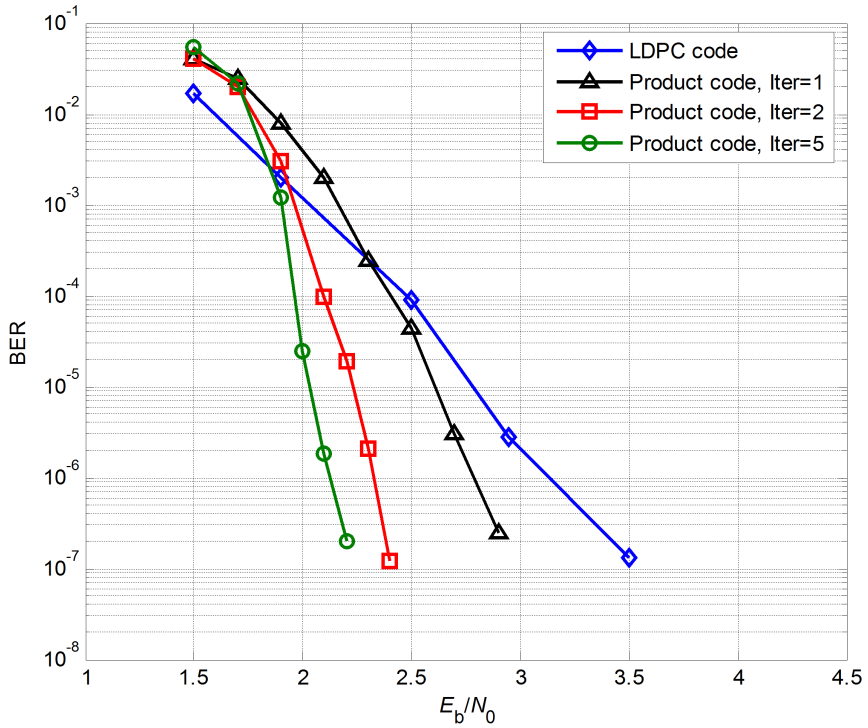

Figure 12. FPGA evaluation results of product codes with LDPC (576, 288) and Hamming (127, 120).

In addition, the hardware test results of product codes using outer codes with different code rates and code lengths are also illustrated, as shown in Figures 13–15. The performance of product codes with iterative decoding has obvious gain over the product code without iteration. It also can be observed that only very few iterations are needed for the proposed schemes, for when the iterative number increases, the additional gain becomes smaller. In summary, the test results verify the superior performance of iterative decoding methods for LDPC-based product codes.

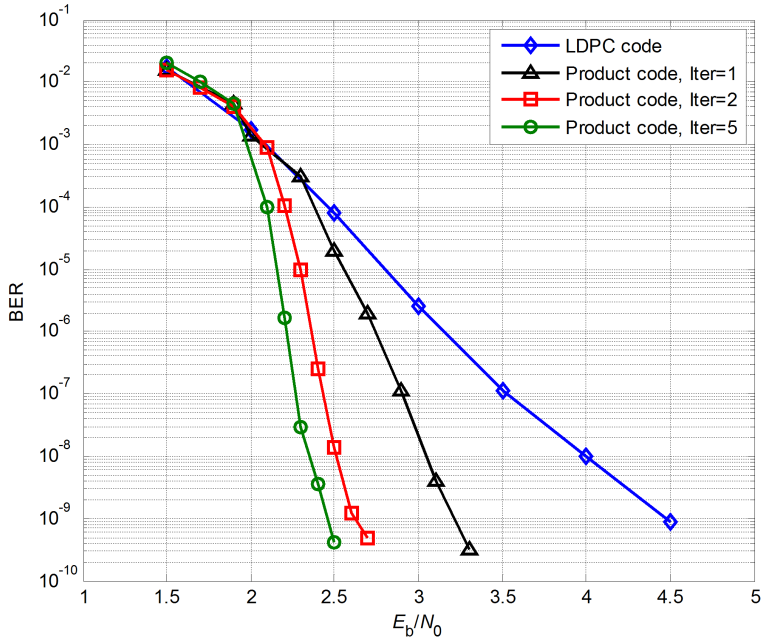

Figure 13. FPGA evaluation results of product codes with LDPC (576, 288) and Hamming (255, 247).

