# A Novel NAND Flash Memory Architecture for Maximally Exploiting Plane-Level Parallelism

Myeongjin Kim, Wontaeck Jung, Hyuk-Jun Lee, and Eui-Young Chung

***Abstract*—Solid-state drive (SSD) has become one of the most dominant storage devices and is rapidly replacing conventional storage devices. The core component of SSD is NAND flash memory (NFM), where the actual data are stored. Cost pressure is the most critical factor limiting the further deployment of SSDs and past researches have focused on developing cost-effective high-density NFM. Although the cost-driven technology development increases per-chip capacity, it reduces channel-/way-level parallelisms for the given device capacity, resulting in the performance degradation. Such observation directs us to focus on a novel NFM architecture exploiting plane-level parallelism. The distinct features of this architecture are: 1) enabling a decoupled word-line (WL) selection for the mated planes and 2) segmenting each plane into subplanes for further maximizing the plane-level parallelism. The experimental results show that decoupled WL selection improves the throughput by up to 21.3% with a marginal overhead of less than 1%, compared to the conventional NFM architecture. In addition, adopting the plane segmentation improves the throughput by up to 43.9% with an additional overhead of 14%. Considering the tradeoff between performance and overhead, the proposed NFM architecture is a cost-efficient method to secure high performance under decreasing channel-/way-level parallelisms in high-density NFM.**

***Index Terms*—Cost-efficient, high-performance, NAND flash memory (NFM), plane-level parallelism.**

## I. Introduction

Many attractive features of NAND flash memory (NFM) enable NAND flash-based storage device (NFSD) to rapidly replace the conventional hard disk drive. The success attributes to the dramatically enhanced performance of NFSD. However, the high cost of NFSD slows down the momentum for replacement. Thus, past researches have mainly focused on cost reduction. Multilevel cell (MLC)-based NFM [1][1] increases the bit density by storing multiple bits in a single memory cell, but it fails to fully solve the cost issue as the process technology scaling slows down. Recently, 3-D stacked NFM [2] has been widely adopted. It places more cells in the unit area in a vertical manner and achieves higher bit density without process technology scaling. Unfortunately, the performance of NFSD is affected adversely by the cost reduction. Using higher density NFM chips means fewer numbers of chips required to build a storage of the same capacity. Such direction might be advantageous from the cost and form factor perspectives. However, it could be undesirable from the performance perspective, since fewer NFM chips limit the architecture-level parallelism, e.g., channel- and way-level parallelisms, which is the key parameter for enhancing the performance of NFSD.

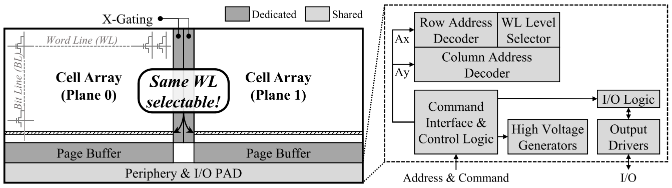


Fig. 1. Architecture of conventional NFM.

An alternative solution to mitigate this problem is to exploit plane-level parallelism, which has been less studied because it could adversely affect the cost. Contemporary NFM architectures impose an architectural constraint to limit plane-level parallelism due to the cost issue. More specifically, adjacent planes are mated and the mated planes[2] share most of internal peripheral circuitry to reduce an area overhead. The mated planes can be accessed in parallel only when their physical page addresses are identical [3]. Such a restriction on addressing limits the exploitation of plane-level parallelism. Previous researches for exploiting parallelism tried to maximize the parallelism while maintaining the imposed addressing restriction [4]–[6]. In this brief, we propose a more aggressive method which eliminates the architectural constraint and dramatically improves the performance with a reasonable area overhead.

The main contribution of our method is twofold. First, we propose an NFM architecture which allows the decoupled word-line (WL) selection for the mated planes and a new flash translation layer (FTL) which is optimized for the proposed architecture to exploit the plane-level parallelism. Second, we propose to segment each plane into subplanes for further maximizing the plane-level parallelism. Such segmentation also improves the access latency of NFM by shortening the absolute width and height of an NFM cell array.

