IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 9, 2019, accepted April 18, 2019, date of publication April 30, 2019, date of current version May 16, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2914076

Variable-Node-Based Belief-Propagation Decoding With Message Pre-Processing for NAND Flash Memory

XINGCHENG LIU[1,2], (Senior Member, IEEE), GUOJUN YANG[1], AND XUECHEN CHEN[1]
[1]School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China
[2]School of Information Science, Xinhua College, Sun Yat-sen University, Guangzhou 510520, China

Corresponding authors: Xingcheng Liu (isslxc@mail.sysu.edu.cn) and Xuechen Chen (chenxchen8@mail.sysu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61572534 and Grant 61873290, and in part by the Special Project for Promoting Economic Development in Guangdong Province under Grant GDME-2018D004.

ABSTRACT With the fast development of non-volatile storage technology, NAND flash memory faces more and more challenges such as data reliability and lifetime. To overcome the issue of the reliability, low-density parity-check (LDPC) codes are considered as a main candidate of error-correction-codes (ECCs) for NAND flash storages. However, conventional soft decoding algorithms for LDPC codes suffer from the drawback of slow convergence speed, which increases the decoding latency. For achieving fast convergence speed, a variable-node-based belief-propagation with message pre-processing (VNBP-MP) decoding algorithm for binary LDPC codes and a non-binary VNBP-MP (NVNBP-MP) decoding algorithm for non-binary LDPC codes are proposed. Both of the algorithms utilize the characteristics of the NAND flash channel to perform the message pre-processing operations. In theory, the propagation of unreliable messages can be effectively prevented and the propagation of reliable messages can be speeded up. To further improve the decoding convergence, the treatment for oscillating variable nodes is considered after the message pre-processing operations. The simulation results also show that the proposed algorithms both have a noticeable improvement in convergence speed and latency, without compromising error-correction performance, compared with the existing soft decoding algorithms.

INDEX TERMS NAND flash memory, low-density parity-check (LDPC) code, message pre-processing, belief-propagation decoding, error-correction performance.

I. INTRODUCTION

NAND flash memory is a non-volatile storage technology which has been widely used in numerous computing systems, data storage systems and consumer electronics such as mobile phones, MP3/MP4 players, digital cameras and solid-state drives (SSDs) [1] [2] because of its noticeable advantages of high performance, small physical size, large storage capacity and low power consumption. To meet the requirement of larger storage capacity and lower cost per bit, the industry reduces the cell size of the NAND flash memory and makes each memory cell store more bits. Based on the difference in the number of the bits stored in each flash memory cell, up to now, the flash memory cells can be divided into four types: single-level cell (SLC, 1bit/cell), multi-level cell (MLC, 2bits/cell), triple-level cell (TLC, 3bits/cell) and quad-level cell (QLC, 4bits/cell) [3], [4].

However, aggressive scaling down of the cell size makes each memory cell store fewer electrons, and narrows the voltage margin between the adjacent storage states. Moreover, as the program/erase (PE) cycles and retention ages increase, it leads to severe noises which incorporate programming noise, cell-to-cell interference (CCI), random telegraph noise (RTN) and data retention noise [5] [6] [7]. Therefore, the data reliability of the NAND flash memory storage is largely degraded, and the lifetime is hence shortened. Therefore, improving the error-correction performance and extending the operational lifetime of the NAND flash memory have become a valuable research hot spot in the field of storage media.

The associate editor coordinating the review of this manuscript and approving it for publication was Soon Xin Ng.

For the purpose of overcoming the reliability issues of the NAND flash memory, it is essential to employ the technology of error-correction-codes (ECCs). Traditional Bose-Chaudhuri-Hocquenghem (BCH) codes are used to guarantee the data integrity of commercial NAND flash storage systems [8]. However, as the cell size becomes smaller and smaller, the raw bit error rate grows so rapidly that BCH codes with hard-decision decoding are becoming inadequate to correct a great number of bit errors [9]. In this situation, ECCs with outstanding error-correction capability are urgently demanded for ensuring the data reliability. Therefore, low-density parity-check (LDPC) codes have been considered as a major candidate of ECCs for future NAND flash storage systems. The soft-decision belief-propagation (BP) decoding algorithms for LDPC codes are adopted to correct the errors induced by the NAND flash channel noises.

The iterative soft-decision BP decoding algorithms have the promising error-correction performance, compared with the hard-decision decoding algorithms for BCH codes. BP scheduling strategies can be divided into three types, flooding scheduling (FS) [10], standard serial scheduling (SSS) [11] [12] and informed dynamic scheduling (IDS) [13], respectively. SSS achieves more outstanding error-correction performance than FS, and provides convergence speed twice as fast as FS [14]. IDS has an obvious improvement in the convergence speed and the error-correction performance, compared with SSS. However, the decoding complexity of IDS algorithms is extremely high because all of them consume a lot of resources for computing residuals and reordering the updating sequence. Therefore, IDS is not suitable for the applications of NAND flash storage systems, which require the low complexity and decoding latency.

In recent years, many BP-based algorithms are applied to the error correction of the NAND flash memory. For example, considering the effect of data retention noise, the retention-aware belief-propagation (RA-BP) decoding scheme is employed to recover the retention errors without additional memory sensing operations [15]. To improve the LDPC decoding throughput for the MLC NAND flash-based storage, the intra-cell data dependence aware decoding algorithm is proposed [16]. This algorithm provides additional belief information for cross fields to help to improve the estimation accuracy. To improve the convergence speed, the low-complexity quantization-aware belief-propagation (QABP) decoding scheme with the unequal allocation of computational resources is presented [17]. It skips the updating of the variable nodes with low error probability during the first few iterations.

The intra-cell data dependence aware algorithm and QABP algorithm both refer to the exploitation of the flash channel under the condition of high PE cycles. However, the above two algorithms and conventional serial algorithm [11] have no consideration for the phenomenon of the propagation of unreliable messages and the effect of the oscillating variable nodes. Thus, as the channel condition steadily deteriorates, the NAND flash suffers from the degradation in performance, especially in the TLC or quadruple-level cell (QLC, 4bits/cell) NAND flash. To further speed up the convergence, shorten the latency and achieve better error-correction performance of soft-decision BP algorithms, the propagation of unreliable messages should be prevented in the initial stage of iterations, and the oscillating variable nodes should be processed in an appropriate way. Indeed, a variable-node-based belief-propagation with message pre-processing (VNBP-MP) decoding algorithm for binary LDPC codes, and a non-binary VNBP-MP (NVNBP-MP) decoding algorithm are proposed to achieve the above-mentioned objectives.

The difference among the proposed decoding algorithm and the existing BP-based decoding algorithms is that the former considered the propagation of unreliable messages and the effect of the oscillating variable nodes, and took the corresponding measures to prevent them from affecting the error correction performance, while the latter did not. The major contributions of this paper are as follows:

- According to a large number of the statistics which incorporate the average error probability for each variable node, the characteristic of the NAND flash channel is found, which is a part of variable nodes getting into overlapping regions are prone to be erroneous. Based on the channel characteristic, variable nodes are divided into reliable ones and unreliable ones.
- Motivated by the channel characteristic, the message pre-processing (MP) strategy is proposed. The MP strategy prevents the unreliable messages from propagating to reliable variable nodes. At the same time, computational resources are preferentially allocated to update unreliable variable nodes with reliable messages. Therefore, reliable messages are utilized effectively in the initial stage of decoding iterations, and hence the decoding process can continue towards the right direction.
- In order to further speed up the decoding convergence, the treatment for the oscillation phenomenon should be considered. To this end, an oscillation-based alternate-selection-scheduling (OB-ASS) strategy is proposed to process the corresponding oscillating variable nodes after performing the MP operations.
- As stated above, combining the MP and the OB-ASS strategies, the VNBP-MP and NVNBP-MP decoding algorithms are proposed for achieving better error-correction performance, faster convergence speed and lower decoding delay. The computer simulations demonstrate that, compared with the existing decoding algorithms, the proposed algorithms both have outstanding improvements not only in the error-correction performance but the convergence speed and decoding throughput.

The rest of the paper is organized as follows. In Section II, we introduce the preliminaries about MLC NAND flash memory and conventional soft-decision algorithms, including the characteristic of the NAND flash channel. In Section III, the proposed VNBP-MP and NVNBP-MP decoding

algorithms are described in detail. The simulation results about the error-correction performance, convergence speed and decoding latency are presented and analyzed in Section IV. Finally, we conclude the paper in Section V.

II. PRELIMINARY AND MOTIVATION

A. MLC NAND FLASH MEMORY CHANNEL MODEL

An SLC flash memory cell can store only one bit which represents data '1' or '0', while an MLC flash memory cell stores two bits which are configured to four different voltage states for representing symbols '11', '10', '00' and '01' [18]. Four voltage states are denoted by V_{min}, V_1, V_2 and V_{max} to represent their own average threshold voltages, respectively.

The voltage state of a NAND flash memory cell can be determined by injecting or removing a certain amount of charges into the floating gate [6]. This process leverages Fowler-Nordheim tunneling and represents the standard PE operation for NAND flash memory. The voltage value for each cell is proportional to the amount of charge stored in a floating gate. However, the actual voltage value is shifted to an uncertain value because the amount of charge stored in a floating gate varies with the different PE cycles and retention ages. As PE cycles and retention ages continuously increase, a flash memory cell is severely disturbed by a variety of noises to the extent that its voltage value has been obviously shifted. After the voltage value shifting, errors are induced as the result of the voltage state difference before and after noise interference.

