

SSALDPC: A Syndrome-Sum Based Adaptive LDPC Decoding Scheme for NAND Flash Memory

Lanlan Cui^{†‡}, Fei Wu[§], Yunlong He^{¶*}, Kun Jiang[†], Yeqiu Xiao[†], Renzhi Xiao^{||} and Changsheng Xie[§]

[†]School of Computer Science and Engineering, XI'AN University of Technology,

Shaanxi Key Laboratory for Network Computing and Security Technology, Xi'an, China

[‡]China-ASEAN Information Harbor Co., Ltd, Nanning, China

[§] Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan, China

[¶] Gnnextech (Shanghai) Intelligent Information Technology Co., Ltd, Shanghai, China

^{||} School of Software Engineering, Jiangxi University of Science and Technology, Nanchang, China

*Corresponding author: Yunlong He, heyl@gnnextech.cn

{cuilanlan, jk_365, xiaoyeqiu}@xaut.edu.cn, {wufei, cs_xie}@hust.edu.cn, heyl@gnnextech.cn, rzxiao@jxust.edu.cn

Abstract—The continuous increase in storage density for 3D NAND flash memory, driven by multi-layer stacking and multi-level cell technology, leads to a significant overlap and shift in the threshold voltage distributions. This phenomenon significantly elevates the raw bit error rate (RBER) and poses serious challenges to data reliability. Although solutions based on low-density parity-check (LDPC) codes and read-retry schemes have become the standard approach to mitigate high RBER, the latency introduced by repeated read operations considerably degrades system read performance. This paper proposes a syndrome-sum based adaptive LDPC decoding scheme, named SSALDPC. After an initial hard decision decoding failure, our scheme utilizes the real-time syndrome sum (SS)—generated during the decoding process—to assess the severity of errors. Based on this assessment, it adaptively selects the most appropriate subsequent decoding strategy from three modes: Efficiency-Mode (E-Mode), Balance-Mode (B-Mode), or Performance-Mode (P-Mode). Experimental results demonstrate that the proposed SSALDPC scheme reduces the number of read-retry operations and decreases decoding latency under various RBER conditions, while maintaining high error correction capability.

Index Terms—NAND Flash Memory, LDPC, Adaptive Decoding, Read Latency

I. INTRODUCTION

In pursuit of higher storage density, the number of layers in 3D NAND flash memory is continuously increased through vertical stacking technology, and multi-level cell technologies are widely adopted [1]. However, this technological evolution exacerbates cell-to-cell interference, wear-out from program/erase (P/E) cycles, and charge leakage from data retention, leading to both a shift and an overlap in the threshold voltage distribution [2]. As a result, the Raw Bit Error Rate (RBER) increases significantly, posing serious challenges to data reliability.

To address high RBER, Low-Density Parity-Check (LDPC) codes coupled with a read-retry mechanism have become the standard solution [3], [4]. The read-retry process typically begins with hard decision decoding. If decoding fails, multiple re-reads are performed by adjusting the read reference voltage to generate log-likelihood ratio (LLR) information for soft

decision decoding. However, these multiple read operations introduce significant latency, severely degrading the average read performance of flash storage systems, especially in high-RBER scenarios [5].

Recent research aims to optimize this process to reduce latency. One direction focuses on read voltage and retry strategies. For example, PEAR [6] adaptively adjusts read voltages by analyzing inter-page error asymmetry, reducing RBER to a level correctable by hard decision decoding and avoiding soft decoding. AR² [7] leverages process similarity to record and prioritize historically successful read levels at chunk granularity, minimizing ineffective retries. Another direction optimizes LDPC decoding using flash characteristics. LVLDPCC [8] adapts code rates per layer based on RBER variation to reduce iterations. IACPO/ULDCO [9] use decoding iteration count as a success probability indicator, combined with inter-page dependencies, to optimize LLR or voltage and reduce latency. Additionally, [10] introduced a retention-aware design that smartly manages data placement to cut latency without increasing write amplification.