## II. Background and Related Works

### A. Basic NFM Architecture

Fig. 1 shows the conventional NFM (CNFM) architecture [7]. It consists of NFM cell arrays, page buffers, and other peripheral circuitry. The NFM cell arrays are grouped into two or more planes. Each plane owns a dedicated page buffer; hence, each plane can operate independently [8]. However, the peripheral circuitry can serve only one page address at a time and is shared by the planes as shown in Fig. 1. Hence, the plane-level parallel operation, i.e., multiplane operation, is allowed only when the page addresses of planes in the same group are identical even though each plane has its dedicated page buffer [3]. Such limitation inherently degrades the exploitation

Manuscript received October 29, 2018; revised January 26, 2019; accepted March 12, 2019. Date of publication April 11, 2019; date of current version July 24, 2019. This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (2016R1A2B4011799); in part by the MOTIE (Ministry of Trade, Industry & Energy) (10080722) and KSRC (Korea Semiconductor Research Consortium) support program for the development of the future semiconductor device; and in part by Samsung Electronics Company, Ltd., Hwaseong, Korea. The EDA tool was also supported by the IC Design Education Center (IDEC), Korea. *(Corresponding author: Eui-Young Chung.)*

M. Kim and E.-Y. Chung are with the School of Electrical and Electronic Engineering, Yonsei University, Seoul 03722, South Korea (e-mail: eychung@yonsei.ac.kr).

W. Jung is with Samsung Electronics Company, Ltd., Hwaseong 18448, South Korea.

H.-J. Lee is with the School of Computer Science and Engineering, Sogang University, Seoul 04107, South Korea.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2019.2905626

[1] Recently, quadruple-level cell (QLC)-based NFM is actively researched.

[2] *Mated planes* refer to the physically adjacent planes within a single way.

TABLE I

NFM ARCHITECTURE TRENDS

| Name | Page size (KB) | # of pages per block | Block size (KB) | Total cap. (GB) |
|---|---|---|---|---|
| ISSCC 2014 [13] | 8 | 384 | 3072 | 16 |
| ISSCC 2015 [14] | 16 | 384 | 6144 | 16 |
| ISSCC 2016 [15] | 16 | 576 | 9216 | 32 |
| ISSCC 2017 [16] | 16 | 768 | 12288 | 64 |
| ISSCC 2018 [17] | 16 | 1024 | 16384 | 128 |

of plane-level parallelism. In order to increase the plane-level parallelism, it is necessary to implement the peripheral circuitry dedicated to each plane. This area overhead, however, prevents most of device manufacturers from adopting such approach.

### B. Parallelism in NFSD

The parallelization techniques can be categorized into channel-level, way-level, and plane-level ones [9]. Past researches have focused on the channel-level and way-level schemes, whereas the plane-level schemes have been rarely investigated. In [10], it is reported that the performance and the endurance of NFSD are dynamically affected by the priority order of exploiting different levels of parallelism. We classify the previous works in these three categories and summarize them in this section.

*1) Channel- and Way-Level Parallelisms:* An intuitive approach for performance improvement is to allocate multiple independent channels to NFMs. Each channel can be accessed independently; hence, the slow NFMs can be operated in parallel [11]. The major drawback of this approach is nonmarginal area overhead of channel interconnects which is a critical design parameter in NFSD design. Extensive researches have addressed this issue and many commercial NFSDs employ only limited number of channels. Due to this limitation, some proposed a multichannel-based FTL which considers channel-level and plane-level parallelisms together to further improve the performance and provide better wear leveling [12].

Interleaving is a cost-effective way-level technique as it does not require additional area overhead. It tries to maximize the bandwidth utilization of each independent channel. In general, the number of NFMs attached to a single channel can be easily computed by dividing channel bandwidth by NFM bandwidth [3].

Many commercial NFSDs combine these channel- and way-level techniques to maximize the performance.

*2) Plane-Level Parallelism:* In CNFM architecture, the request rescheduling methods have been proposed to exploit the plane-level parallelism [4]–[6]. They scan the request queue and reschedule the requests to better utilize multiplane operations. The drawback of this approach is that the chance for finding the requests accessing the same page address is reduced as the bit density of NFMs becomes higher. This is because the block size, i.e., number of pages per block, increases more rapidly than the page size [18]. In particular, as shown in Table I, the number of pages per block has continuously increased, while the page size has been saturated. Therefore, exploiting plane-level parallelism is harder for contemporary NFSDs as the bit density is getting higher. In this brief, we tackle the architectural constraint of CNFM, i.e., same page address restriction for multiplane operations, for enhancing the plane-level parallelism.

## III. NFSD WITH ENHANCED PLANE-LEVEL PARALLELISM

We introduce two major architectural improvements of NFM for maximally exploiting plane-level parallelism. The first improvement eliminates the restriction on the page address to increase multiplane operations by dedicating the peripheral circuitry to each plane and optimizing the FTL for the proposed NFM architecture. The second improvement segments a plane into subplanes to exploit the plane-level parallelism more aggressively and reduce the access latency of NFM. Both improvements enhance the plane-level parallelism with reasonable area costs.