To simulate the actual MLC NAND flash memory, the noises including the programming noise, CCI, RTN and data retention noise [7] [18] [19] are considered to establish the MLC NAND flash channel model. The overall threshold voltage distributions of four voltage states, $p_{s_{11}}$, $p_{s_{10}}$, $p_{s_{00}}$ and $p_{s_{01}}$, computed as the convolutional integral of the corresponding initial threshold voltage distribution and the distributions induced by a variety of noises [18], are given by

$$p_{s_{11}}(v) = \frac{1}{\sigma_{s_{11}}\sqrt{2\pi}} e^{-\frac{\left(v-\left(\widetilde{V}_{min}-\mu_{r_{s_{11}}}\right)\right)^2}{2\sigma_{s_{11}}^2}}, \tag{1}$$

$$p_{s_{10}}(v) = \frac{1}{\Delta V_{pp}}\left(erf\left(\frac{V_1+\Delta V_{pp}-v-\mu_{r_{s_{10}}}}{\sqrt{2}\sigma_{s_{10}}}\right)\right) - \frac{1}{\Delta V_{pp}}\left(erf\left(\frac{V_1-v-\mu_{r_{s_{10}}}}{\sqrt{2}\sigma_{s_{10}}}\right)\right), \tag{2}$$

$$p_{s_{00}}(v) = \frac{1}{\Delta V_{pp}}\left(erf\left(\frac{V_2+\Delta V_{pp}-v-\mu_{r_{s_{00}}}}{\sqrt{2}\sigma_{s_{00}}}\right)\right) - \frac{1}{\Delta V_{pp}}\left(erf\left(\frac{V_2-v-\mu_{r_{s_{00}}}}{\sqrt{2}\sigma_{s_{00}}}\right)\right), \tag{3}$$

$$p_{s_{01}}(v) = \frac{1}{\Delta V_{pp}}\left(erf\left(\frac{V_{max}+\Delta V_{pp}-v-\mu_{r_{s_{01}}}}{\sqrt{2}\sigma_{s_{01}}}\right)\right) - \frac{1}{\Delta V_{pp}}\left(erf\left(\frac{V_{max}-v-\mu_{r_{s_{01}}}}{\sqrt{2}\sigma_{s_{01}}}\right)\right), \tag{4}$$

$$erf(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-v^2}dv, \tag{5}$$

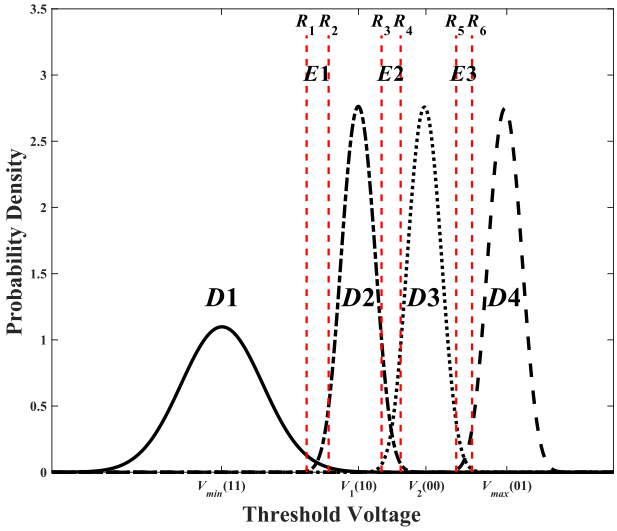


FIGURE 1. Threshold voltage distribution functions for MLC NAND flash memory.

where $\mu_{r_{s_{ij}}}$ and $\sigma_{s_{ij}}$ ($ij \in \{11, 10, 00, 01\}$) represent the mean shift induced by the data retention noise and the standard deviation of the overall threshold voltage distribution, respectively. $\widetilde{V}_{min}$ denotes the average threshold voltage affected by CCI, and ΔV_{pp} the programming voltage step size. The configuration of parameters mentioned in the threshold voltage distribution functions refers to [18]. In this way, the MLC NAND flash memory channel model can be set up, as Fig. 1 shows.

B. NON-UNIFORM MEMORY SENSING STRATEGY AND CALCULATION FOR SOFT INFORMATION

To get the soft information stored in each flash memory cell after the noise interference, the read operations should be executed. A certain number of read reference voltages are applied to the control gate of a transistor to identify which region a memory cell belongs to. Therefore, each variable node gets the soft information according to its located region. Compared with the conventional uniform memory sensing strategy, a non-uniform memory sensing strategy [19] [20] is executed, in order to reduce the read latency without degrading the error-correction performance. In addition, further increase in memory sensing levels can produce diminishing returns [15]. Figure 1 shows that 6 non-uniform read reference voltages, R_1, R_2, R_3, R_4, R_5 and R_6, are configured to divide the threshold voltage range into 7 regions, which incorporate 4 data regions ($D1$, $D2$, $D3$, $D4$) and 3 overlapping regions ($E1$, $E2$, $E3$). The default read operation is

that, the upper page requires 2 reads for the most significant bit (MSB) and the lower page requires 4 reads for the least significant bit (LSB).

The configuration of read reference voltages is based on the voltage entropy function [18], given by

$$H(v) = \sum_i \left[\frac{p_{s_i}(v)}{\sum_i p_{s_i}(v)} log_2 \left(\frac{\sum_i p_{s_i}(v)}{p_{s_i}(v)} \right) \right], \tag{6}$$

where $i \in \{11, 10, 00, 01\}$. The values of read reference voltages are set to satisfy the condition of $H(R_n) = 0.35$ [18].

Based on the threshold voltage distribution functions of 4 states including $p_{s_{11}}$, $p_{s_{10}}$, $p_{s_{00}}$ and $p_{s_{01}}$, and 6 specified read reference voltages, the log likelihood ratio (LLR) information for each region can be calculated. An MLC flash memory cell can store two bits, the MSB and the LSB [15]. In order to transform the voltage stored in an MLC flash memory cell into soft-decision information, the LLR of MSB and LSB can be calculated [17] by

$$L_{msb}(n) = log \frac{\int_{R_{n-1}}^{R_n} \left\{ p_{s_{00}}(v) + p_{s_{01}}(v) \right\} dv}{\int_{R_{n-1}}^{R_n} \left\{ p_{s_{10}}(v) + p_{s_{11}}(v) \right\} dv}, \tag{7}$$

$$L_{lsb}(n) = log \frac{\int_{R_{n-1}}^{R_n} \left\{ p_{s_{00}}(v) + p_{s_{10}}(v) \right\} dv}{\int_{R_{n-1}}^{R_n} \left\{ p_{s_{01}}(v) + p_{s_{11}}(v) \right\} dv}, \tag{8}$$

where $n \in \{1, 2, 3, 4, 5, 6, 7\}$ denotes the index of seven regions, $D1$, $E1$, $D2$, $E2$, $D3$, $E3$ and $D4$. v represents the voltage with a range of $R_{n-1} \leq v \leq R_n$, where $R_0 = -\infty$ and $R_7 = +\infty$.

C. STANDARD SERIAL SCHEDULING FOR LDPC CODES

An LDPC code can be described with an $M \times N$ matrix H, where M represents the number of check nodes c_m for $1 \leq m \leq M$ and N the number of variable nodes v_n for $1 \leq n \leq N$ [21]. For convenience, the matrix can also be considered as a bipartite graph, Tanner graph, where the message propagation between v_n and c_m can be clearly shown. For binary LDPC codes, the flooding log-likelihood ratios BP (LLRBP) algorithm [10] and the serial shuffled BP (SBP) algorithm [11] both compute the soft information in the logarithmic domain. Nevertheless, the LLRBP algorithm suffers from slow convergence speed and low error-correction performance. The message updates of the SBP algorithm are given as follows [22]

$$m^{(i)}_{c_m \to v_n} = 2tanh^{-1} \left(\prod_{v_j \in N(c_m), j<n} tanh \left(\frac{m^{(i)}_{v_j \to c_m}}{2} \right) \times \prod_{v_k \in N(c_m), k>n} tanh \left(\frac{m^{(i-1)}_{v_k \to c_m}}{2} \right) \right), \tag{9}$$

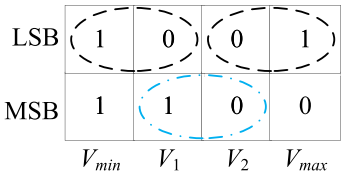


FIGURE 2. The difference of stored bits between adjacent voltage states in the MLC NAND flash memory.

$$m^{(i)}_{v_n \to c_m} = C_{v_n} + \sum_{c_j \in M(v_n) \backslash c_m} m^{(i)}_{c_j \to v_n}, \tag{10}$$

$$L^{(i)}_{v_n} = C_{v_n} + \sum_{c_j \in M(v_n)} m^{(i)}_{c_j \to v_n}, \tag{11}$$

where, C_{v_n} is the same as L_{msb} computed by (7) or L_{lsb} by (8). i represents the current iteration, while $i-1$ represents the previous iteration. The check-to-variable (C2V) messages from c_m to v_n, $m^{(i)}_{c_m \to v_n}$, can be updated by exploiting the new variable-to-check (V2C) messages $m^{(i)}_{v_j \to c_m} (j < n)$ and old ones $m^{(i-1)}_{v_k \to c_m} (k > n)$. Therefore, the SBP algorithm has faster convergence and better error-correction performance, than the LLRBP algorithm [11]. $N(c_m)$ denotes the set that includes all variable nodes connected to check node c_m and $M(v_n) \backslash c_m$ represents the set that includes all check nodes connected to variable node v_n except check node c_m. $L^{(i)}_{v_n}$ denotes the decoded LLR value after the update with (11) for making a hard decision.

For the specified scheduling strategy, the decoding procedure for non-binary LDPC codes is the same as that for binary LDPC codes. However, unlike the computation of V2C, C2V and LLR values for binary LDPC codes, the computation of the corresponding soft information for non-binary LDPC codes is performed in the probability domain.

D. MOTIVATION

As Fig. 2 shows, there is only one bit difference between the adjacent voltage states, as a consequence of the usage of gray code. Therefore, one bit of the flash memory cell, which is fallen in the overlapping region, is likely to be correct. To obtain the characteristic of the MLC NAND flash channel, 1 GB random data, generated with the uniform distribution of '0' and '1', is programmed and read in the simulated model. The number of PE cycles is set as 15000, and the data retention time (T) is set as 1.

Based on the above experiments, Table 1 is obtained. In the table, N_r and E_r denote the number of total bits and the number of erroneous bits, respectively. Accordingly, the average error probability, denoted as $\overline{P_r}$ (where $r \in \{D1, E1, D2, E2, D3, E3, D4\}$), can be computed. For example, N_{E1} represents the number of bits fallen in region $E1$, and $\overline{P_{D2}}$ represents the average error probability of each bit fallen in region $D2$. As observed in Table 1, the bits in the 3 overlapping regions have higher error probability than those in the 4 data regions. However, the intracell data dependence [16] shows that the MSBs in region $E2$, and the

TABLE 1. Statistics of bits in each region.

	D1	*E1*	*D2*	*E2*	*D3*	*E3*	*D4*
N_r	2.11×10^9	9.10×10^7	1.95×10^9	2.80×10^8	1.97×10^9	7.74×10^7	2.11×10^9
E_r	2.68×10^5	1.56×10^7	7.21×10^6	6.79×10^7	3.40×10^6	1.78×10^7	8.08×10^5
$\overline{P_r}$ (%)	0.013	17.119	0.370	24.215	0.173	23.071	0.038

TABLE 2. Statistics of MSBs and LSBs in each overlapping region.

	E1_msb	*E1_lsb*	*E2_msb*	*E2_lsb*	*E3_msb*	*E3_lsb*
N_r	4.55×10^7	4.55×10^7	1.40×10^8	1.40×10^8	3.87×10^7	3.87×10^7
E_r	0	1.56×10^7	6.78×10^7	4.17×10^4	0	1.78×10^7
$\overline{P_r}$ (%)	0	34.237	48.401	0.030	0	46.141

TABLE 3. The ratio of the minimum llr magnitude in G_2 to the maximum LLR magnitude in G_1.

PE	1.0×10^4	1.1×10^4	1.2×10^4	1.3×10^4	1.4×10^4	1.5×10^4
R_{LLR}	8.884	11.012	13.856	12.736	9.950	8.300

LSBs in regions $E1$ and $E3$ are likely to be correct. Hence, the MSBs and LSBs in the 3 overlapping regions should be separated, as Table 2 shows. The channel characteristic is obviously observed that the MSBs in region $E2$, and the LSBs in regions $E1$ and $E3$, have high error probability. In addition, our simulations also demonstrated that the other bits have relatively high reliability.