To address these issues, this paper proposes a syndrome-sum based adaptive LDPC decoding scheme, named SSALDPC. The core idea is that after an initial hard decision decoding failure, instead of immediately initiating the high-latency multi-read process, the scheme leverages real-time feedback information intrinsically generated during decoding: the Syndrome Sum (SS). The SS is defined as the number of unsatisfied parity-check equations after a decoding attempt. This metric enables a fast and low-overhead assessment of the error severity in the current frame, as a strong correlation exists between the SS and the RBER, making it a reliable indicator of the channel state. Based on this real-time assessment, the system adaptively selects one of three subsequent decoding strategies: For low error levels, it prioritizes the Efficiency Mode (E-Mode), attempting a limited number of hard decision retries. For medium error levels, it employs the Balanced Mode (B-Mode), initiating a single soft decision decoding attempt. For high error levels, it directly activates the Performance Mode

(P-Mode), adopting a more aggressive soft decision strategy to avoid ineffective and inefficient retry operations. Simulation results demonstrate that the SSALDPC scheme effectively reduces the average number of read retries and decoding iterations, significantly improving system read latency and energy efficiency while guaranteeing reliability.

II. BACKGROUND

A. Reliability Challenges in Flash Memory

3D NAND flash technology has significantly increased storage density by vertically stacking memory cells, with the number of layers evolving from 24 to more than 200. Concurrently, the evolution from multi-level cell (MLC) and triple-level cell (TLC) to quad-level cell (QLC) and penta-level cell (PLC) technologies has further multiplied storage capacity by partitioning the threshold voltage into a greater number of discrete states, albeit at the cost of a significantly narrowed voltage window. However, this dramatic increase in storage density introduces critical reliability challenges. High-density integration exacerbates inherent issues like cell-to-cell and inter-layer interference in 3D architecture. Furthermore, the thinner dielectric layers in charge trap flash structures degrade cell insulation, worsening charge leakage and data retention performance. These factors collectively lead to a significant shift in the threshold voltage distribution over time, driven by both P/E cycling and data retention [11]. Additionally, read and program operations induce significant disturb effects on other cells within the same string or block. Collectively, these factors cause the RBER of QLC/PLC to be several orders of magnitude higher than that of SLC/MLC. Process variation during manufacturing further introduces RBER variations across different chips, wafers, and layers, posing ongoing challenges for error control in high-density flash memory.

B. Error Correction Code

To address the continuously deteriorating channel conditions, error correction code (ECC) technology has become essential for ensuring data reliability in NAND flash memory. Among various ECC schemes, LDPC codes have completely replaced BCH codes as the standard solution in modern high-density 3D NAND flash memory controllers, owing to their exceptional error correction capability and near-Shannon-limit capacity [12], [13]. LDPC decoding operates in two primary modes: hard decision decoding and soft decision decoding. Hard decision decoding uses only single-bit hard information (0 or 1) for decoding, and its error correction capability is limited—generally only effectively handling errors with a RBER below 10^{-3} . In contrast, soft decision decoding achieves more accurate decoding by utilizing multi-bit reliability information from the channel output, specifically the LLR. It significantly outperforms hard decision decoding and can correct errors with a RBER as high as 10^{-2} or even higher.

Fig. 1 illustrates the performance difference in error correction capability between hard decision decoding and soft decision decoding. The X-axis shows the RBER, while the Y-axis plots the frame error rate (FER). The figure further

demonstrates the specific decoding strategies employed across a range of RBER levels, where each strategy corresponds to a distinct number of read-retry operations. For instance, the label "1 hard" denotes a single hard decision read-retry, and "soft 3V" indicates a soft decision read performed with three reference voltages. The "2 hard" strategy was originally proposed in [14]. It can be observed that the number of read-retry operations needed to successfully decode data increases significantly with higher RBER. This mechanism introduces substantial read latency, consequently degrading the overall read performance of flash-based storage systems.

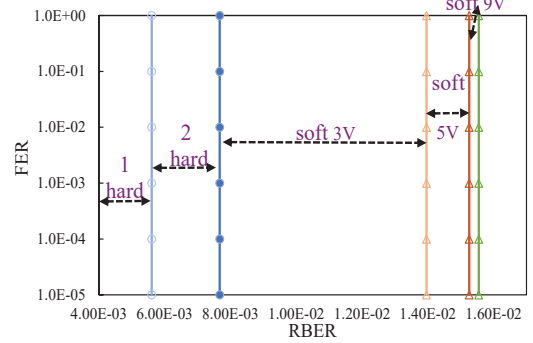


Fig. 1. The error correction ranges of hard decision decoding and soft decision decoding.