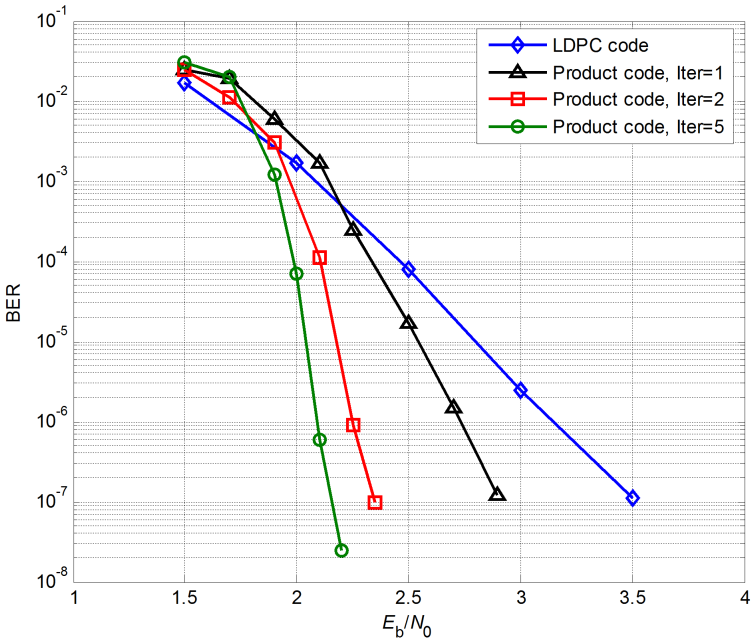

Figure 14. FPGA evaluation results of product codes with LDPC (576, 288) and Hamming (511, 502).

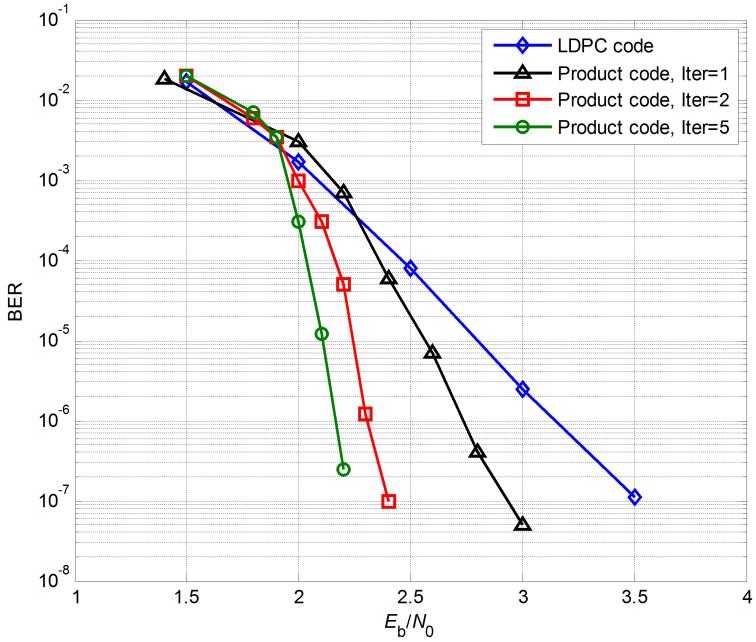

Figure 15. FPGA evaluation results of product codes with LDPC (576, 288) and Hamming (1023, 1013).

5. Conclusions

In this paper, LDPC-based product codes under iterative decoding are evaluated using FPGA. The product code with iterative decoding can reduce the error floor and reach an extremely low BER. Furthermore, considering the limitations of software simulation, the hardware test platform is built to accelerate the performance evaluation and evaluate the performance loss due to the fixed-point implementation. When BER is 10^{-9}, the product code with iterative decoder has the performance gain of 2.0 dB compared with the LDPC code of equivalent code rate. Meanwhile, the evaluation using FPGA also proves that using iterative decoding of product codes with LDPC codes and high-rate algebraic codes can achieve a good trade-off between complexity and throughput compared with a single long LDPC code.

Author Contributions: W.C. proposed the methods and wrote the paper. W.Z. performed simulations and revised the manuscript. H.L. participated in the simulations and revising the manuscript. S.D. participated in revising the manuscript. C.H. participated in designing the scheme. J.Y. participated in designing the scheme. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China (61671324), the Director's Funding from Pilot National Laboratory for Marine Science and Technology (Qingdao), and the Seed Foundation of Tianjin University.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, T.; Wellbrock, G.; Ishida, O. Next generation optical transport beyond 100G. *IEEE Commun. Mag.* **2012**, *50*, 10–11. [CrossRef]
2. Yamazaki, E.; Tomizawa, M.; Miyamoto, Y. 100-Gb/s optical transport network and beyond employing digital signal processing. *IEEE Commun. Mag.* **2012**, *50*, 43–49. [CrossRef]
3. Djordjevic, I.B. On advanced FEC and coded modulation for ultra-high-speed optical transmission. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 1920–1951. [CrossRef]
4. Tzimpragos, G.; Kachris, C.; Djordjevic, I.B. A survey on FEC codes for 100 G and beyond optical networks. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 209–221. [CrossRef]