In addition to analyzing $\overline{P_r}$, the LLR magnitudes of MSB and LSB in each region are also considered. Since the MSBs in region $E2$, and the LSBs in regions $E1$ and $E3$, have high error probability, the corresponding LLR magnitudes, $|L_{msb}(4)|$, $|L_{lsb}(2)|$ and $|L_{lsb}(6)|$, are classified into a group, G_1. Corresponding to the other bits which tend to be correct, there are 11 LLR magnitudes, $|L_{msb}(1)|$, $|L_{lsb}(1)|$, $|L_{msb}(2)|$, $|L_{msb}(3)|$, $|L_{lsb}(3)|$, $|L_{lsb}(4)|$, $|L_{msb}(5)|$, $|L_{lsb}(5)|$, $|L_{msb}(6)|$, $|L_{msb}(7)|$ and $|L_{lsb}(7)|$, to be classified into a group, G_2. Each LLR magnitude in G_2 is much bigger than any other LLR magnitude in G_1. To explain the above phenomenon, R_{LLR} is denoted as the ratio of the minimum LLR magnitude in G_2 to the maximum LLR magnitude in G_1, given by

$$R_{LLR} = \frac{min\{G_2\}}{max\{G_1\}}, \tag{12}$$

as shown in Table 3. Jointly considering Tables 1, 2 and 3, a regular pattern is found that the LSBs in regions $E1$ and $E3$, and the MSBs in region $E2$, are prone to be erroneous. In addition, the bits in regions $D1$, $D2$, $D3$ and $D4$, the MSBs in regions $E1$ and $E3$, and the LSBs in region $E2$, have low error probability and the relatively large LLR magnitudes.

III. VARIABLE-NODE-BASED BELIEF-PROPAGATION DECODING WITH MESSAGE PRE-PROCESSING

To achieve faster convergence speed, lower decoding latency and better error-correction performance, the MP strategy motivated by the above regular pattern is proposed. Furthermore, considering the oscillation phenomenon, the OB-ASS strategy is described. Combined the MP strategy with the treatment for the oscillating variable nodes, the VNBP-MP algorithm for binary LDPC codes and NVNBP-MP algorithm for non-binary LDPC codes, are proposed.

A. VNBP-MP DECODING ALGORITHM FOR BINARY LDPC CODES

For binary LDPC codes, a variable node corresponds to one bit. Therefore, as the above regular pattern hinted, most of the variable nodes have the relatively large LLR magnitudes and small error probability. Over $GF(2)$, to utilize the regular pattern, each codeword should be equally stored at both of pages. Thus, the MSB and LSB pages are decoded simultaneously. The variable nodes are divided into two types, blurry ones and non-blurry ones. The blurry variable nodes, which correspond to the LSBs in regions $E1$ and $E3$, and the MSBs in region $E2$, have high error probability, as Table 2 shows. Indeed, the blurry ones are considered to be unreliable. In contrast, the non-blurry variable nodes, which correspond to the bits in regions $D1$, $D2$, $D3$ and $D4$, the MSBs in regions $E1$ and $E3$, and the LSBs fallen in region $E2$, are considered to be reliable with the relatively large LLR magnitudes.

Before the iteration, each V2C value is initialized with the LLR value of the corresponding variable node. If the C2V message is only updated with V2C messages from the non-blurry variable nodes, it is considered to be reliable and has comparatively large magnitude based on (9). Otherwise, once any unreliable V2C message generated from the blurry variable node participates in the update of the C2V message, the updated C2V message will become unreliable due to its indeterminate sign.

Conventional SBP decoding algorithm [11] serially updates all variable nodes in sequence. Therefore, a part of the non-blurry variable nodes with high reliability are updated

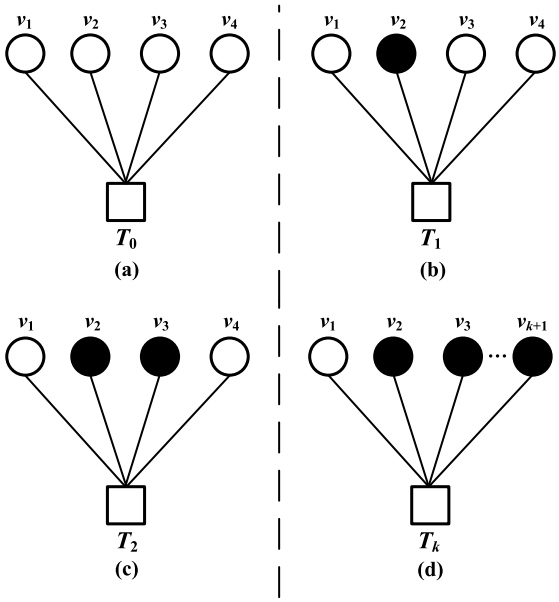


FIGURE 3. Various types of check nodes in the MP strategy.

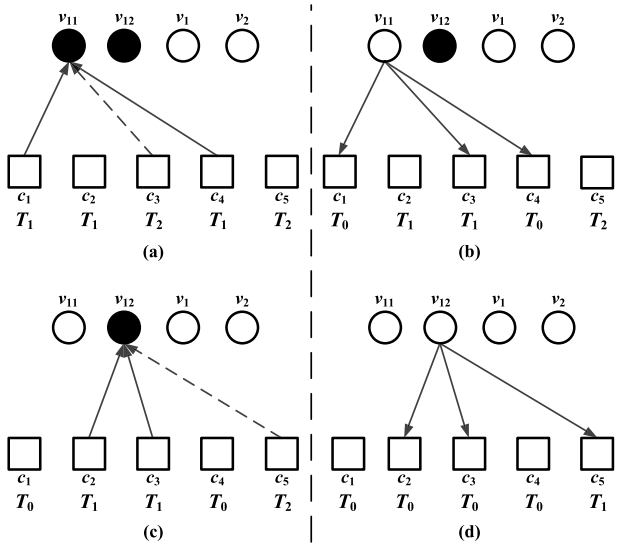


FIGURE 4. The decoding procedure at the first step in the MP strategy.

with the corresponding C2V messages, which are updated with a part of unreliable V2C messages from the blurry ones. Indeed, these updated non-blurry variable nodes are prone to be erroneous, and the convergence speed cannot be further improved due to the wide propagation of erroneous messages. Therefore, the MP strategy is proposed to stop the V2C messages associated with each blurry variable node from propagating in the initial stage of decoding. All the computational sources are allocated to update the blurry variable nodes with reliable V2C messages.

In the MP strategy, let V denote a blurry set, which is used to store the blurry variable nodes. Besides, T_0, T_1, T_2 and $T_k (k \geq 3)$ denote various types of check nodes respectively, as shown in Fig. 3. For example, a T_1 type check node connects only one blurry variable node. Black circles represent the blurry variable nodes while white circles represent the non-blurry ones. The specific procedure of MP strategy is divided into three steps, described as follows

1) STEP 1:

The MP strategy firstly puts all the blurry variable nodes into the blurry set, V, then searches all the blurry ones from V in sequence. The blurry variable node, v_{1j}, connected with at least two T_1 type check nodes, will be chosen to be updated. The variable nodes, with which each T_1 type check node connects, are all the non-blurry ones except for v_{1j}. Therefore, the updated C2V messages from T_1 type check nodes to v_{1j} are all reliable according to (9), as previously mentioned. Except for v_{1j}, a T_2 type check node connects another blurry variable node and a T_k type check node connects two or more blurry ones. The update of each C2V message associated with T_2 or T_k type check node, exploits at least one V2C message associated with the blurry variable node. Hence, the updated C2V message from T_2 or T_k type check node to v_{1j} are unreliable. To prevent the propagation of the unreliable messages, the computation of each C2V message associated with T_2 or T_k type check node should be ignored.

As mentioned above, after generating the new C2V messages, the LLR value of the chosen v_{1j} is updated by (11). Moreover, the updated v_{1j} is considered to be reliable after receiving the reliable V2C messages and the unreliable V2C messages are hence stopped propagating. It is because the updated C2V messages with high reliability have comparatively large magnitudes to execute the domination of the LLR value of v_{1j}. The definition of the domination can be given below:

Domination: *If the LLR sign of v_{1j} is wrong, the updated C2V messages can change its LLR sign. Otherwise, the updated C2V messages can strengthen its LLR sign since the signs of the updated C2V messages are the same as the original LLR sign of v_{1j}.*

Similarly, all the corresponding V2C messages associated with v_{1j}, are propagated to the connected check nodes with high reliability, according to (10). At the same time, the types of all check nodes connected with v_{1j} change from T_k to T_{k-1}. For example, T_1 type check nodes will become T_0 type check nodes. At present, v_{1j} becomes a non-blurry variable node and its reliable V2C messages can be propagated for the updates of the rest blurry variable nodes.

In this way, each blurry variable node v_{1j} is updated one by one and removed from the blurry set, V. *Step 1* ends until V is empty or v_{1j} cannot be found. If V is empty, it indicates that all of the blurry variable nodes are processed in *Step 1* such that the MP operation has completed. Otherwise, *Step 2* will be executed subsequently. The intuitive updating process is shown in Fig. 4. The dashed lines represent unreliable messages, while the solid lines represent reliable messages.

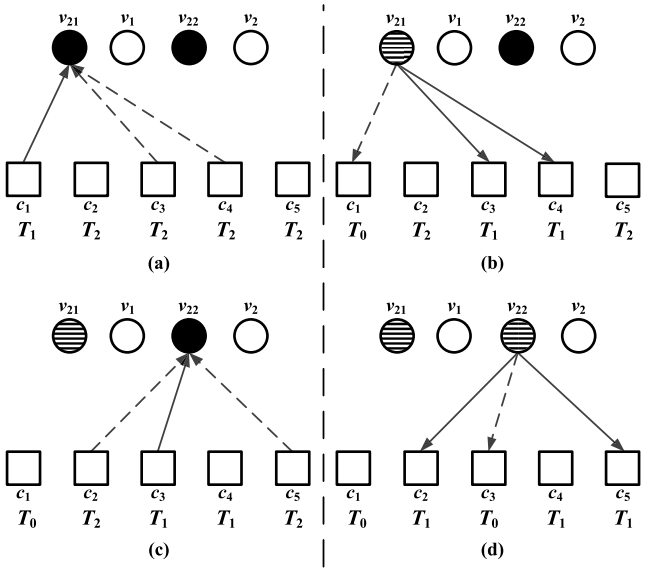


FIGURE 5. The decoding procedure at the second step in the MP strategy.

2) STEP 2:

Since v_{1j} cannot be found, the rest of variable nodes in the non-empty blurry set, V, are searched in sequence. Each variable node v_{2j}, which is connected with at least one T_1 type check node, will be updated. Generally, if the chosen variable node is connected with only one T_1 type check node, it will receive just one reliable C2V message. In addition, the computation of the unreliable C2V message associated with T_2 or T_k type check node will be ignored. Based on the domination by the reliable C2V message in the computation of (11), the updated LLR value of the chosen variable node is considered to be reliable. Furthermore, each V2C message from the chosen variable node to T_2 or T_k type check node is reliable. However, the V2C message from v_{2j} to the only T_1 type check node is still unreliable since it cannot receive any new and reliable C2V message according to (10). To limit the propagation of the unreliable messages, each updated variable node, which generates an unreliable V2C message as stated above, should be put into a set, V_1, and processed once more in *Step 3*.