The performance advantage of soft decision decoding highly depends on accurate LLR values. To obtain the LLR, the flash controller must adopt a read-retry mechanism, which involves performing multiple reads on the same memory cell using a set of different reference voltages. This process reconstructs the threshold voltage distribution probability to calculate more accurate LLR values. Although this process significantly improves decoding success rate, each additional read operation introduces an extra latency on the order of tens of microseconds along with corresponding power consumption. The accumulated latency and bandwidth overhead from multiple read-retry operations have become a major bottleneck restricting the overall performance of high-performance storage systems—particularly in terms of read latency and throughput.

C. LDPC Decoding and the Min-Sum Algorithm

LDPC codes are a class of linear block codes defined by sparse parity-check matrices. Decoding is typically performed iteratively via message passing on a Tanner graph. Among various decoding algorithms, the Min-Sum Algorithm (MSA) is widely used in flash memory controllers due to its low hardware complexity and competitive error correction performance [15], [16]. The main steps of the MSA are as follows:

Initialization: For each variable node, the initial channel LLR value is taken as the message to be sent to every adjacent check node.

$$\mu_{v \rightarrow c}^{(0)} = \text{LLR}_{\text{ch}}(v) \quad (1)$$

Check Node Update: For each check node, compute the message to be sent to an adjacent variable node. This message is based on the product of the signs and the minimum of the absolute values of the messages received from all other adjacent variable nodes.

$$\mu_{c \rightarrow v}^{(l)} = \left(\prod_{v' \in \mathcal{N}(c) \setminus v} \text{sign}(\mu_{v' \rightarrow c}^{(l-1)}) \right) \cdot \min_{v' \in \mathcal{N}(c) \setminus v} |\mu_{v' \rightarrow c}^{(l-1)}| \quad (2)$$

Here, l denotes the iteration number, and $\mathcal{N}(c)$ represents the set of variable nodes connected to check node c .

Variable Node Update: For each variable node, compute the message to be sent to an adjacent check node. This is calculated as the sum of the initial channel LLR value and all messages received from other adjacent check nodes, excluding the target check node.

$$\mu_{v \rightarrow c}^{(l)} = \text{LLRch}(v) + \sum_{c' \in \mathcal{M}(v) \setminus c} \mu_{c' \rightarrow v}^{(l)} \quad (3)$$

Here, $\mathcal{M}(v)$ represents the set of check nodes connected to variable node v .

Decoding Decision: After each iteration, a hard decision is made on the posterior probability value (the sum of the initial LLR and all incoming messages from check nodes) for each variable node to obtain a candidate codeword. If this codeword satisfies all parity-check equations (i.e., the SS is zero), decoding is successful and the iteration terminates early. Otherwise, steps 2 and 3 are repeated until the maximum iteration count is reached.

The key innovation of SSALDPC lies in utilizing the SS—generated during standard LDPC decoding—as a low-overhead metric for assessing frame error severity. Based on this metric, the system adaptively selects the optimal decoding strategy, avoiding unnecessary read-retry operations and iterations, thereby significantly reducing average read latency.

III. THE PROPOSED SCHEME

This section introduces the proposed SSALDPC decoding mechanism from four aspects: the overall architecture, the SS and its mapping relationship with RBER, the adaptive threshold decision scheme, and the algorithm flow.

A. Overall Architecture

Conventional flash LDPC decoding typically relies on a passive, iterative “read-verify” cycle to handle varying error rates, which often introduces significant computational and latency overhead [17]. In contrast, this paper proposes SSALDPC, an adaptive decoding strategy whose core innovation is a low-complexity Pre-Judgment Layer. Before initiating iterative decoding, this module calculates the SS from the initial hard read. The SS serves as an efficient and reliable metric for estimating the current error level, enabling the system to dynamically select the most suitable decoding path. This approach substantially reduces average decoding latency and power consumption while maintaining reliability.

As illustrated in Fig. 2, the overall architecture of SSALDPC centers on a three-mode decision scheme guided

by the SS value, which activates one of three decoding paths offering distinct performance-complexity trade-offs. The system infers the current error severity level directly from the SS value, and accordingly selects one of the following strategies: for low error levels, SSALDPC employs a single hard decision attempt; for moderate error levels, it uses two hard decision attempts; and for high error levels, the system directly triggers soft decision decoding without prior hard decision trials. If decoding fails in the low or moderate error modes, soft decision decoding serves as a safeguard to ensure correctness.