5. Mizuochi, T.; Konishi, Y.; Miyata, Y. Experimental demonstration of concatenated LDPC and RS codes by FPGAs emulation. *IEEE Photonics Technol. Lett.* **2009**, *18*, 1302–1304. [CrossRef]
6. Sheikh, A.; Amat, A.G.I.; Liva, G.; Christian, H.; Pfister, H.D. On low-complexity decoding of product codes for high-throughput fiber-optic systems. In Proceedings of the IEEE International Symposium on Turbo Codes & Iterative Information Processing (ISTC), Hong Kong, China, 3–7 December 2018.
7. Fougstedt, C.; Larsson-Edefors, P. Energy-efficient high-throughput VLSI architectures for product-like codes. *J. Light. Technol.* **2019**, *37*, 477–485. [CrossRef]
8. Fougstedt, C.; Sheikh, A.; Amat, A.G.I.; Liva, G.; Larsson-Edefors, P. Energy-efficient soft-assisted product decoders. In Proceedings of the 2019 Optical Fiber Communications Conference and Exhibition (OFC), San Diego, CA, USA, 3–7 March 2019.
9. Furrer, S.; Lantz, M.A.; Reininger, P.; Ysamaguchi, A. 201 Gb/in^2 recording areal density on sputtered magnetic tape. *IEEE Trans. Magn.* **2018**, *54*, 1–8. [CrossRef]
10. Gallager, R.G. Low-density parity-check codes. *Trans. IRE Prof. Group Inf. Theory* **1962**, *8*, 21–28. [CrossRef]
11. Mackay, D.J.C.; Neal, R.M. Near Shannon limit performance of low density parity check codes. *Electron. Lett.* **1996**, *32*, 1645–1646. [CrossRef]
12. Richardson, T.J.; UrbankeNeal, R.L. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Trans. Inf. Theory* **2001**, *47*, 599–618. [CrossRef]
13. Richardson, T.J. Error floors of LDPC codes. In Proceedings of the Annual Allerton Conference on Communication, Control and Computing, Monticello, VA, USA, 1–3 October 2003.
14. Dolecek, L.; Lee, P. Zhang, Z.; Anantharam, V.; Nikolic, B.; Wainwright, M. Predicting error floors of structured LDPC codes: Deter-ministic bounds and estimates. *IEEE J. Sel. Areas Commun.* **2009**, *27*, 908–917.
15. Kyung, G.B.; Wang, C.C. Finding the exhaustive list of small fully absorbing sets and designing the corresponding low error floor decoder. *IEEE Trans. Commun.* **2012**, *60*, 1487–1498. [CrossRef]
16. Li, J.; Liu, K.; Lin, S.; Abdel-Ghaffar, K. Algebraic quasi-cyclic LDPC codes: Construction, low error-floor, large girth and a reduced-complexity decoding scheme. *IEEE Trans. Commun.* **2014**, *62*, 2626–2637. [CrossRef]
17. Zhao, S; Ma, X. Construction of high-performance array-based non-binary LDPC codes with moderate rates. *IEEE Commun. Lett.* **2015**, *20*, 13–16.
18. Li, J.; Liu, K.; Lin, S.; Abdel-Ghaffar, K. A matrix-theoretic approach to the construction of non-binary quasi-cyclic ldpc codes. *IEEE Trans. Commun.* **2015**, *63*, 1057–1068. [CrossRef]
19. Chen, W.; Yin, L.; Lu, J. Error floor behavior study of LDPC codes for concatenated codes design. In Proceedings of the Second International Conference on Space Information Technology (SPIE), Wuhan, China, 10–11 November 2007; Volume 6795.
20. Eroz, M.; Sun, F.W.; Lee, L.N. DVB-S2 low density parity check codes with near Shannon limit performance. *Int. J. Satell. Commun. Netw.* **2004**, *22*, 269–279. [CrossRef]
21. Li, K.; Motani, M. The error distribution of linear block bodes. In Proceedings of the IEEE International Symposium on Information Theory, Yokohama, Japan, 29 June–4 July 2003; p. 464.
22. Elias, P. Error-free coding. *Trans. IRE Prof. Group Inf. Theory* **1954**, *4*, 29–37. [CrossRef]
23. Chen, W.; Dong, T. Low complexity product codes with LDPC codes achieving ultra low BER. In Proceedings of the Communication Technology (ICCT), Chengdu, China, 9–11 November 2012; pp. 1312–1316.
24. Zhang, Z.; Dolecek, L.; Nikolic, B.; Anantharam, V.; Wainwright, M. J. Design of LDPC decoders for improved low error rate performance: Quantization and algorithm choices. *IEEE Trans. Commun.* **2009**, *57*, 3258–3268. [CrossRef]
25. Lin, S.; Costello, D.J. *Error Control Coding*, 2nd ed.; Prentice Hall: New York, NY, USA, 2004; pp. 194–231.
26. Liu, Y.; Chen, W. A local flat histogram method for evaluating the performance of LDPC codes. In Proceedings of IEEE Wireless Communications & Networking Conference (WCNC), New Orleans, LA, USA, 9–12 March 2015.