If the chosen variable node is connected with at least two T_1 type check nodes, it will become a non-blurry one without propagating unreliable V2C messages, after the update. In this way, each v_{2j} is updated and removed from V, while the types of all check nodes connected with v_{2j} change from T_k to T_{k-1}. Because of the changes of the types of check nodes, all the variable nodes in the set, V, can have the chance to be updated, such as the variable node, v_{22}, as shown in Fig. 5. When *Step 2* ends, V will become empty. The intuitive updating procedure is shown in Fig. 5, where a circle with the horizontal stripes represents a variable node belonging to the set, V_1.

3) STEP 3:

It is essential to prevent the propagation of the unreliable V2C messages associated with the variable nodes, which belong to the set, V_1. In this step, each variable node, v_{3j} in the set, V_1, will be updated in a backward sequence. Otherwise, a forward updating sequence will raise a problem that, a part of C2V messages used for updating the corresponding variable nodes are unreliable. The reason why the problem happens is that, a part of V2C messages used for updating the above C2V messages, are unreliable, as mentioned in *Step 2*. The backward updating sequence can prevent the above problem from happening. After all the variable nodes in the set, V_1, are updated, and propagate the reliable V2C messages, the overall MP operations end.

After performing the above operations, all the blurry variable nodes turn into the non-blurry ones, and their generated V2C messages with high reliability can be devoted to the subsequent decoding process. Let v_{b_j} represent a type of the blurry variable nodes without connecting T_1 type check nodes. During the serial updating process, each variable node v_{b_j} will be updated with the unreliable C2V messages from T_2 or T_k type check nodes. In other words, the updated variable node v_{b_j} cannot receive any reliable C2V message from T_1 type check node, and hence the V2C messages generated at node v_{b_j} are all unreliable. In this way, it is likely that each v_{b_j} cannot be corrected. Therefore, the error-correction performance and convergence performance of the serial algorithms are restricted, since variable nodes v_{b_j} account for a considerable proportion in the blurry ones. With the MP strategy, each v_{b_j} cannot be updated until it meets the updating condition in *Step 1* or *Step 2*. Indeed, the reliable messages can be utilized effectively and propagated widely in the initial stage of decoding. The decoding process can take a step towards the right direction, without being affected by the unreliable messages from the blurry variable nodes. Compared with the conventional serial scheduling, the MP strategy can achieve faster convergence speed. The MP strategy in pseudo-code is shown in Algorithm 1 and the corresponding flow chart is shown in Fig. 6. In Algorithm 1, $M'(v_{1j})$ (or $M'(v_{2j})$) denotes the set that includes all T_1 type check nodes connected to variable node v_{1j} (or v_{2j}).

To analyze the effect of the domination of the LLR values of the blurry variable nodes, P_{C2V} is denoted as the proportion of the number of the updated C2V messages $m_{c\to v}$ where $|m_{c\to v}| < max\{G_1\}$, $N_{|C2V|<max\{G_1\}}$, to the total number of the updated C2V messages in the MP operations, N_{total}, given by

$$P_{C2V} = \frac{N_{|C2V|<max\{G_1\}}}{N_{total}} \times 100\%. \tag{13}$$

Observing the values of P_{C2V} from Table 4, at each simulated PE cycle, $N_{|C2V|<max\{G_1\}}$ keeps the extremely low proportion in N_{total}. Hence, it can demonstrate that almost all of the updated C2V messages in the MP operations have the relatively large magnitudes to dominate the LLR values

Algorithm 1 Message Pre-Processing

Perform the memory sensing strategy to obtain the region information of each variable node
Compute L_{msb}, L_{lsb} using (7), (8)
Initialize all $m_{c_m \to v_n} = 0$ and $m_{v_n \to c_m} = C_{v_n}$
Initialize the blurry set $V = \varnothing$ and the set $V_1 = \varnothing$
Put the blurry variable nodes which correspond to the LSBs in regions $E1$ and $E3$, and the MSBs in region $E2$, into the set, V
Let v_{1j} denote each blurry variable node connected with at least two T_1 type check nodes
while v_{1j} can be found from V **do**
  for every T_1 type check node $c_i \in M'(v_{1j})$ **do**
    Generate and propagate $m_{c_i \to v_{1j}}$ using (9)
  end for
  Update $L_{v_{1j}}$ using (11)
  for every check node $c_i \in M(v_{1j})$ **do**
    Generate and propagate $m_{v_{1j} \to c_i}$ using (10)
  end for
  Remove v_{1j} from V and search the next v_{1j}
end while
if V is empty **then**
  Exit
end if
Let v_{2j} denote each blurry variable node connected with at least one T_1 type check node
while v_{2j} can be found from V **do**
  if v_{2j} connects only one T_1 type check node **then**
    Put v_{2j} into the set, V_1
  end if
  for every T_1 type check node $c_i \in M'(v_{2j})$ **do**
    Generate and propagate $m_{c_i \to v_{2j}}$ using (9)
  end for
  Update $L_{v_{2j}}$ using (11)
  for every check node $c_i \in M(v_{2j})$ **do**
    Generate and propagate $m_{v_{2j} \to c_i}$ using (10)
  end for
  Remove v_{2j} from V and search the next v_{2j}
end while
Let v_{3j} denote each variable node stored in the set, V_1
for every $v_{3j} \in V_1$ **do**
  for every check node $c_i \in M(v_{3j})$ **do**
    Generate and propagate $m_{c_i \to v_{3j}}$ using (9)
  end for
  Update $L_{v_{3j}}$ using (11)
  for every check node $c_i \in M(v_{3j})$ **do**
    Generate and propagate $m_{v_{3j} \to c_i}$ using (10)
  end for
end for

of the blurry variable nodes. Moreover, since the updated C2V messages are considered to be reliable, the updated blurry ones prone to be erroneous have high probability to be corrected.

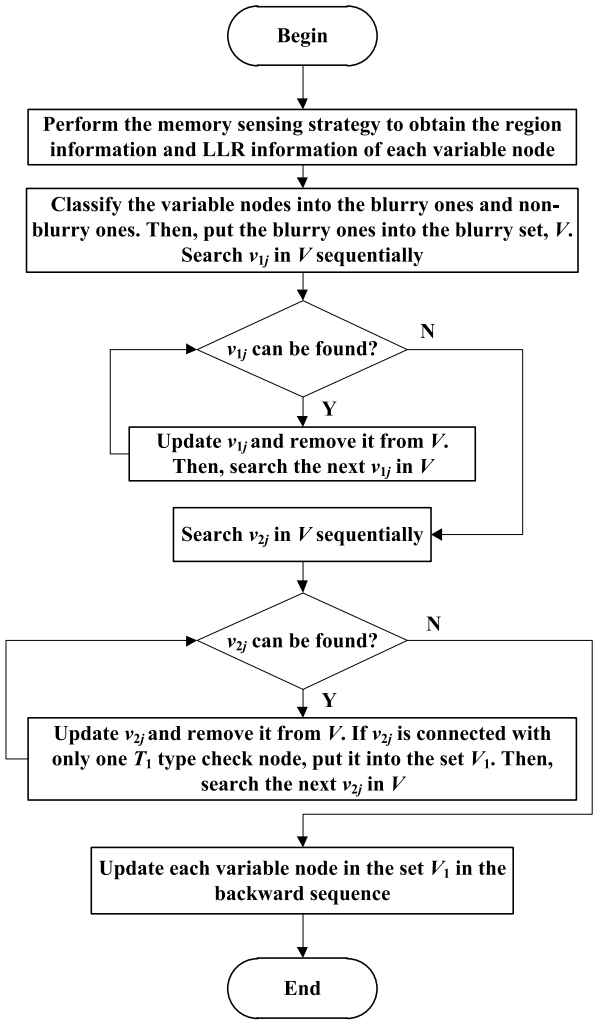


FIGURE 6. The flow chart of the MP strategy.

TABLE 4. P_{C2V} of (3780, 3402) binary LDPC Codes at different PE cycles.

PE	1.1×10^4	1.2×10^4	1.3×10^4	1.4×10^4	1.5×10^4
P_{C2V}	0.80%	0.94%	1.29%	2.01%	2.73%

With the increase of the number of iterations, a part of variable nodes, which stays in the oscillating state, is error-prone [23]. The oscillation reason is related to the structure of the parity check matrix. Hence, the oscillation phenomenon appears in nearly all the BP algorithms [21]. To further speed up the convergence, the oscillating variable nodes should be considered in subsequent iterations. The definition of the oscillating set, O, is given below.

Oscillating set O over $GF(2)$: *In the iterative decoding for binary LDPC codes, a variable node is oscillating if the sign of its LLR value in the current update is different from that in the last update. In each iteration, all variable nodes are updated first, and the oscillating ones are judged and classified as the oscillating set, O.*

As shown in Tables 1 and 2, a minority of the non-blurry variable nodes which corresponds to the bits with

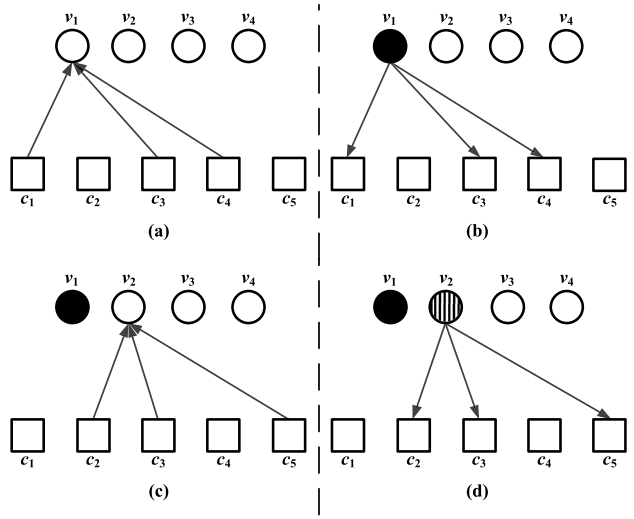


FIGURE 7. The serial decoding process for all variable nodes.