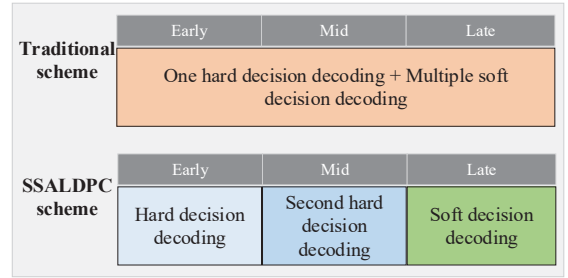


Fig. 2. The overall architecture and workflow of SSALDPC.

B. SS Based on First Decision and Its Mapping to RBER

In the LDPC decoding process, the SS serves as an internal metric that becomes available immediately after the initial hard decision with negligible computational overhead. Specifically, within the MSA or other message-passing decoders, the SS is defined as the number of unsatisfied parity-check equations after the first hard decision is applied to the received sequence. It can be efficiently computed in parallel by taking the Hamming weight of the syndrome vector. Hardware implementations typically employ an adder-tree structure for high-speed summation, introducing only minimal additional timing latency.

The value of the SS lies in its strong nonlinear positive correlation with the frame’s RBER. By measuring 2300 data frames, we obtained the SS values under different RBER conditions and accordingly plotted the relationship curve between the two. As illustrated in Fig. 3, which depicts the relationship between SS and RBER obtained from offline simulations, the SS increases monotonically with RBER—exhibiting a steeper slope in high-RBER regions. This robust statistical behavior enables the SS to serve as a reliable real-time channel-state indicator. By comparing the instantaneous SS value against two pre-optimized thresholds (T_L and T_H), the decoder’s control logic can accurately assess error severity without performing complex online computations, thereby facilitating efficient decoding mode switching.

According to Fig. 1, the RBER measured when switching from single to double hard decision mode is $< 5.61 \times 10^{-3}$, and further increases to 7.67×10^{-3} under soft decision. Using these two RBER levels and the SS–RBER relationship plotted in Fig. 3, we locate the corresponding SS values and take

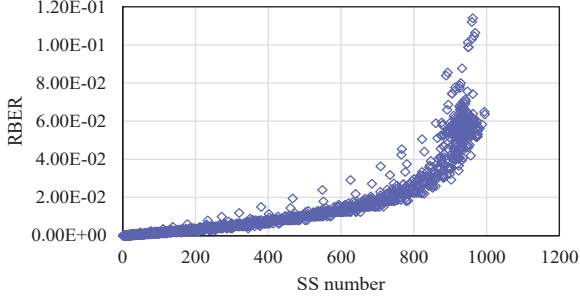


Fig. 3. The relationship of SS number and RBER.

their averages as the decision thresholds. In our experiments, this gives $T_L = 280$ and $T_H = 350$. These thresholds are experiment-dependent and should be determined statistically from data such as those represented in Fig. 3.

C. Adaptive Threshold Decision Scheme

The core of the SSALDPC decoding architecture is its adaptive decision unit, which classifies the channel state into three regions using two pre-optimized thresholds—the low threshold T_L and the high threshold T_H —and activates corresponding decoding modes to optimally balance reliability, latency, and power consumption.

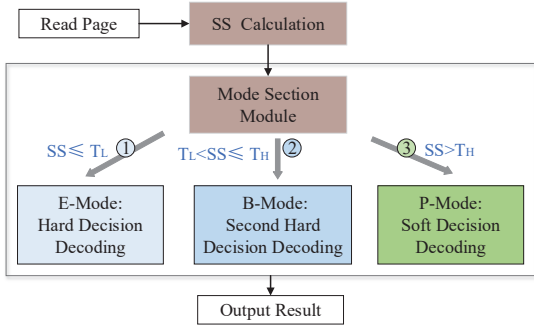


Fig. 4. The flow of SSALDPC scheme.