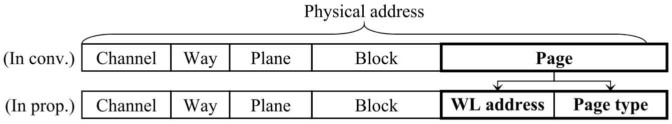


Fig. 2. Address segmentation in the conventional and the proposed architecture.

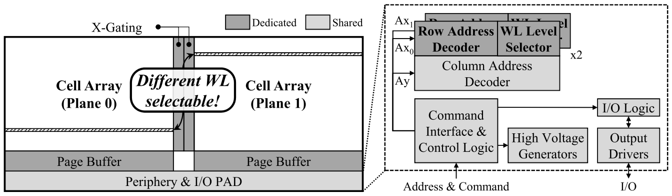


Fig. 3. Architecture of decoupled WL NFM.

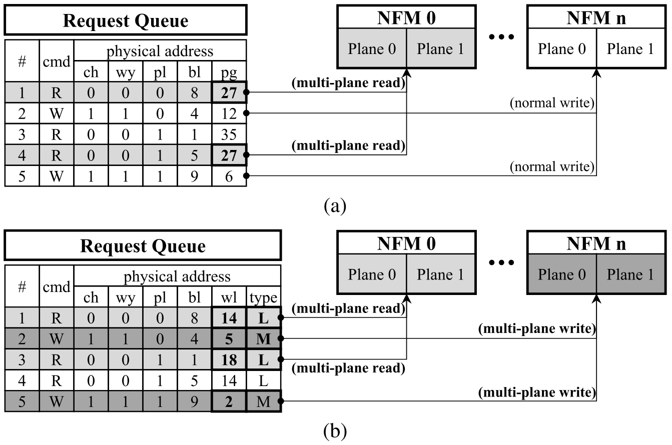
### A. NFM Architecture With Decoupled WL Selection

For the decoupled WL selection, the proposed method requires modifications in hardware and software. The former represents the architectural implementation for decoupled WL selection, while the latter represents the FTL optimization to support the architectural change.

*1) Architectural Implementation:* Once a physical address comes in, it is partitioned into the channel, way, plane, block, and page address as shown in Fig. 2. The page address actually consists of *WL address* and *page type*. The WL address determines a physical location within NFM cell arrays for the given page. On the other hand, the page type maps the given physical page into one of logical pages (e.g., LSB and MSB pages in MLC NFM). As discussed earlier, the CNFM requires the same page address for multiplane operations. Such a restriction can be eliminated to allow only different WL addresses or both different WL addresses and page types. For different WL addresses, each plane requires a separate address decoding and voltage-level selection circuitry. These additional row address decoder and WL-level selector consist of small logic gates. On the other hand, the control circuitry over the operation timing and input voltage can be shared by the mated planes.

For supporting different page types, we need to control the operation timing and input voltage separately for each page type [19]. In this case, peripheral circuitry should be duplicated, which results in significant area overhead.

Fig. 3 shows the proposed decoupled WL NFM (DW-NFM) architecture that resolves the constraint of CNFM on the plane-level access. The dotted square box represents the circuitry for decoupled WL selection, which includes dedicated row address decoders and WL-level selectors for each plane. The row address decoder transmits a block address to the x-gating circuitry for physical block selection and a page address to the WL-level selector for physical page selection. The WL-level selector includes the control logic for voltage selection zone, which determines a proper voltage level for the adjacent WLs to prevent cell disturbance. It also includes the WL control logic that selects an appropriate voltage level for each read/write operation based on the physical location of selected WL. Dedicated peripheral circuitry provides separate page address and voltage level to each plane when different WL addresses are given.



Request Queue

| # | cmd | physical address | | | | |
|---|---|---|---|---|---|---|
| | | ch | wy | pl | bl | pg |
| 1 | R | 0 | 0 | 0 | 8 | **27** |
| 2 | W | 1 | 1 | 0 | 4 | 12 |
| 3 | R | 0 | 0 | 1 | 1 | 35 |
| 4 | R | 0 | 0 | 1 | 5 | **27** |
| 5 | W | 1 | 1 | 1 | 9 | 6 |

Request Queue

| # | cmd | physical address | | | | | |
|---|---|---|---|---|---|---|---|
| | | ch | wy | pl | bl | wl | type |
| 1 | R | 0 | 0 | 0 | 8 | **14** | **L** |
| 2 | W | 1 | 1 | 0 | 4 | **5** | **M** |
| 3 | R | 0 | 0 | 1 | 1 | **18** | **L** |
| 4 | R | 0 | 0 | 1 | 5 | 14 | L |
| 5 | W | 1 | 1 | 1 | 9 | **2** | M |

Fig. 4. Scheduling method. (a) Page address aware. (b) Page-type aware.