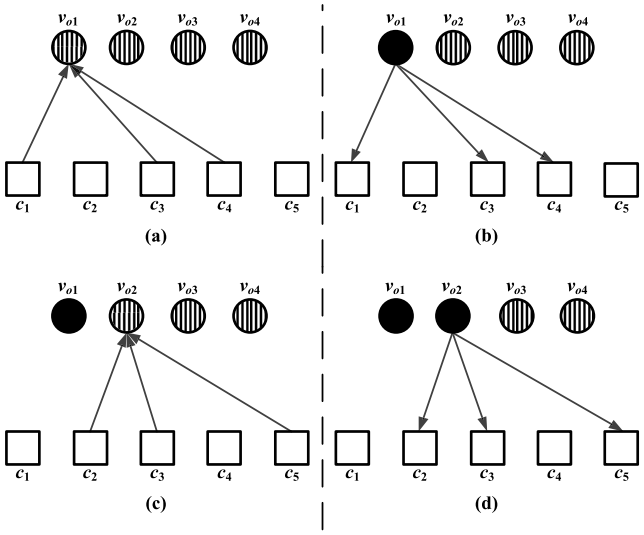


FIGURE 8. The serial decoding process for the oscillating set.

low error probability, has wrong LLR information. However, the updates of all the non-blurry variable nodes are ignored in the MP strategy, causing a problem that those erroneous non-blurry ones mentioned above have no chance to be updated. Hence, a minority of the blurry ones cannot be corrected, and the MP strategy is affected in a negative way. To solve the problem induced by the erroneous non-blurry ones and locate all the oscillating ones, an OB-ASS strategy is proposed, described as follows:

1) After the MP operations, all variable nodes are processed according to the serial scheduling.
2) The oscillation decision for each variable node is executed, and the oscillating set, O, is generated.
3) In the oscillating set, O, each oscillating variable node is updated in sequence. When the updates finish, O will be cleared.

As shown in Figs. 7 and 8, the above OB-ASS strategy can be illustrated with a Tanner graph. The black circle represents an updated variable node, and the circle with the vertical stripes represents an oscillating variable node. In Fig. 7, all the C2V messages associated with v_1, are updated with (9). Then, L_{v_1} is updated by the new C2V messages, and the oscillation decision for v_1 is made. In the end, the corresponding V2C messages associated with v_1 are generated with (10). In the same way, v_2 is updated. It can be observed that v_2 is an oscillating variable node. Hence, it will be put into the oscillating set, O. In this way, all the variable nodes are updated in sequence and the entire oscillating set, O, is formed. Subsequently, the above scheduling strategy is also executed for the oscillating set, O, as shown in Fig. 8. After the treatment for the oscillating variable nodes, the oscillating set, O, should be cleared for the preparation of the next round of the alternate selection decoding.

The (10, 5) LDPC Tanner graph is sketched to intuitively illustrate the detailed updating process of the proposed algorithm, as shown in Figs. 9 and 10. In Fig. 9, the black circles, v_2, v_4 and v_6, are the blurry variable nodes while the white circles are the non-blurry ones. The circle with horizontal stripes represent the variable node stored in V_1. The dashed lines with arrows represent the unreliable messages, which are ignored in the computation. The solid lines with arrows represent the messages considered to be reliable. Firstly, *Step 1* of the MP strategy is performed, and v_4 meeting the updating condition is chosen to be processed. After the process of updating v_4, *Step 1* ends since v_2 and v_4 do not meet the updating condition. Therefore, in *Step 2*, the process of successively updating v_2 and v_4 is started. Since v_2 connects only one T_1 type check node, it is put into V_1 to prevent the propagation of the unreliable V2C message $m_{v_2 \to c_1}$. Finally, v_2 is processed again in *Step 3* and the MP strategy has completed.

After finishing the MP scheme, the OB-ASS strategy is performed as shown in Fig. 10. Contrasting to the case of Fig. 9, the black circles in Fig. 10 represent the updated variable nodes, the white circles denote the ones without being updated, and the circles with vertical stripes indicate the oscillating ones. During the first phase, the variable nodes from v_0 to v_9 are processed successively. At the same time, the oscillating decisions are executed and the corresponding variable nodes with oscillations, v_0, v_2 and v_3 are put into the set, O. Hence, during the second phase, the variable nodes in O are updated in sequence. Finally, the OB-ASS strategy ends and hard decisions are performed.

Assuming the maximum number of iterations is I_{max}. The OB-ASS strategy will be executed repeatedly, until the LDPC code of block-length N is successfully decoded, or the number of updated variable nodes reaches $I_{max} \times N$. The proposed VNBP-MP algorithm in pseudo-code is described in Algorithm 2.

B. NVNBP-MP DECODING ALGORITHM FOR NON-BINARY LDPC CODES

In the simulation of the MLC NAND flash channel model, we exploit non-binary LDPC codes [24] con-

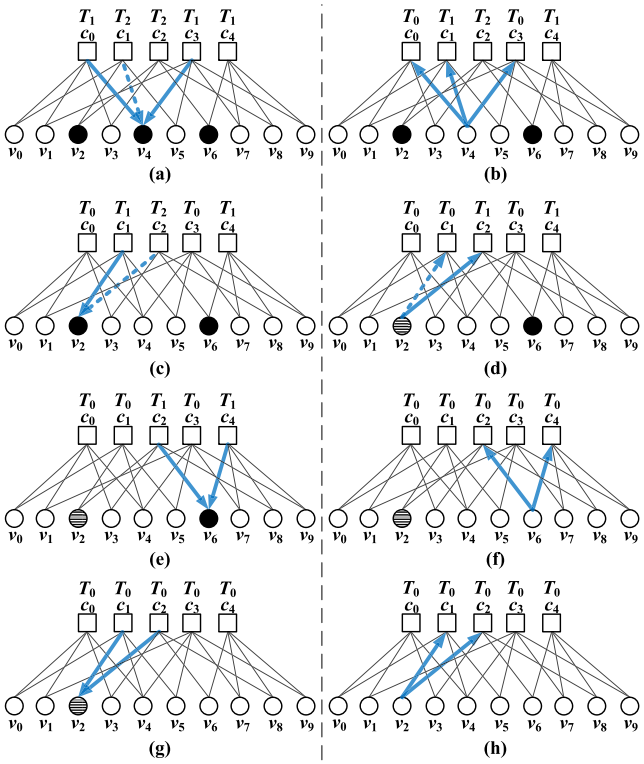


FIGURE 9. The updating process of the MP strategy with (10, 5) LDPC code as an instance.

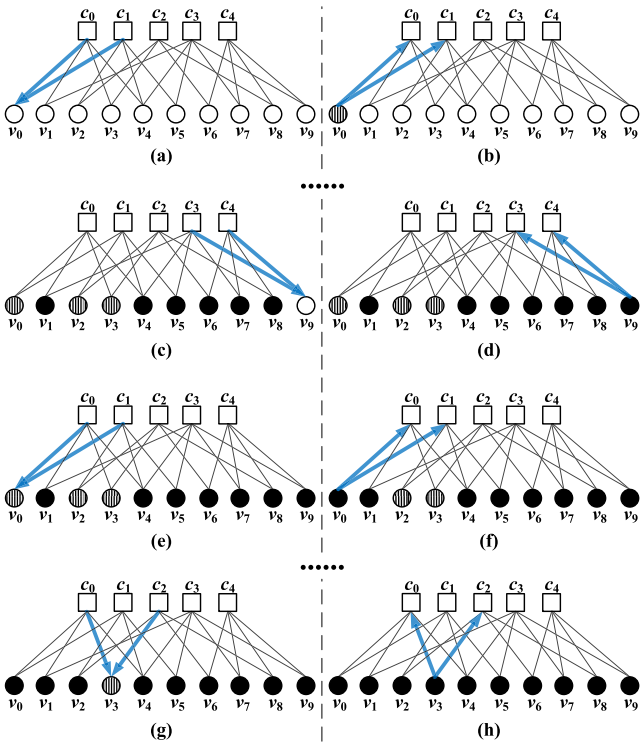


FIGURE 10. The updating process of the OB-ASS strategy with (10, 5) LDPC code as an instance.

structed over $GF(4)$. Each quaternary symbol represents the value '0', '1', '2' or '3'. To store a quaternary symbol into an MLC NAND flash memory storage cell,

Algorithm 2 VNBP-MP

1: Execute the message pre-processing operation
2: Initialize the oscillating set $O = \varnothing$
3: **for** every variable node v_j **do**
4: **for** every check node $c_i \in M(v_j)$ **do**
5: Generate and propagate $m_{c_i \to v_j}$ using (9)
6: **end for**
7: Update L_{v_j} using (11)
8: **if** v_j oscillates **then**
9: Put v_j into the set, O
10: **end if**
11: **for** every check node $c_i \in M(v_j)$ **do**
12: Generate and propagate $m_{v_j \to c_i}$ using (10)
13: **end for**
14: **end for**
15: Let v_{oj} denote each oscillating variable node stored in the set, O
16: **for** every $v_{oj} \in O$, **do**
17: **for** every check node $c_i \in M(v_{oj})$ **do**
18: Generate and propagate $m_{c_i \to v_{oj}}$ using (9)
19: **end for**
20: Update $L_{v_{oj}}$ using (11)
21: **for** every check node $c_i \in M(v_{oj})$ **do**
22: Generate and propagate $m_{v_{oj} \to c_i}$ using (10)
23: **end for**
24: **end for**
25: **if** stopping rule is not satisfied **then**
26: Go to line 2
27: **end if**

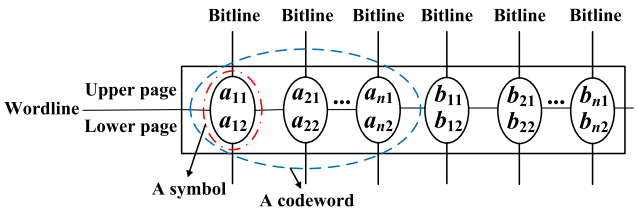


FIGURE 11. Arrangement scheme for quaternary symbols in the MLC NAND flash memory cells.

the quaternary symbol should be transformed into two bits.

In the conventional arrangement scheme, each codeword is stored over the same page, such that two bits stored in a flash memory cell belong to different codewords. To make use of the channel characteristic, the MP strategy should be applied to the arrangement scheme, where two bits from the same quaternary symbol are stored in the same cell [9], as shown in Fig. 11.

In the perspective of the Tanner graph, each variable node represents one quaternary symbol. The variable nodes can also be classified into the blurry ones and non-blurry ones. The classification only depends on the region where a specified cell lies. As stated before, the definition of the blurry variable node over $GF(4)$ is different from that over $GF(2)$.

If a specified cell falls in the data region, the corresponding variable node is denoted as the non-blurry one. On the other hand, if a specified cell falls in the overlapping region, the corresponding variable node is denoted as the blurry one. Besides, the computation of the soft information over $GF(q)(q > 2)$ is performed in the probability domain. To perform the OB-ASS strategy, the definition of the oscillating set, O, over $GF(q)$ is modified as follows

Oscillating set O over $GF(q)$: *In the iterative decoding for non-binary LDPC codes, a variable node is oscillating if the decoded symbol in the current update is different from that in the last update. In each iteration, all variable nodes are updated first, and the oscillating variable ones are judged and classified as the oscillating set, O.*

According to the modification of the definitions mentioned above, the NVNBP-MP decoding algorithm is proposed for non-binary LDPC codes. In terms of the VNBP-MP algorithm and the NVNBP-MP algorithm, they have the same way in scheduling between variable nodes and check nodes.