The thresholds T_L and T_H are determined through offline statistical learning. The decision rule is mathematically expressed as follows:

- If $SS \leq T_L$: indicates that the channel is in a low RBER state, typically corresponding to the early stage of flash memory lifespan. Empirical data show that the success probability of single hard decision decoding, P_{succ}^{HDD} , is very high. The system enters E-Mode, aiming to complete decoding with minimal overhead.
- If $T_L < SS \leq T_H$: indicates that the channel is in a medium RBER state with complex error patterns, often associated with the middle stage of device life. The system enters B-Mode, initiating a secondary decision process involving one voltage read-retry to enhance reliability without significantly increasing latency.
- If $SS > T_H$: indicates that the channel is in a high RBER state, typically occurring in the late stage of flash life, where the success probability of initial hard decision

Algorithm 1 SSALDPC Decoding Procedure

Input: Received codeword vector \mathbf{r}

Output: Decoded codeword $\hat{\mathbf{u}}$ or decoding failure

- 1: Perform initial hard decision: $\mathbf{y} \leftarrow \text{HD}(\mathbf{r})$
- 2: Compute syndrome: $\mathbf{s} \leftarrow \mathbf{H}\mathbf{y}^T \bmod 2$
- 3: Calculate syndrome sum: $SS \leftarrow w_h(\mathbf{s})$
- 4: **if** $SS = 0$ **then**
- 5: **return** $\hat{\mathbf{u}} \leftarrow \mathbf{y}$ ▷ Decoding succeeds immediately
- 6: **end if**
- 7: **if** $SS \leq T_L$ **then**
- 8: Mode \leftarrow E-Mode
- 9: **else if** $SS \leq T_H$ **then**
- 10: Mode \leftarrow B-Mode
- 11: **else**
- 12: Mode \leftarrow P-Mode
- 13: **end if**
- 14: Execute the corresponding decoding subroutine for the selected mode
- 15: **return** decoding result from the subroutine

decoding is very low. The system directly enters P-Mode, concentrating all resources on high-gain soft decision decoding to avoid latency penalties from futile operations.

It is emphasized that both E-Mode and B-Mode incorporate built-in reliability mechanisms. If decoding fails in either mode, the system automatically and seamlessly switches to P-Mode to perform high-performance soft decision decoding, ensuring that final data reliability is not compromised.

D. SSALDPC Algorithm Flow

The proposed SSALDPC decoding framework is implemented as outlined in Algorithm 1. The algorithm accepts the received codeword \mathbf{r} as input and incorporates an adaptive control mechanism designed to jointly optimize decoding performance and energy efficiency. In contrast to conventional static decoding schemes, which follow a fixed operational sequence regardless of channel state, SSALDPC dynamically adjusts its decoding strategy in response to real-time signal conditions—captured via the SS metric.

The SSALDPC method offers several key advantages by leveraging the inherently available SS metric within the decoding process:

First, the method utilizes the SS—an internally generated metric—as a real-time channel state indicator. This enables the system to dynamically assess error levels and autonomously adjust the decoding strategy without introducing additional computational overhead.

Second, the SS value guides adaptive decision-making, effectively reducing unnecessary read-retry operations and voltage re-reads. By aligning decoding effort with actual error conditions, the method significantly improves operational efficiency while maintaining decoding reliability.

Finally, a built-in fallback mechanism, triggered based on the SS value, ensures reliable error recovery. If initial decoding

attempts fail, the system seamlessly transitions to higher-performance decoding modes, guaranteeing data integrity without requiring manual intervention or external monitoring.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

This chapter evaluates the performance of the proposed SSALDPC method via simulations. After describing the experimental setup, we compare SSALDPC with existing schemes in terms of decoding latency, number of read operations, and overhead.

A. Experimental Setup

To accurately emulate a realistic NAND flash environment and evaluate the performance of SSALDPC, we developed a simulation platform using the C language. This platform integrates a NAND flash channel model and a configurable LDPC decoder. The flash channel model employs a Gaussian Mixture Model-based threshold voltage distribution to simulate the effects of distribution broadening and shifting caused by different P/E cycles (e.g., 1k, 3k, 5k) and data retention times (e.g., 6 months, 9 months). By adjusting the overlap between adjacent distributions, we generate a range of RBER conditions from 10^{-4} to 2×10^{-2} , covering scenarios from fresh to heavily worn memory cells. The LDPC decoder uses a Quasi-Cyclic LDPC (QC-LDPC) code with a code length of $n = 18656$ and a code rate of $R = 0.9$, consistent with codes widely used in commercial flash controllers. The core decoding algorithm is the min-sum algorithm, with a maximum iteration count set to 30. The LLR values required for soft decision decoding are obtained through up to 9 read-retry operations using distinct reference voltages. The compared schemes include: a traditional fixed-pipeline scheme using “one hard decision attempt followed by maximum soft decision attempts” [17], the Double Hard Decision (DHD) scheme [14], and the proposed SSALDPC scheme. The thresholds T_L and T_H used in SSALDPC are optimized through offline simulation. Key performance metrics include: (i) the average number of read operations required per successful decoding, and (ii) the average read latency per successful decoding instance, where T_{read} denotes the latency of a single read operation and T_{iter} the latency per decoding iteration.