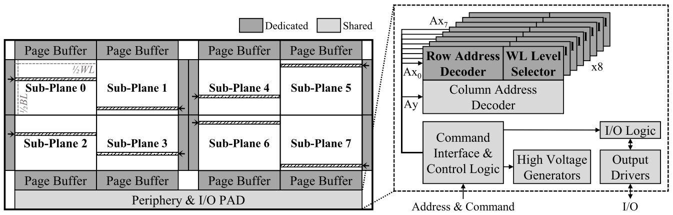


Fig. 5. Architecture of segmented decoupled WL NFM.

Note that DW-NFM does not support for multiplane operations on different page types because the related circuitry incurs a large area overhead. However, the experimental results show that the performance is notably improved just by the decoupled WL selection.

*2) FTL Optimization:* Along with the architectural change, we optimize FTL to fully exploit plane-level parallelism. We implement a page-type aware scheduling algorithm within the FTL to maximally utilize the multiplane operations, which reorders enqueued requests according to the page type.

Fig. 4 shows the difference between page address aware and page-type aware schedulings. The page address aware scheduling is used by CNFM architecture due to the aforementioned restriction on multiplane operations. It can only pair the requests indicating the same page address within the mated planes, as depicted in Fig. 4(a). On the other hand, the page-type aware scheduling shown in Fig. 4(b) is able to pair the enqueued requests indicating the same page type within the mated plane. It increases multiplane operations, leading to significant performance gain whereas it incurs marginal time overhead for scheduling the enqueued requests.

### B. NFM Architecture With Plane Segmentation

To further enhance the plane-level parallelism, we propose segmented decoupled WL NFM (SDW-NFM) architecture. As shown in Fig. 5, each plane is segmented into four subplanes, and each subplane owns dedicated x-gating circuitry and a page buffer as the conventional plane does. It also includes the dedicated row address decoder and WL-level selector, which enable the decoupled WL selection. More exploitable planes increase the plane-level parallelism. In other words, up to eight requests, as many as the number of subplanes, can be processed by a multiplane operation depending on the characteristic of input workload. Note that the page-type aware scheduling is further extended to better exploit this architecture.

In addition, SDW-NFM architecture improves latency due to the segmented subplanes. The access latency of NFM, $t$R and $t$PROG, is directly related to the width and height of NFM cell array, i.e., WL length and bitline (BL) length [20]. SDW-NFM architecture further improves the access latency of NFM as the WL and BL are shortened by half, respectively. Specifically, $t$R is improved by 25%–30% due to the reduced WL setup and BL precharing time, and $t$PROG is improved by 15%–20% due to the reduced program verification time.

TABLE II

NFSD Configurations

| NFSD capacity | | 64 GB MLC | | |
|---|---|---|---|---|
| # of channels* | | 1 | | |
| # of ways | | 1 | 2 | 4 |
| NFM density (GB) | | 64 | 32 | 16 |
| # of planes per NFM | | 2 | | |
| Page size (KB) | | 8 | | |
| # of pages per block | | 512 | 256 | 128 |
| Latency | $t$R ($\mu$s) | 45 | | |
| | $t$PROG ($\mu$s) | 700 | | |
| | $t$BERS ($ms$) | 3.5 | | |
| Depth of request queue | | 32 | | |

*Each channel operates independently,thereby the channel-level parallelism is orthogonal to the plane-level parallelism. Therefore, we fix the number of channels to 1 to focus on the evaluation of plane-level parallelism itself.