IV. SIMULATION RESULTS

In order to analyze the error-correction performance, convergence speed and decoding latency of the proposed algorithms, a large number of simulations are performed over the MLC NAND flash channel. Over $GF(2)$, LLRBP [10], SBP [11] and QABP [17] algorithms are performed for comparison with the proposed VNBP-MP algorithm. In addition, FFT-BP [25] and FFT-SBP [26] algorithms are used for comparison with the NVNBP-MP algorithm over $GF(4)$. The above two algorithms, which exploit fast fourier transform to reduce the computation complexity, are non-binary versions of the LLRBP and SBP algorithms.

The LDPC codes are denoted with $(N, N - M)$, where N represents the block length and $(N - M)$ represents the number of information symbols. The regular LDPC codes employed for simulations are constructed with the progressive edge-growth (PEG) algorithm [27]. The irregular LDPC codes are constructed with the algebraic approach and masking technique [28]. Furthermore, to fulfill the requirement of the large storage capacity in the MLC NAND flash memory, the LDPC codes with high code rate and long block length, should be applied to the simulated channel model. For example, in the binary version, the irregular LDPC code (4032, 3264), and the regular LDPC codes, (3780, 3402), (8000, 7360), and (12000,10800), are explored. In the non-binary version, regular (2304, 2048) LDPC codes are used. In this paper, we focus on the effect of PE cycles on the performance of the decoding algorithms. Therefore, in the analysis of the frame error rate (FER) performance, the value of retention time, T, is set to 1 to mitigate the effort of retention noise as much as possible, and the maximum number of iterations, I_{max}, is set to 5.

Figures 12, 13 and 14 show the FER performance of the decoding algorithms for (3780, 3402), (8000, 7360) and (12000, 10800) binary LDPC codes. It is clearly shown that the proposed algorithm obtains the best FER performance

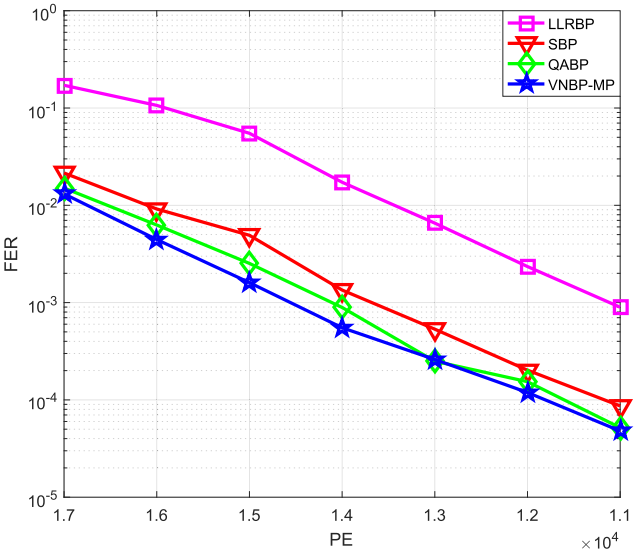


FIGURE 12. FER vs. PE cycles performance of LLRBP, SBP, QABP, and VNBP-MP algorithms for (3780, 3402) binary LDPC codes at $I_{max} = 5$ in the MLC NAND flash memory channel ($T = 1$).

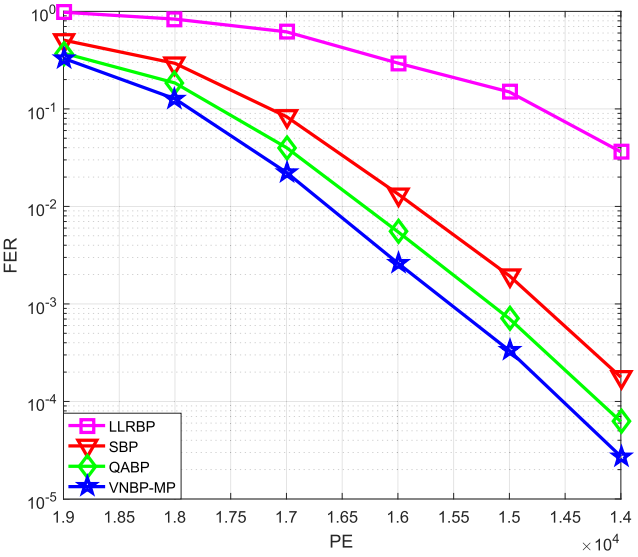


FIGURE 13. FER vs. PE cycles performance of LLRBP, SBP, QABP, and VNBP-MP algorithms for (8000, 7360) binary LDPC codes at $I_{max} = 5$ in the MLC NAND flash memory channel ($T = 1$).

among the 4 decoding algorithms within the range of the simulated PE cycles. The main reason is that the MP strategy exploits reliable messages to update unreliable blurry variable nodes and forbids the propagation of unreliable messages. Thus, the MP strategy can furthest reduce the number of decoded bit errors induced by the overlapping regions for fast convergence. Furthermore, the oscillating variable nodes can be processed in time, with the aid of the OB-ASS strategy. From Figs. 12, 13 and 14, we can observe that the performance gain of the proposed algorithm becomes increasingly outstanding with the increase of the block-length of LDPC codes. The similar FER performance for (4032, 3264) LDPC

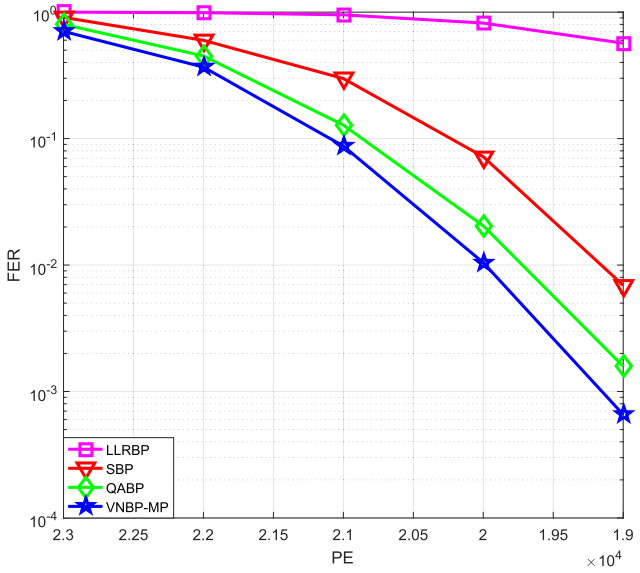


FIGURE 14. FER vs. PE cycles performance of LLRBP, SBP, QABP, and VNBP-MP algorithms for (12000, 10800) binary LDPC codes at $I_{max} = 5$ in the MLC NAND flash memory channel ($T = 1$).

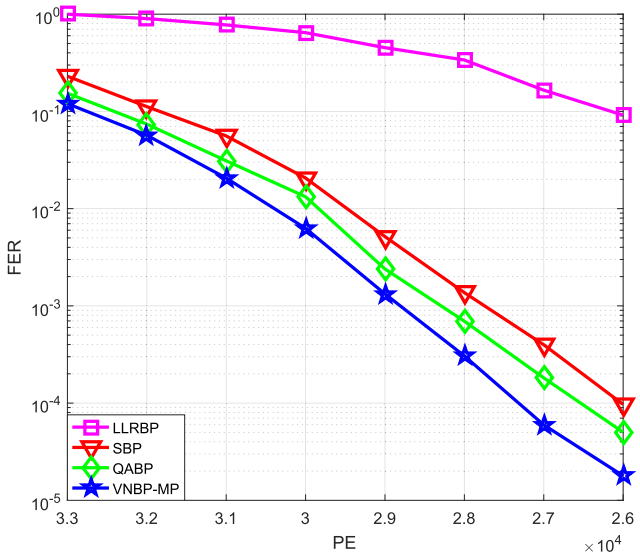


FIGURE 15. FER vs. PE cycles performance of LLRBP, SBP, QABP, and VNBP-MP algorithms for (4032, 3264) binary LDPC codes at $I_{max} = 5$ in the MLC NAND flash memory channel ($T = 1$).

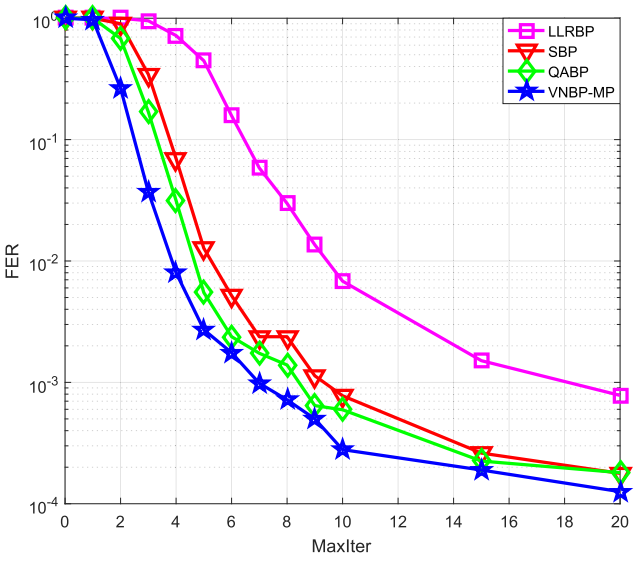


FIGURE 16. FER vs. the maximum number of iterations performance of LLRBP, SBP, QABP, and VNBP-MP algorithms for (8000, 7360) binary LDPC codes at $PE = 16000$ in the MLC NAND flash memory channel ($T = 1$).

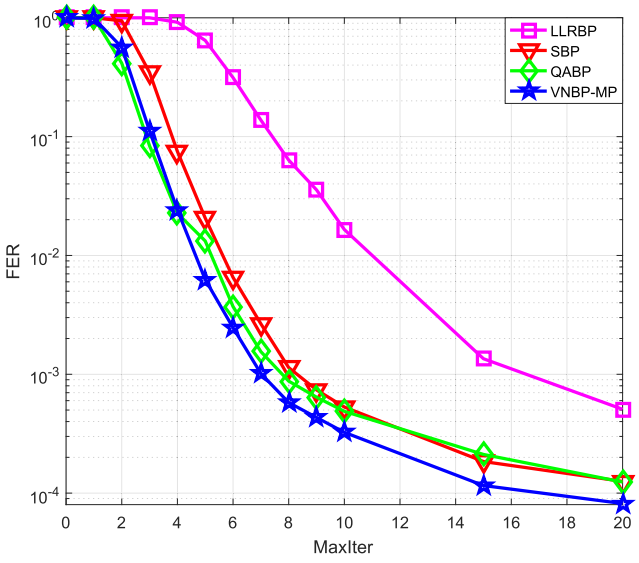


FIGURE 17. FER vs. the maximum number of iterations performance of LLRBP, SBP, QABP, and VNBP-MP algorithms for (4032, 3264) binary LDPC codes at $PE = 30000$ in the MLC NAND flash memory channel ($T = 1$).

codes are shown in Fig. 15. Compared to the SBP algorithm, when FER is 1.0×10^{-4}, the proposed algorithm earns about 1400 PE cycles. In Figs. 12, 13, 14 and 15, the LLRBP algorithm suffers from the worst FER performance, since the latest available information cannot be utilized at the same iteration.

