Write-awareness prefetching for non-volatile cache in energy-constrained IoT device

Mao Ni1, 2, 3, Lan Chen1, 3, a), Xiaoran Hao1, 3, Chenji Liu1, 2, 3, Yiheng Zhang1, 3, and Lei Pan2

Abstract Large-sized cache is beneficial to improve CPU performance especially for IoT applications with huge amount of data. However, large-sized SRAM cache also increases chip area and energy which is not friendly to resources limited IoT terminals. The STT-RAM has high storage density and near zero leakage is regarded as an ideal technology to replace SRAM. Prefetching is a vital method to hide the access latency of off-chip memory. Nevertheless, traditional prefetchers for SRAM is inadequate for MRAM cache with read-write asymmetry. Aggressive prefetching for STT-RAM cache would cause cache congestion and dynamic energy rise because of STT-RAM long write latency and high write energy. In response to the above problems, this paper novelty proposes WANCP (Write-awareness Adaptive Non-volatile Cache Prefetch), which adaptively adjusts the prefetch aggressiveness according to the saturation of MSHR (Miss-status Handling Registers) in the L2 cache. Experiments show that, for applications that are sensitive to L2 cache capacity, the CPU performance with STT-RAM L2 cache can be improved by up to 33.2% and 10.9% on average compared to the same sized SRAM L2 cache. The proposed WANCP can further improve the CPU performance 0.4% on average, and reduce the prefetch energy by 9.4% on average.

Keywords: L2 cache, NVM, prefetcher, STT-RAM

1. Introduction

With the current trend of technology scaling, large-sized cache is needed to accommodate the huge amount of data and to reduce the latency of off-chip memory accesses [1]. Large-sized SRAM cache can effectively hide the access delay of off-chip storage, but its high leakage power and low storage density increase the chip area and energy [2]. The emerging Non-volatile memory (NVM), such as PCM, RRAM, STT-RAM, etc., have advantages of near-zero leakage energy, high storage density, and high read-write speed. Among them, STT-RAM with excellent durability, fast access speed, and CMOS compatibility [3] is regarded as a promising replacement for the traditional SRAM cache.

However, STT-RAM has the disadvantages of long write latency and high write energy [4, 5], which have negative impact on performance and energy of processors. Some scholars [6, 7] reduce the STT-RAM write latency by upsizing the write access transistor or using differential bits at the STT-RAM circuit level. Chen and Wang [8, 9] propose a heterogeneous hybrid cache composed of STT-RAM and SRAM. Park and Sayed [10, 11] propose a hybrid cache composed of STT-MRAM and semi-volatile STT-RAM. Korgaonkar [12] adjusts the bypass ratio according to the cache congestion caused by its long write latency.

For SRAM cache, the coverage of prefetch has always been an important indicator of prefetch algorithm [13, 14]. However, high coverage prefetch also amplifies the impact of STT-RAM’s long write latency and high write energy. To solve the above problems, this paper deeply analyzes the impact of traditional prefetch strategies on STT-RAM L2 cache and propose a novel prefetch method named WANCP, which improves the processor performance and decreases the energy effectively. The contributions of this article are as follows:

• For applications that are sensitive to L2 cache capacity, under the same cache area, the STT-RAM L2 cache can improve the CPU performance by up to 33.2% and 10.9% on average compared to CPUs with SRAM L2 cache.

• Experiments show that, although it brings the improvement of cache performance, aggressive prefetch greatly increase STT-RAM cache write operations which causes the cache congestion and amplifies the impact of its high write energy and long write latency.

• We propose WANCP to adaptively adjust the prefetch density according to the MSHR saturation and the proportion of read miss and prefetch requests in the MSHR. WANCP can effectively reduce dynamic energy of cache write and alleviate cache congestion caused by long write latency of the non-volatile cache.

2. Related works

In order to meet the ever-increasing demand for computing power and low power of IoT device, high storage density, near zero leakage power NVM (PCM, STT-RAM, RRAM, etc.) has been studied and tried at different levels from external memory to main memory [15, 16, 17]. These methods of solve the asymmetry of NVM are very valuable. There are also a lot of works [8, 9, 10, 11, 12, 18, 19, 20, 21, 22] focusing on the research of STT-RAM-based cache. Komalan [22] obtains a smaller on-chip area and lower energy by adding a buffer between the CPU and L1 D-cache at the expense of 8% performance loss. Sayed [11] changes the data...
retention time of STT-RAM to reduce its write latency and dynamic energy, and proposed a hybrid L1 D-cache composed of semi-volatile and non-volatile STT-RAM. Chen, Wang and Korgaonkar [8, 9, 12] propose a hybrid L2/LLC cache solution composed of SRAM and STT-RAM, in which the frequently read data is stored in STT-RAM, and the frequently write data is stored in SRAM to make good use of STT-RAM and SRAM.