TABLE III

Experimental Workloads

| Name | Number of request | Read ratio(%) | Avg. request length(sector) | | Random ratio(%) | |
|---|---|---|---|---|---|---|
| | | | Read | Write | Read | Write |
| Finantial1 | 5334938 | 23.2 | 4.5 | 7.5 | 94.3 | 86.6 |
| Finantial2 | 3699194 | 82.3 | 4.6 | 5.8 | 94.3 | 87.8 |
| MSN | 1081878 | 70.0 | 19.5 | 21.2 | 0.0 | 17.6 |
| MSR | 1211035 | 11.9 | 47.4 | 14.5 | 53.3 | 72.4 |
| TPC_C | 348644 | 62.9 | 16.2 | 18.3 | 0.1 | 0.1 |
| TPC_E | 1147310 | 90.4 | 16.0 | 24.1 | 0.0 | 3.9 |

On the other hand, SDW-NFM architecture provides a considerable performance gain, it requires some area overhead. For high-performance NFSD, SDW-NFM is a good alternative solution to increasing the number of channels and ways.

## IV. Experiments

### A. Experimental Setup

We implement an in-house simulator to evaluate our proposed method, which includes the page-level FTL [21] and greedy garbage collection [22]. The hardware configurations of NFSD are summarized in Table II. We fix the total NFSD capacity as 64 GB for fair comparison and vary the NFM density from 16 to 64 GB according to the number of ways. The latency of read, write, and erase operation is obtained from [14]. The workloads used in experiments are listed in Table III. They are extracted from [23] and [24]. We assume that NFSD is a serial advanced technology attachment (SATA) device and the depth of native command queue (NCQ) is 32 as specified in SATA specification [25]. The depth of a request queue for the proposed scheduling schemes is also configured to be 32 to properly handle multiple requests from SATA interface.

### B. Experimental Results

*1) Parallelism Exploitation:* The exploitation ratio of multiplane operations is shown in Fig. 6. In CNFM, 2.1%–2.2% of total requests turn into multiplane operations, while DW-NFM and SDW-NFM exploit 49.3%–62.1% and 67.2%–78.4% of total requests, respectively. The considerable number of total requests can be handled with multiplane operations, thanks to the decoupled WL selection within mated planes. The exploitation ratio is further increased by 16.3%–17.9% in SDW-NFM as the number of selectable planes

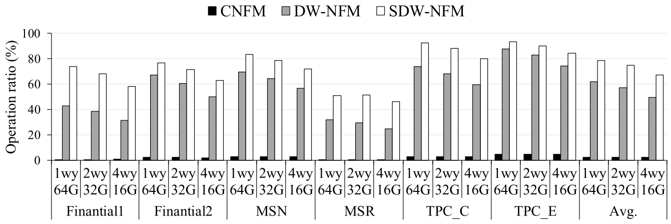
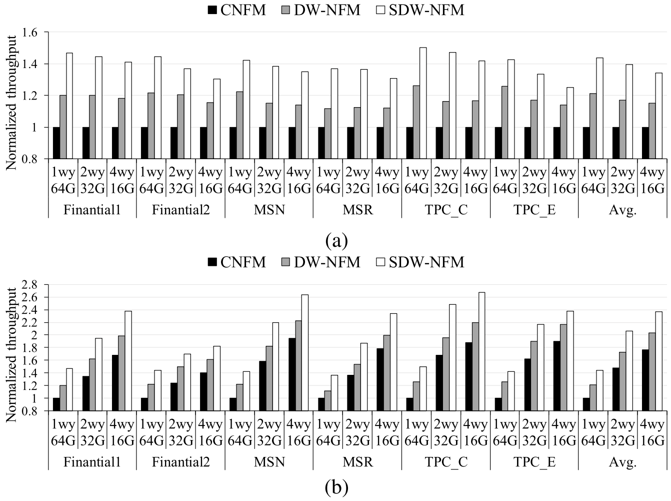


Fig. 6. Exploitation ratio of multiplane operation.



(a)

(b)

Fig. 7. Throughput improvement. (a) Normalized to CNFM. (b) Normalized to one-way 64-GB CNFM.

increases due to the plane segmentation. The exploitation ratio is most enhanced when the higher density NFM, i.e., one-way 64-GB NFM, is used because it provides more opportunities to perform multiplane operations with a wider range of page addresses.

*2) Throughput:* Fig. 7(a) shows the throughput improvement normalized to CNFM for one, two, and four ways. In DW-NFM, the throughput is improved by 21.3% for one-way 64-GB NFM, 16.9% for two-way 32-GB NFM, and 15% for four-way 16-GB NFM on average over CNFM. DW-NFM shows more improvement when the higher density NFM is used, since the plane-level parallelism becomes a dominant factor as the other parallelism factors, channel- and way-level parallelisms, are less available as shown in Fig. 6.

The results for SDW-NFM show that the throughput is improved by 43.9%, 39.5%, and 34% respectively, over CNFM. Compared with DW-NFM, it is improved by 18.2% on average.