To investigate the convergence speed of the algorithms, the FER vs. the number of iterations is simulated. The performance of the FER vs. the number of iterations for (8000,7360) LDPC codes at $PE = 16000$ is presented in Fig. 16, where the maximum number of iterations, I_{max}, varies from 1 to 20. The similar simulations are also performed for (4032, 3264) LDPC codes at $PE = 30000$, as shown in Fig. 17. In Figs. 16 and 17, the QABP algorithm achieves faster convergence speed compared to the SBP algorithm, since a part of variable nodes are skipped during the initial stage of iterations. However, as the above two algorithms converge, the FER of the QABP algorithm is nearly the same as that of the SBP algorithm. In addition, it can be demonstrated that, for different LDPC codes constructed with different matrices, the proposed algorithm always converges faster than any other decoding algorithms within 20 iterations.

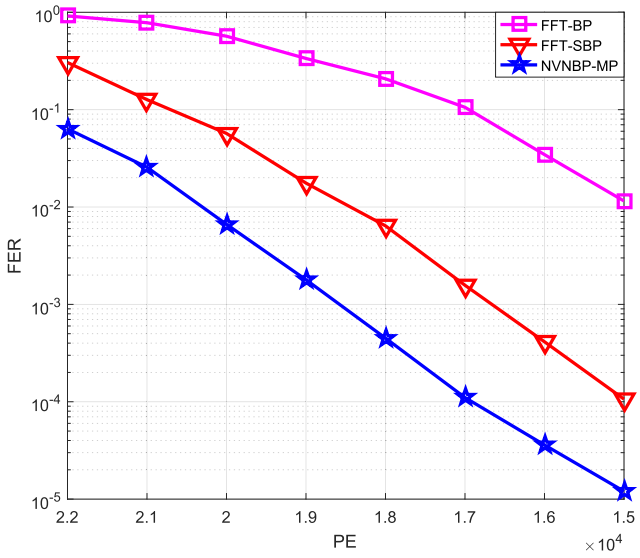


FIGURE 18. FER vs. PE cycles performance of FFT-BP, FFT-SBP, and NVNBP-MP algorithms for (2304, 2048) quaternary LDPC codes at $I_{max} = 5$ in the MLC NAND flash memory channel ($T = 1$).

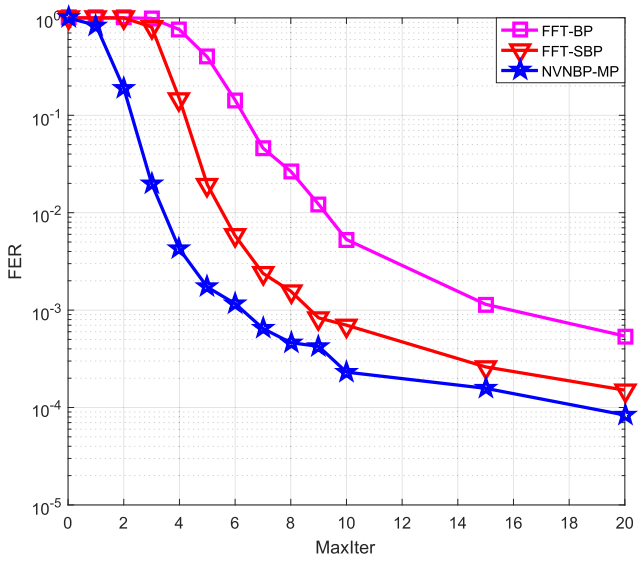


FIGURE 19. FER vs. the maximum number of iterations of FFT-BP, FFT-SBP, and NVNBP-MP algorithms for (2304, 2048) quaternary LDPC codes at $PE = 19000$ in the MLC NAND flash memory channel ($T = 1$).

Figure 18 shows the FER performance among FFT-BP, FFT-SBP and proposed NVNBP-MP algorithms for (2304, 2048) quaternary LDPC codes. We can notice that the proposed algorithm has an outstanding improvement in the error correction performance compared with any other decoding algorithms. When FER is 1.0×10^{-4}, the proposed algorithm can get 2000 PE cycles gain, compared with the FFT-SBP algorithm. It can be demonstrated that the proposed algorithm can adapt to worse channel condition than the conventional FFT-SBP algorithm, and prolong the lifetime of the MLC NAND flash memory.

Figure 19 shows the FER performance vs. the maximum number of iterations for (2304, 2048) quaternary LDPC codes at $PE = 19000$. The maximum number of iterations, I_{max}, varies from 1 to 20 to compare the convergence speed among the three decoding algorithms. Since the FFT-BP algorithm is based on the flooding scheduling, it has the slowest convergence speed. Furthermore, it is clearly shown that the proposed NVNBP-MP algorithm obtains the fastest convergence speed within 20 iterations, as a result of the MP strategy and the treatment for the oscillating set. In other words, the proposed NVNBP-MP algorithm has better error-correction performance with fewer iterations. After getting convergent, the error-correction performance of the proposed algorithm is better than that of the FFT-BP and FFT-SBP algorithms.

In the proposed schemes, there is no additional computation complexity in updating each variable node. In the MP strategy, to reorder the updating sequence of the blurry variable nodes, the additional complexity of comparison operations is needed. As long as each blurry variable node is scanned, $\overline{d_{bv}}$ times of comparison operations are performed to check the types of the corresponding check nodes, where $\overline{d_{bv}}$ denotes the average degree of the blurry ones. This operation is performed to determine whether the blurry one meets the updating condition or not. For each frame, the average number of the blurry variable nodes is denoted as $\overline{N_{bv}}$. The factor α is employed to intuitively show the number of the scans for the blurry variable nodes during the MP strategy operation, represented by $\alpha \cdot \overline{N_{bv}}$. In other words, to reorder the sequence of each frame, the MP strategy totally needs $\alpha \cdot \overline{N_{bv}} \cdot \overline{d_{bv}}$ times of comparisons. Table 5 shows the simulated value of the factor α with the varied PE cycles when different LDPC codes are used. At each PE cycle and each type of LDPC codes, the number of frames in simulation is set to 50000.

In the table, the value of α is small, especially in the case of (4032, 3264) LDPC code. For each frame, $\overline{N_{bv}}$ is less than 10% of the block length N. Furthermore, since the NAND flash utilizes LDPC codes with high code rate, the average degree of variable nodes, $\overline{d_v}$, is much less than that of check nodes, $\overline{d_c}$. For example, considering the regular (3780, 3402) LDPC code, $\overline{d_{bv}} = \overline{d_v} = 3$ and $\overline{d_c} = 30$. When the number of PE cycles is 1.3×10^4, the factor α is 2.07. Therefore, $\alpha \cdot \overline{N_{bv}} \cdot \overline{d_{bv}} = 6.21\overline{N_{bv}} < 0.621N$ times of comparison operations are required to reorder the updating sequence. Besides, there are N times of additional comparisons for the oscillating decisions. Compared to the IDS-based algorithms, the additional complexity in computation for the proposed algorithms is significantly smaller, because the IDS-based algorithm adds $N(N-1)$ times of comparison operations for the serial BP algorithms per iteration [21]. Thus, the proposed algorithms slightly increase the complexity of the serial BP algorithms.

However, with the fast convergence speed of the proposed algorithms, the average number of iterations can be obviously reduced. Hence, the overall decoding latency is lower, compared to the serial BP algorithms. To make a fair comparison in decoding latency, the throughput is defined to represent the number of frames successfully decoded per unit time [29]

TABLE 5. The factor α of different LDPC codes at varied PE cycles.

PE	1.1×10^4	1.2×10^4	1.3×10^4	1.4×10^4	1.5×10^4	1.6×10^4	1.7×10^4
α of (3780, 3402) LDPC codes	1.66	1.96	2.07	2.51	2.64	2.92	3.16
PE	1.2×10^4	1.3×10^4	1.4×10^4	1.5×10^4	1.6×10^4	1.7×10^4	1.8×10^4
α of (8000, 7360) LDPC codes	1.65	1.98	2.52	3.52	4.17	4.58	4.61
PE	2.6×10^4	2.7×10^4	2.8×10^4	2.9×10^4	3.0×10^4	3.1×10^4	3.2×10^4
α of (4032, 3264) LDPC codes	1.29	1.33	1.37	1.43	1.48	1.54	1.62

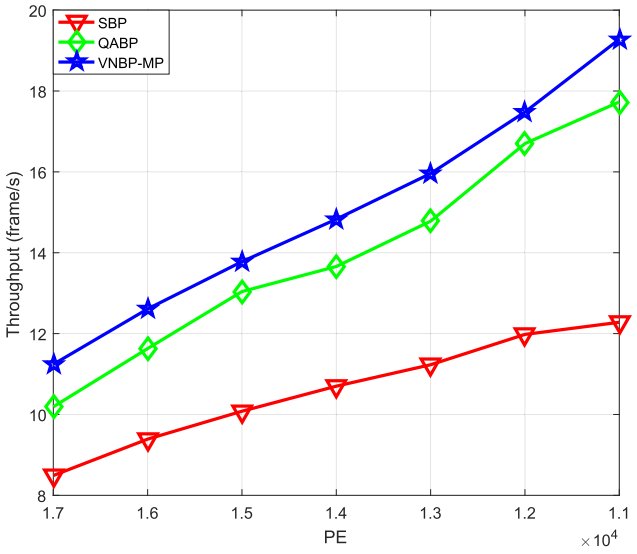


FIGURE 20. Throughput vs. PE cycles performance of SBP, QABP, and VNBP-MP algorithms for (3780, 3402) binary LDPC codes at $I_{max} = 5$ in the MLC NAND flash memory channel ($T = 1$).

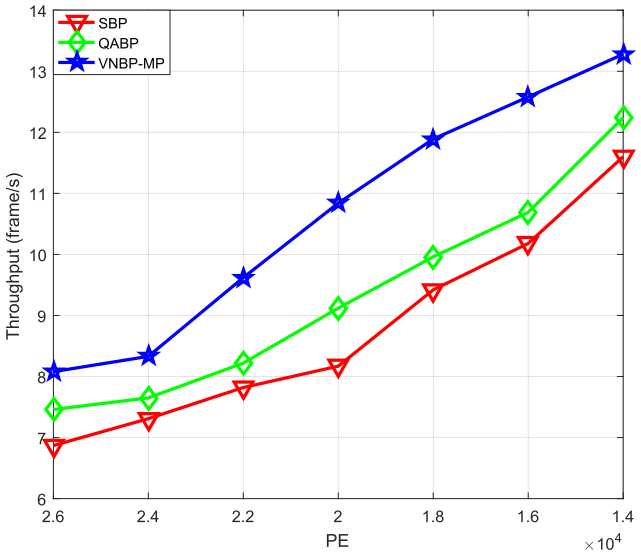


FIGURE 21. Throughput vs. PE cycles performance of SBP, QABP, and VNBP-MP algorithms for (4032, 3264) binary LDPC codes at $I_{max} = 5$ in the MLC NAND flash memory channel ($T = 1$).

at the fixed PE cycles. All simulations are performed over the MLC NAND flash channel model with Visual Studio 2015 programming, and the maximum number of iterations is up to 5. In terms of the decoding latency, the simulation results of the throughput are presented in Figs. 20 and 21. From the figures, it is clearly shown that, as the number of PE cycles increases, the throughput of all simulated algorithms decreases. Although the complexity of proposed algorithm is slightly higher than SBP algorithm and QABP algorithm, the results of Figs. 20 and 21 can totally prove that the efficiency of decoding of proposed algorithm is better than the state-of-the-art algorithms in these cases. However, the throughput performance of the proposed algorithms surpasses that of the existing algorithms.