The above efforts aim to solve the problems of STT-RAM read-write asymmetry and lifespan etc. In addition, the research on the prefetch of large-capacity cache is also very valuable for CPU performance improvement. Lai [23] proposes a trace-based predictor to replace dead blocks in L1 cache with prefetched data. Bhatia [24] introduces Perception-based Prefetch Filtering (PPF) as the way to increase the coverage of prefetch strategy. However, these prefetch strategy did not comprehensively consider STT-RAM read-write asymmetry. Therefore, these paper [23, 24, 25, 26] focus on the prefetch strategy for the STT-RAM L2 cache.

3. Background and motivation

The L2 cache shown in Fig. 1 is vital to the CPU performance. Compared with the SRAM cache, although the same sized STT-RAM can improve the performance of processors because of its high density, STT-RAM long write latency and high write energy have negative effect on processors.

This section will evaluate the performance of the single-core processor with STT-RAM L2 cache in the gem5 simulator [27]. Gem5 is a cycle-accurate simulator and provides an accurate cache model. We add STT-RAM write latency parameter to gem5 according to [7, 8]. Table I shows the configuration parameters of gem5 and read latency of STT-RAM and SRAM.

We use SPEC CPU2006 [28] as our experiment benchmarks. While these benchmarks were originally intended for desktop and server systems, they exhibit wide range of architecture and cache behavior expected for IoT terminals. We selected 10 benchmarks from SPEC CPU2006, which cover different cache access behaviors as much as possible. We run benchmarks for 800 million instructions to ensure that the occupancy rate of L2 cache is beyond 99% on average.

Table I System configurations.

| Parameter | Value                      |
|-----------|----------------------------|
| CPU       | 0.32GHz                    |
| L1 Cache  | 32KB iCache, 64KB dCache, 64KB line size, 2-way set associative, R/W delay 2 cycles |
| L2 Cache  | 512KB line size, 8-way set associative, read 4 cycle, write 4-20-30 cycles, cache size 256KB-1024KB, MSHR=18 |
| Test bench| spec CPU2006                |

3.1 Processor’s performance with large capacity STT-RAM L2 cache

KorgaonkarK [12] proposes that the capacity of STT-RAM is 3 times that of SRAM when the STT-RAM write latency is 10 ns, and it can be 5 times when the latency is 30 ns. We suppose that the STT-RAM write latency is 15 ns, and the STT-RAM capacity is about 4 times that of SRAM. Fig. 2 shows that the processor performance can be effectively improved when the capacity of SRAM cache is increased from 256 KB to 1024 KB without prefetch. The performance improvement is obvious especially for applications sensitive to L2 cache capacity, such as leslie, omnetpp, xalancbmk, bzip2.inputsource, and bzip2.control. And the performance can be improved by up to 43.3%. So increasing the L2 cache capacity can improve the terminal processor performance distinctly. However, due to the large leakage power and low storage density of SRAM, increasing the cache capacity also increases the area and power consumption of the processor. Therefore, STT-RAM with high storage density and low leakage energy has its potential advantages to construct a large-capacity cache. It also can be seen from Fig. 2 that when we replace 256KB SRAM with 1 MB STT-RAM, most of the applications can still achieve a performance improvement by up to 38%. However, for applications such as milc and sjeng which are not sensitive to L2 cache capacity, the IPC performance drop slightly. So we’ll further analyze the impact of different write latency on the IPC performance with STT-RAM L2 Cache.

3.2 Impact of write latency on L2 cache performance

In order to analyze the impact of long write latency on IPC performance, we assume 4 different STT-RAM write latency
of 2 ns, 10 ns, 15 ns, and 30 ns. And the STT-RAM read latency is considered to be the same as SRAM. It can be seen from Fig. 3 that compared to the 1 MB SRAM, when the write latency increases, the IPC performance shows a significant decrease. When the write latency is 30 ns, there is 32% performance down. So, the benefits of STT-RAM’s high storage density are gradually being swallowed as the write latency increases. Therefore, how to reduce the number of STT-RAM cache write is the vital problem needs to be resolved in STT-RAM based cache architecture.

3.3 Write sources on the STT-RAM L2 cache

Based on the above analysis, the long write latency has a serious impact on the STT-RAM cache performance, so analyzing the write sources in the L2 cache is very valuable to improve the cache performance. Cache write is mainly composed of prefetch write, core write, and demand write [8]. Prefetch write is writing data blocks into the cache in advance caused by cache prefetch mechanisms. Core write is CPU writing back during CPU run-time. And demand write is the cache replacement caused by write miss. In Fig. 4, “others” is the sum of core write and demand write. It can be seen from Fig. 4 that the prefetch data accounts for more than 20% of all cache data on average, and 44% the most. Obviously, prefetch write is an important source of L2 cache write. A large number of prefetch write will amplify the negative effects of STT-RAM’s long write latency and high write energy. We’ll focus on the impact of classic prefetch strategies on STT-RAM L2 cache performance in 3.4 section.