Fig. 7(b) shows the throughput improvement normalized to one-way 64-GB CNFM. It presents the throughput improvement from exploiting plane-level and way-level parallelisms at the same time. The throughput of one-way 64-GB SDW-NFM is almost comparable to that of two-way 32-GB CNFM and the throughput of two-way 32-GB SDW-NFM is superior to that of four-way 16-GB CNFM. This result implies that SDW-NFM provides an equivalent or better throughput even if the number of ways halves due to the adoption of high density NFM. We discuss about the relationship between throughput improvement and the required area in Section IV-B3.

Fig. 8 shows the sensitivity to the depth of a request queue as it varies from 16 to 256. The throughput gradually increases with respect to the queue depth. Moreover, the improvement rate is higher in SDW-NFM than DW-NFM. This is because parallelism for multiplane operations can be exploited more efficiently under the segmented subplanes. This result also implies that the performance can be potentially improved with the larger request queue.

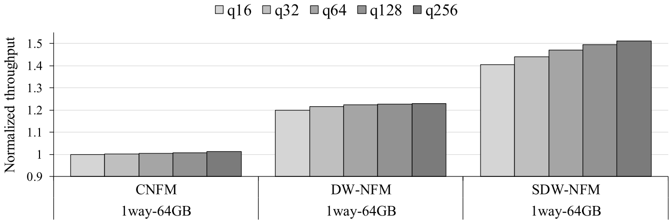


Fig. 8. Sensitivity to the depth of a request queue.

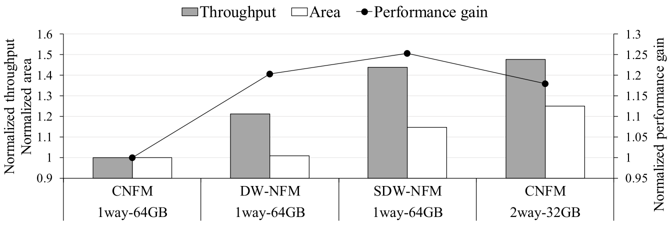


Fig. 9. Performance gain.

*3) Area Overhead:* DW-NFM architecture requires a dedicated row address decoder and a WL-level selector. As the row address decoder consists of only a few logic gates, its overhead is small. In contrast, the WL-level selector composed of numerous voltage switches is a main factor for area overhead. Based on the actual layout size of peripheral circuitry in contemporary NFM [14], the total overhead for DW-NFM is about 0.8%.[3]

As mentioned in Section III, doubling page buffers and x-gating circuitry are needed for the segmented subplanes in SDW-NFM architecture. The dedicated row address decoder and WL-level selector are additionally required as many as the number of subplanes. Therefore, its overhead increases by 14% compared to DW-NFM architecture.

In addition, the estimated cost overhead of DW-NFM and SDW-NFM is 0.9% and 16.2%, respectively, over CNFM, using the formula for cost per die [27].

For comparison, we define the normalized performance gain (NPG) as follows:

$$\mathrm{NPG} = \frac{\mathrm{Throughput}}{\mathrm{Area}} \tag{1}$$

This metric intends to compare throughput improvements per unit area. As shown in Fig. 9, the NPG of one-way 64-GB DW-NFM is 1.2 and one-way 64-GB SDW-NFM is 1.25. On the other hand, the NPG is 1.18 if we choose two-way 32-GB CNFM.[4] This demonstrates that the proposed DW-NFM and SDW-NFM architecture shows comparable performance improvement with reasonable overhead with respect to the number of ways, which is beneficial in high density NFM where the number of ways is constrained.

*4) Timing Overhead:* There is very little software timing overhead for the proposed scheduling scheme. It mainly performs queue searching and request pairing. This overhead is translated into hundreds of nanoseconds when we assume the clock speed of an embedded processor in the device controller is 400 MHz [28]. It is under 1%, which is insignificant considering performance gain from the proposed scheme.

[3] The size of each functional block in CNFM was obtained from [14]. Based on this information, we compare the circuitry layout of CNFM, DW-NFM, SDW-NFM using PARQUET [26], which is a simulated annealing-based floorplanner.

[4] Adding a way requires a page buffer and peripheral circuitry additionally. This overhead is about 25% as shown in Fig. 9.