B. Results Analysis

We first compare the number of read operations required by different schemes under varying RBER conditions. As shown in Fig. 5, SSALDPC effectively activates E-Mode in the low RBER region ($< 5.61 \times 10^{-3}$), significantly reducing unnecessary read retries. By predicting error levels early and initiating corrective processes before substantial errors develop, the scheme decreases system response latency and improves overall efficiency. As RBER increases to the medium-high region ($> 5.61 \times 10^{-3}$), SSALDPC adaptively increases the use of B-Mode and P-Mode. It is particularly noteworthy that

the introduction of B-Mode markedly suppresses the initiation of read-retry operations. Specifically, for RBER below 1.4×10^{-2} , the proposed scheme reduces read-retry initiations by 33.3% compared to conventional schemes. Under high RBER conditions, where the read-retry count of traditional schemes increases sharply, SSALDPC avoids between 11.1% and 20% of read-retry operations by directly switching to P-Mode, demonstrating strong adaptive capability.

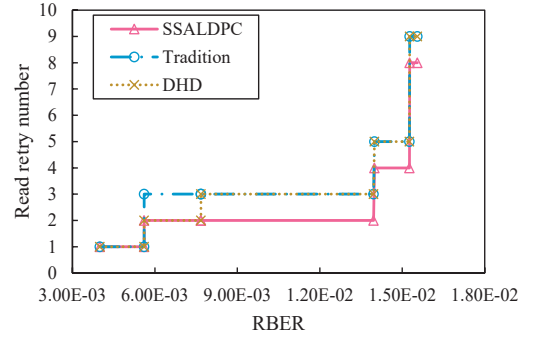


Fig. 5. The comparison of the read retry number for three schemes.

To further elaborate, the advantage of bypassing unnecessary retries not only improves responsiveness but also reduces energy consumption and wear on memory cells. When compared with other advanced schemes, SSALDPC maintains a more stable retry count even as RBER increases, highlighting its practical robustness. For instance, relative to the framework presented in Scheme [17], our approach reduces retry initiations by 24.4%, and by 16.1% compared to the DHD scheme, as supported by the reduction ratios shown in Fig. 6. These results confirm that SSALDPC provides a scalable and efficient solution for NAND flash storage under a wide range of error conditions, ensuring both performance and reliability.

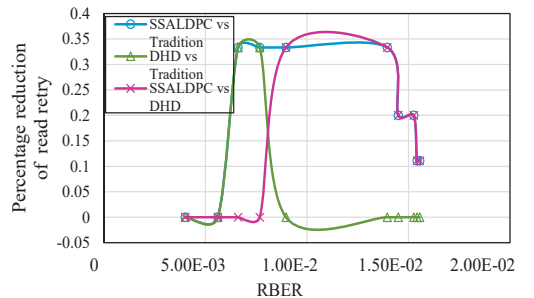


Fig. 6. The comparison of reduced read retry for three schemes.

Further comparisons under different Program/Erase (P/E) cycles and data retention times, as illustrated in Fig. 7, reveal that SSALDPC consistently yields the lowest number of read retries. Specifically, at 5k P/E cycles with extended retention times of 6 months and 9 months—conditions representative of aged NAND flash memory—SSALDPC significantly outperforms alternative approaches. This advantage stems from its ability to dynamically activate appropriate error recovery modes tailored to the deteriorating health of the memory cells.

In contrast, under scenarios characterized by lower RBER, all three schemes exhibit comparable performance since initial hard decision attempts are generally sufficient to achieve successful decoding, minimizing the need for repeated retries.