In the proposed work, extensive simulations over the MLC NAND flash channel model also demonstrate that the proposed algorithms perform well. The channel model established in this paper slightly deviates from the real NAND flash behavior. Nevertheless, the simulation channel is established by reference to the real NAND flash channel [18]. In order to obtain the characteristics of the channel, a large set of random data, generated with the uniform distribution of '0' and '1', is employed in the simulated model. At first, the generated data were not encoded and decoded. Thus, the error probability of each region can be calculated, by which the accuracy of each cell can be determined. There always exist overlapping regions between adjacent states in the actual channel. As the overlap becomes obvious, which is induced by the channel deterioration, more and more blurry variable nodes can be processed effectively by determining the value of read reference voltages. Thus, the proposed algorithms can work in the actual NAND flash on this occasion. Furthermore, the proposed algorithms can be easily extended for TLC and QLC cases.

V. CONCLUSION

In this paper, the MP strategy, which makes use of the characteristic of the MLC NAND flash channel where the blurry variable nodes have high error probability, is introduced. In the MP strategy, the propagation of unreliable messages associated with the blurry variable nodes is forbidden, while reliable messages associated with the non-blurry ones can be preferentially propagated to update the blurry ones. Hence, in the initial stage of decoding, the decoding process can be performed towards the right direction. Besides, the proposed OB-ASS strategy can process the oscillating variable

nodes, after the MP operations. Based on the above MP and OB-ASS strategies, two decoding algorithms, the VNBP-MP over $GF(2)$ and the NVNBP-MP over $GF(q)$, are proposed for better error-correction performance, faster convergence speed and lower decoding latency. Simulation results demonstrate that, the VNBP-MP algorithm for binary LDPC codes and the NVNBP-MP algorithm for quaternary LDPC codes outperform the other decoding algorithms in terms of the FER performance. Furthermore, both of the proposed algorithms have faster convergence speed and larger throughput than any other decoding algorithms.

REFERENCES

[1] Y. Pan, G. Dong, Q. Wu, and T. Zhang, "Quasi-nonvolatile SSD: Trading flash memory nonvolatility to improve storage system performance for enterprise applications," in *Proc. IEEE Int. Symp. High-Perform. Comp Archit. (HPCA)*, New Orleans, LA, USA, Feb. 2012, pp. 1–10.

[2] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling," in *Proc. Conf. Design, Automat. Test Eur. (DATE)*, Grenoble, France, Mar. 2013, pp. 1285–1290.

[3] Y. Luo, S. Ghose, Y. Cai, E. F. Haratsch, and O. Mutlu, "Enabling accurate and practical online flash channel modeling for modern MLC NAND flash memory," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2294–2311, Sep. 2016.

[4] L. Zuolo, C. Zambelli, A. Marelli, R. Micheloni, and P. Olivo, "LDPC soft decoding with improved performance in 1X-2X MLC and TLC NAND flash-based solid state drives," *IEEE Trans. Emerg. Topics Comput.*, to be published.

[5] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis," in *Proc. Conf. Design, Automat. Test Eur. (DATE)*, Dresden, Germany, Mar. 2012, pp. 521–526.

[6] T.-Y. Chen, A. R. Williamson, and R. D. Wesel, "Increasing flash memory lifetime by dynamic voltage allocation for constant mutual information," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, Feb. 2014, pp. 1–5.

[7] G. Dong, N. Xie, and T. Zhang, "Enabling NAND flash memory use soft-decision error correction codes at minimal read latency overhead," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 9, pp. 2412–2421, Sep. 2013.

[8] K. Zhao, W. Zhao, H. Sun, T. Zhang, and X. Zhang, "LDPC-in-SSD: Making advanced error correction codes work effectively in solid state drives," in *Proc. USENIX Conf. File Storage Technol. (FAST)*, San Jose, CA, USA, Feb. 2013, pp. 243–256.

[9] M. Zhang, F. Wu, X. He, P. Huang, S. Wang, and C. Xie, "REAL: A retention error aware LDPC decoding scheme to improve NAND flash read performance," in *Proc. 32nd Symp. Mass Storage Syst. Technol. (MSST)*, Santa Clara, CA, USA, May 2016, pp. 1–13.

[10] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[11] J. Zhang and M. P. C. Fossorier, "Shuffled iterative decoding," *IEEE Trans. Commun.*, vol. 53, no. 2, pp. 209–213, Feb. 2005.

[12] D. E. Hocevar, "A reduced complexity decoder architecture via layered decoding of LDPC codes," in *Proc. IEEE Workshop Signal Process. Syst. (SIPS)*, Austin, TX, USA, Oct. 2004, pp. 107–112.

[13] A. I. V. Casado, M. Griot, and R. D. Wesel, "Informed dynamic scheduling for belief-propagation decoding of LDPC codes," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Glasgow, U.K., Jun. 2007, pp. 932–937.

[14] X. Liu, C. Fan, and X. Chen, "Dynamic scheduling decoding of LDPC codes based on tabu search," *IEEE Trans. Commun.*, vol. 65, no. 11, pp. 4612–4621, Nov. 2017.

[15] C. A. Aslam, Y. L. Guan, and K. Cai, "Retention-aware belief-propagation decoding for NAND flash memory," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, vol. 64, no. 6, pp. 725–729, Jun. 2017.

[16] H. Sun, W. Zhao, M. Lv, G. Dong, N. Zheng, and T. Zhang, "Exploiting intracell bit-error characteristics to improve min-sum LDPC decoding for MLC NAND flash-based storage in mobile device," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 24, no. 8, pp. 2654–2664, Aug. 2016.

[17] C. A. Aslam, Y. L. Guan, and K. Cai, "Low-complexity quantization-aware belief-propagation (QA-BP) decoding for MLC NAND flash memory," in *Proc. 10th Int. Conf. Inf., Commun. Signal Process. (ICICS)*, Singapore, Dec. 2015, pp. 1–5.

[18] C. A. Aslam, Y. L. Guan, and K. Cai, "Read and write voltage signal optimization for multi-level-cell (MLC) NAND flash memory," *IEEE Trans. Commun.*, vol. 64, no. 4, pp. 1613–1623, Apr. 2016.

[19] G. Dong, N. Xie, and T. Zhang, "On the use of soft-decision error-correction codes in NAND flash memory," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 2, pp. 429–439, Feb. 2011.

[20] S. Qi, D. Feng, and J. Liu, "Optimal voltage signal sensing of NAND flash memmory for LDPC code," in *Proc. IEEE Workshop Signal Process. Syst. (SiPS)*, Belfast, U.K., Oct. 2014, pp. 1–6.

[21] X. Liu, Y. Zhang, and R. Cui, "Variable-node-based dynamic scheduling strategy for belief-propagation decoding of LDPC codes," *IEEE Commun. Lett.*, vol. 19, no. 2, pp. 147–150, Feb. 2015.

[22] C. A. Aslam, Y. L. Guan, K. Cai, and G. Han, "Informed fixed scheduling for faster convergence of shuffled belief-propagation decoding," *IEEE Commun. Lett.*, vol. 21, no. 1, pp. 32–35, Jan. 2017.

[23] J. Lim and D.-J. Shin, "UIBF decoding to lower the error floors of high-rate systematic LDPC codes," *Electron. Lett.*, vol. 53, no. 4, pp. 247–249, Feb. 2017.

[24] M. C. Davey and D. MacKay, "Low-density parity check codes over GF(q)," *IEEE Commun. Lett.*, vol. 2, no. 6, pp. 165–167, Jun. 1998.

[25] L. Barnault and D. Declercq, "Fast decoding algorithm for LDPC over GF(2^q)," in *Proc. IEEE Inf. Theory Workshop*, Paris, France, Mar./Apr. 2003, pp. 70–73.

[26] N. Yacov, H. Efraim, H. Kfir, I. Kanter, and O. Shental, "Parallel vs. sequential belief propagation decoding of LDPC codes over GF(q) and Markov sources," *Phys. A, Stat. Mech. Appl.*, vol. 378, no. 2, pp. 329–335, 2007.

[27] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 386–398, Jan. 2005.

[28] G. Han, Y. L. Guan, and L. Kong, "Construction of irregular QC-LDPC codes via masking with ACE optimization," *IEEE Commun. Letters*, vol. 18, no. 2, pp. 348–351, Feb. 2014.

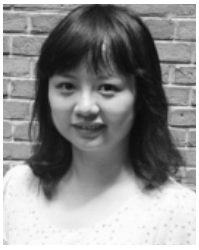
[29] X. Liu, Z. Zhou, R. Cui, and E. Liu, "Informed decoding algorithms of LDPC codes based on dynamic selection strategy," *IEEE Trans. Commun.*, vol. 64, no. 4, pp. 1357–1366, Apr. 2016.

XINGCHENG LIU (SM'12) was born in Anfu, Jiangxi, China. He received the B.E. and M.E. degrees in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, and the Ph.D. degree from Sun Yat-sen University (SYSU), Guangzhou, China.

He received the Royal Society KC Wong Fellowship of the U.K. for his postdoctoral research at the University of Southampton, Southampton, U.K., from 2002 to 2003. He was a Visiting Scientist at Oregon State University, Corvallis, OR, USA, from 2004 to 2005. He is currently a Full Professor with the School of Electronics and Information Technology, SYSU. He is also with the Xinhua College of SYSU, as a Leader in the field of electrical engineering and automation. He is currently the primary investigator of several projects on wireless communications and networking. He has authored and coauthored over 100 peer-reviewed papers in journals and conferences. His main research interests include channel coding theory and applications, mobile communications, wireless sensor networks, and the Internet of Things.

Dr. Liu is a Senior Member of the China Institute of Communications.

GUOJUN YANG was born in Jiangmen, Guangdong, China. He received the B.E. degree in information engineering from South China Normal University, Guangzhou, China, in 2016, and the M.E. degree in electronics and communication engineering from Sun Yat-sen University, Guangzhou, China, in 2018. He is currently a Research Associate with the Research Group of Networking and Communications, School of Electronics and Information Technology, Sun Yat-sen University. His main research interests include channel coding theory and mobile communications.



XUECHEN CHEN received the B.S.E. degree in electrical engineering from the University of Science and Technology of China, Hefei, China, in 2007, and the Ph.D. degree in electrical engineering from the University of California at Riverside, in 2012. She has been with the Bell Labs Research Center as a Research Scientist focusing on multimedia transmission over next generation wireless networks. She is currently an Assistant Professor of the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China. Her research interests include source-channel coding and distributed source coding especially in delay-constrained applications and distributed compressed sensing localization by wireless sensor networks.