*5) Power Overhead:* In [14], NFM cell array and voltage generation circuitry account for 45% and 50% of total power consumption, respectively.[5] The remaining peripheral circuitry accounts for only 5%. The additional circuitry in the proposed NFM architecture is part of the remaining peripheral circuitry and its power consumption is negligible. On the other hand, we confirm that power consumption increases when multiple pages are activated due to multiplane operations. As static power consumption is smaller by several orders than dynamic power consumption, we focus on analyzing dynamic power consumption. DW-NFM shows roughly the same power consumption for multiplane operations as CNFM because two planes in both architectures can be simultaneously accessed. As the exploitation ratio of multiplane operations in DW-NFM is higher than CNFM, the total power consumption is increased by 20.6%, 15.3%, and 14.8% for one, two, and four-way DW-NFMs, respectively, but the energy consumption is same as or smaller than CNFM considering the throughput improvement. In SDW-NFM, the number of activated planes varies from 1 to 8 due to the plane segmentation, whereas 1 to 2 in CNFM and DW-NFM. The total power consumption is increased by 38.7%, 35.9%, and 29.5% for one-, two-, and four-way SDW-NFMs, respectively, if all subplanes are activated simultaneously ($N_{\text{act}} = 8$).[6] If $N_{\text{act}}$ is 2, the total power consumption is reduced by 43.5% compared to $N_{\text{act}} = 8$. Similarly, the total power consumption with $N_{\text{act}} = 4$ is reduced by 28.1%. Since all subplanes are not always activated simultaneously, we expect the average power consumption will be smaller. In addition, we can control the subplane parallelism (maximum of $N_{\text{act}}$) flexibly subject to the performance and power constraint of a storage device, thanks to more selectable planes than CNFM. Similar to DW-NFM, there is no increase in energy consumption as throughput increases. Current shows similar increase as it is proportional to the power consumption.

## V. Conclusion

As a result of developing cost-saving techniques such as MLC and 3-D NFM, the bit density has been continuously increasing. Meanwhile, the performance is degraded because channel-/way-level parallelisms are reduced for given storage capacity in high density NFM.

To address this problem, we focus on the plane-level parallelism that has not been thoroughly studied due to the architectural constraint. In this brief, we propose a novel NFM architecture that allows the decoupled WL selection for multiplane operations within the mated planes and propose the segmented plane NFM architecture that aggressively exploits subplane-level parallelism and reduces the latency of NFM. In addition, we propose the page-type-aware scheduling to optimally exploit the proposed NFM architecture.

The simulation results demonstrate that DW-NFM architecture presents the improvement of 21.3% for one way, 16.9% for two way, and 15% for four way on average. SDW-NFM architecture shows the improvement of 43.9%, 39.5%, and 34%, respectively. The required overhead is quite reasonable if it provides the same or better performance than adding a way, which is more costly.

Our proposed method will be more beneficial as the market moves toward high capacity and high density NFMs.

## References

[1] S. Aritome, "Nand flash innovations," *IEEE Solid State Circuits Mag.*, vol. 5, no. 4, pp. 21–29, Feb. 2013.

[2] J. Elliott and B. Brennan, "Industry innovation with samsung's next generation V-Nand," in *Proc. Flash Memory Summit*, 2014. [Online]. Available: https://www.flashmemorysummit.com/English/Collaterals/Proceedings/2014/20140805_Keynote2_Samsung.pdf

[3] Y. Hu, H. Jiang, D. Feng, L. Tian, H. Luo, and C. Ren, "Exploring and exploiting the multilevel parallelism inside SSDs for improved performance and endurance," *IEEE Trans. Comput.*, vol. 62, no. 6, pp. 1141–1155, Jun. 2013.

[4] S.-Y. Park, E. Seo, J.-Y. Shin, S. Maeng, and J. Lee, "Exploiting internal parallelism of flash-based SSDs," *IEEE Comput. Archit. Lett.*, vol. 9, no. 1, pp. 9–12, Jan. 2010.

[5] M. Jung, E. H. Wilson, and M. Kandemir, "Physically addressed queueing (PAQ): Improving parallelism in solid state disks," in *Proc. 39th Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2012, pp. 404–415.

[6] A. Tavakkol, P. Mehrvarzy, and H. Sarbazi-Azad, "TBM: Twin block management policy to enhance the utilization of plane-level parallelism in SSDs," *IEEE Comput. Archit. Lett.*, vol. 15, no. 2, pp. 121–124, Dec. 2016.

[7] R. Micheloni, L. Crippa, and A. Marelli, *Inside NAND Flash Memories*. New York, NY, USA: Springer, 2010.

[8] K. Park, "Memory controller, memory system including the same, and method for operating the same," U.S. Patent 12 777 676, May 15, 2015.