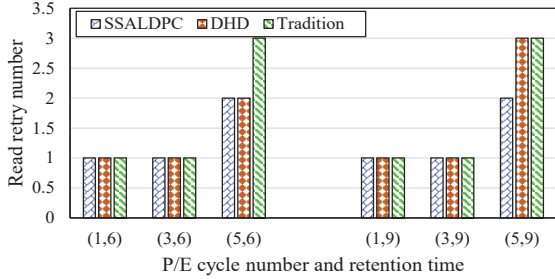


Fig. 7. The comparison of the iteration number for various P/E cycles and retention time.

Additionally, we evaluated the decoding iteration counts and overall read latency across the schemes over a range of RBER conditions. The latency analysis is based on a system where each read operation incurs $60 \mu s$, each hard decision decoding iteration requires $5 \mu s$, and each soft decision decoding iteration takes $3 \mu s$. Under these constraints, the normalized read latency for each scheme was calculated and compared, as summarized in Fig. 8. The results clearly demonstrate that the SSALDPC scheme achieves the lowest decoding latency across all tested RBER conditions. In certain cases, it matches the performance of the DHD scheme, particularly in low-error regimes where both strategies converge toward minimal retries. More importantly, as RBER increases, SSALDPC’s adaptive mode-switching strategy proves highly effective in circumventing unnecessary operations, thereby substantially reducing cumulative latency. These findings underscore the practical efficacy of SSALDPC in enhancing responsiveness and efficiency in real-world NAND flash applications.

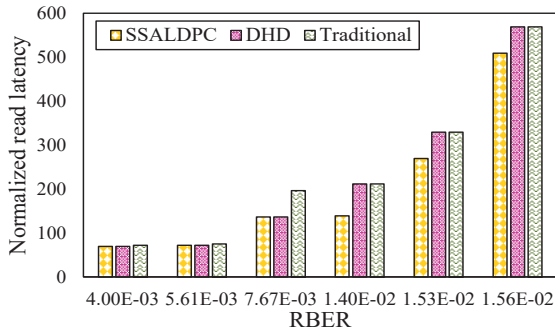


Fig. 8. The comparison of the normalized read latency under different RBER.

Regarding hardware overhead, the additional cost introduced by SSALDPC is minimal. The scheme intelligently reuses the SS calculation module—a core component inherently present in the LDPC decoder, primarily used for iterative termination checking—thus avoiding the need for dedicated computing units to support its early-decision mechanism. As a result, the main incremental hardware cost consists of only two

registers for storing the dynamic thresholds (T_L and T_H) and a small amount of control logic. Compared to approaches that rely on large lookup tables or extensive historical data storage, SSALDPC’s efficient reuse of existing decoder resources leads to a negligible increase in both area and power consumption, offering a clear advantage in implementation complexity.

C. Overhead Analysis

The hardware and computational overhead of SSALDPC is minimal. The method uses the SS, which becomes available immediately after the first hard decision decoding and requires only a Hamming weight calculation—easily implemented via an adder-tree structure, introducing negligible latency. SSALDPC relies solely on two pre-optimized thresholds (T_L , T_H), eliminating the need for large lookup tables or complex online computations. Mode switching is deterministic and executes in constant time, adding no perceptible delay. In comparison to conventional read-retry schemes, SSALDPC imposes virtually no additional resource burden on the memory controller.

V. CONCLUSION

This paper addresses the read latency issue in high-density NAND flash memory by proposing a syndrome sum-based adaptive LDPC decoding scheme. The approach utilizes the SS obtained after the initial decoding attempt as a low-overhead channel-state indicator to drive a triple-mode decoding architecture, which includes Efficient, Balanced, and Performance modes. Guided by a strategy that attempts the most efficient decoding path first and seamlessly falls back to higher performance modes when necessary, SSALDPC dynamically selects the optimal decoding strategy for given error conditions. The proposed method reduces unnecessary read retries and decoding latency through adaptive syndrome sum-based mode switching.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China under Grant No.62372197, No. U22A2071, No.62102155, in part by Shandong Provincial Natural Science Foundation under Grant No.ZR2024LZH004, in part by the 111 Project under Grant No. B07038, in part by the Natural Science Basic Research Program of Shaanxi Province under Grant 2025JC-YBQN-835, in part by General Special Project of Scientific Research Program of Shaanxi Provincial Department of Education under Grant No. 25JK0569, in part by the Jiangxi Provincial Natural Science Foundation under Grant No.20252BAC240214.