[9] F. Chen, R. Lee, and X. Zhang, "Essential roles of exploiting internal parallelism of flash memory based solid state drives in high-speed data processing," in *Proc. IEEE 17th Int. Symp. High Perform. Comput. Archit.*, Feb. 2011, pp. 266–277.

[10] Y. Hu, H. Jiang, D. Feng, L. Tian, H. Luo, and S. Zhang, "Performance impact and interplay of SSD parallelism through advanced commands, allocation strategy and data granularity," in *Proc. Int. Conf. Supercomput.*, May 2011, pp. 96–107.

[11] J.-U. Kang, J.-S. Kim, C. Park, H. Park, and J. Lee, "A multi-channel architecture for high-performance NAND flash-based storage system," *J. Syst. Archit.*, vol. 53, no. 9, pp. 644–658, Sep. 2007.

[12] J.-W. Hsieh, H.-Y. Lin, and D.-L. Yang, "Multi-channel architecture-based FTL for reliable and high-performance SSD," *IEEE Trans. Comput.*, vol. 63, no. 12, pp. 3079–3091, Dec. 2014.

[13] K.-T. Park *et al.*, "Three-dimensional 128gb MLC vertical NAND flash-memory with 24-WL stacked layers and 50mb/s high-speed programming," in *Proc. Int. Solid-State Circuits Conf.*, Jun. 2014, pp. 334–336.

[14] J.-W. Im *et al.*, "A 128Gb 3b/cell V-NAND flash memory with 1Gb/s I/O rate," in *Proc. Int. Solid-State Circuits Conf.*, Jan. 2015, pp. 130–132.

[15] D. Kang *et al.*, "256gb 3b/cell v-NAND flash memory with 48 stacked WL layers," in *Proc. Int. Solid-State Circuits Conf.*, Jun. 2016, pp. 130–132.

[16] C. Kim *et al.*, "A 512gb 3b/cell 64-stacked WL 3d v-NAND flash memory," in *Proc. Int. Solid-State Circuits Conf.*, Sep. 2017, pp. 202–204.

[17] S. Lee *et al.*, "A 1tb 4b/cell 64-stacked-WL 3d NAND flash memory with 12mb/s program throughput," in *Proc. Int. Solid-State Circuits Conf.*, Apr. 2018, pp. 340–342.

[18] M. Abraham, "Architectural considerations for optimizing SSDS," in *Proc. Flash Memory Summit*, 2014, pp. 45–65.

[19] T. Parnell and R. Pletka, "Nand flash basics error characteristics: Why do we need smart controllers," *Flash Memory Summit*, vol. 3, nos. 2–4, pp. 3–4, 2017.

[20] J.-Y. Kim, S.-H. Park, H. Seo, K.-W. Song, S. Yoon, and E.-Y. Chung, "NAND flash memory with multiple page sizes for high-performance storage devices," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 2, pp. 764–768, Feb. 2016.

[21] A. Gupta, Y. Kim, and B. Urgaonkar, "Dftl: a flash translation layer employing demand-based selective caching of page-level address mappings," in *Proc. 14th Int. Conf. Architectural Support Program. Lang. Operating Syst.*, Feb. 2009, pp. 229–240.

[22] A. Kawaguchi, S. Nishioka, and H. Motoda, "A flash-memory based file system," in *Proc. USENIX Winter Tech. Conf.*, Sep. 1995, pp. 155–164.

[23] (2007). *U Mass Trace Repository*. [Online]. Available: http://traces.cs.umass.edu/

[24] (1997). *Storage Networking Industry Association*. [Online]. Available: http://iotta.snia.org/

[25] D. Landsman and D. Walker, (2013). *AHCI and NVMe as Interfaces for SATA Express–Devices*. [Online]. Available: https://sata-io.org/

[26] S. N. Adya and I. L. Markov, "Fixed-outline floorplanning: Enabling hierarchical design," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 6, pp. 1120–1135, Dec. 2003.

[27] D. A. Patterson and J. L. Hennessy, *Computer Organization Design: The Hardware/Software Interface*. Burlington, MA, USA: Morgan Kaufmann, 2013.

[28] S. Kim, H. Oh, C. Park, S. Cho, and S.-W. Lee, "Fast, energy efficient scan inside flash memory SSDS," in *Proc. Int. Worksope Accelerating Data Manage. Syst.*, 2011, pp. 45–65.

[5] The power consumption ratio of each functional block in CNFM was obtained from [14].

[6] $N_{\text{act}}$ means the number of activated subplanes.