REFERENCES

- [1] Y. Fan, Y. Du, and S. H.Noh, “Remapcom: Optimizing compaction performance of LSM trees via data block remapping in SSDs,” in *2025 Design, Automation and Test in Europe Conference (DATE)*, 2025, pp. 1–7.
- [2] R. Frickey, J. Doller, R. Norton, R. Sancho, R. Sayyad, D. Ustinov, R. Wang, and H. Xu, “Comparing the reliability of solid-state drives based on TLC and QLC NAND flash memories (invited),” in *2024 IEEE International Reliability Physics Symposium (IRPS)*, 2024, pp. 3A.1–1–3A.1–7.

- [3] Y. Du, Y. Zhou, M. Zhang, W. Liu, and S. Xiong, "Adapting layer RBERs variations of 3D flash memories via multi-granularity progressive LDPC reading," in *2019 56th ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.
- [4] R. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.
- [5] C. Duangthong, W. Phakphisut, and P. Wardkein, "Efficient design of read voltages and LDPC codes in NAND flash memory using density evolution," *IEEE Access*, vol. 11, pp. 74 420–74 437, 2023.
- [6] M. Zhang, F. Wu, Q. Yu, and C. Xie, "PEAR: Unbalanced inter-page errors aware read scheme for latency-efficient 3-D NAND flash," *IEEE Transactions on Device and Materials Reliability*, vol. 24, no. 1, pp. 49–58, 2024.
- [7] S. Nie, Z. Li, F. Yu, S. Liu, and W. Wu, "Adaptive read level recording for read performance improvement in 3-D NAND flash," *IEEE Transactions on Reliability*, pp. 1–11, 2025.
- [8] L. Cui, F. Wu, M. Zhang, Z. Zhao, X. Zhang, K. Jiang, and C. Xie, "Enhanced LDPC coding for 3D TLC NAND flash memory: Leveraging RBER difference from intra-layer variation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2025.
- [9] D. Wei, Y. Wang, D. Zhang, H. Xiang, and L. Qiao, "Iteration and SDA-Driven LDPC decoding latency reduction for 3-D TLC NAND flash memory," *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 1–13, 2025.
- [10] T.-Y. Wang, C.-W. Tsao, Y.-H. Chang, and T.-W. Kuo, "Retention-aware read acceleration strategy for LDPC-based NAND flash memory," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 12, pp. 4597–4605, 2023.
- [11] S. Yang, M. Zhang, X. Zhan, P. Guo, X. Zhao, G. Yang, X. Guo, J. Wu, F. Wu, and J. Chen, "Retention accelerated testing for 3-D QLC nand flash memory: Characterization, analysis, and modeling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 44, no. 7, pp. 2779–2788, 2025.
- [12] Y. Song, Y. Lv, W. Li, J. Liu, and L. Shi, "Revisiting multiple ECC on high-density NAND flash memory," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 33, no. 6, pp. 1679–1692, 2025.
- [13] M. Sandell and A. Ismail, "Machine learning for LLR estimation in flash memory with LDPC codes," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 2, pp. 792–796, 2021.
- [14] L. Cui, Y. Wang, R. Xiao, M. Li, X. Liu, and X. Hei, "DHD: Double hard decision decoding scheme for NAND flash memory," in *2025 Design, Automation and Test in Europe Conference (DATE)*, 2025, pp. 1–7.
- [15] T. Kim and J. Sung Park, "Neural self-corrected min-sum decoder for NR LDPC codes," *IEEE Communications Letters*, vol. 28, no. 7, pp. 1504–1508, 2024.
- [16] M. Yu, X. Zhai, M. Hu, S. Tao, Y. Fang, and G. Han, "Adaptive normalized min-sum decoding algorithm for LDPC codes in flash memory," in *2025 IEEE/CIC International Conference on Communications in China (ICCC)*, 2025, pp. 1–6.
- [17] L. Cui, F. Wu, X. Liu, M. Zhang, R. Xiao, and C. Xie, "Improving LDPC decoding performance for 3D TLC NAND flash by LLR optimization scheme for hard and soft decision," *ACM Trans. Design Autom. Electr. Syst.*, vol. 27, pp. 5:1–5:20, 2022.