3.4 Classic prefetch strategy on STT-RAM L2 cache

According to Fig. 5, although prefetch amplifies the negative impact of STT-RAM long write latency and high write energy, the Stream prefetch strategy still improves cache performance by up to 24.9% with the configurations of P3 in Table II. Therefore, it is valuable to optimize the classic prefetch scheme to adapt to the read-write asymmetry of STT-RAM for better performance.

3.5 STT-RAM cache performance and prefetch energy

Fig. 6 shows that as the prefetch aggressiveness increases from P1 to P3 in Table II, the cache performance gradually improves for leslie, sjeng, bzip2 and other access-intensive applications. However, for GemsFETD, h264ref, etc., aggressive prefetch cannot improve the IPC performance. It can be seen from Fig. 7 that the prefetch energy increases for all applications as the prefetch aggressiveness increases, and the maximum increase is nearly 600%. The classic prefetch strategy doesn’t take into account the effects of STT-RAM long write latency and high write energy. So it’s necessary to optimize the classic prefetch strategy for the STT-RAM cache.

4. Write-awareness adaptive non-volatile cache prefetch

Stream prefetcher is a popular prefetching strategy for L2 cache. Several state-of-the-art prefetcher FDP [29], NST [30] developed from stream prefetcher effectively improve SRAM cache performance. However these solutions didn’t consider STT-RAM poor write performance which leads to redundant write operations and more energy consumption. In order to reduce the prefetch energy of STT-RAM L2 cache, we propose a stream-based WANC mechanism according to the MSHR saturation and the ratio of read miss
4.1 MSHR saturation monitoring

4.1.1 Introduction of MSHR

MSHR [31] is a hardware structure that keeps track of all in-flight memory requests. When the cache hits, the requested data will be directly obtained from the cache, and when the cache misses, it will check whether the same cache block is being serviced by an earlier memory request. If it is, discard it. Otherwise, MSHR entry will be allocated for the new request. The prefetch request triggered by each CPU access is also placed in the MSHR as shown in Fig. 8. And the requests in the MSHR queue are sent to the bus sequentially.

4.1.2 MSHR saturation

We assume the depth of MSHR queue is 18. As shown in Fig. 9, the saturation peak of MSHR with P3 significantly increases for all applications compared to non-prefetch. Since the requests in MSHR are sent to the bus sequentially, the increase of the saturation peak would increase the average waiting time of read miss in the queue as shown in Fig. 8. Therefore, aggressive prefetching will affect the response time of read misses.

Due to STT-RAM long write latency, the access request of STT-RAM L2 cache is prone to congestion, which will be aggravated by a large amount of prefetched data. Cache congestion would delay CPU read requests which leads to CPU performance degradation. This can be confirmed from Fig. 10 that the maximum of MSHR usage, compared with SRAM, is slightly lower for the same application (STT-RAM cache receives fewer read requests compared with SRAM cache within the same time.). Therefore, when the MSRH tends to saturate, reducing the prefetch density can not only reduce the dynamic write energy of the STT-RAM cache, but also obtain certain performance benefits.

4.2 WANCP algorithm

Table II lists the prefetcher configurations with different prefetch aggressiveness. The larger the prefetch degree and prefetch distance are, the more aggressive the prefetch strategy is. Numerous experiments show that when the prefetch degree is greater than 32, the cache performance improvement is negligible, and the performance evened decreases significantly for some applications. In addition, the large prefetch distance would cause more cache block prefetch which make the useful cache data be prematurely replaced. Similarly, too large prefetch degree would also increase the cache pressure and main memory bandwidth [29].

In response to the above problems, our previous work proposed ANCP [32] which dynamically adjusts prefetch strategy according to the level of saturation in MSHR. However, this method has a large granularity for prefetch. It only considered the MSHR saturation, but ignored the ratio of read miss and prefetch in MSHR. Based on ANCP, we propose the fine-grained and more flexible WANCP to further reduce the dynamic energy of the STT-RAM cache.

4.2.1 Implementation of WANCP

WANCP adaptively adjusts the prefetch configurations of Stream algorithm according to the current MSHR saturation and the proportion of prefetch and read miss requests in the MSHR. MSHR saturation is tracked by the mshr_allocated register. When an entry of MSHR is allocated to read miss or prefetch request, the mshr_allocated is incremented by 1, and when the request is sent from the MSHR to the bus, it is decremented by 1. One bit added to each entry of MSHR identifies the request is read miss or prefetch. 1 means a prefetch request, and 0 means a read miss request.

A register allocated_pf is used to count the number of prefetch requests in the MSHR. When CPU access triggers MSHR request, if it is prefetch request, allocated_pf is increased by 1, and if the prefetch request is sent to the bus or is replaced by a read miss request, allocated_pf is decreased by 1. The algorithm is shown in Fig. 11. We found that when MSHR saturation exceeds 28% and the number of read miss in MSHR is greater than 5, we should choose P1 as the prefetch strategy, when MSHR saturation is less than 28% we should choose P3 prefetch strategy, otherwise we should choose P2 prefetch strategy. Here the saturation and read miss thresholds comes from large quantities of experiments for different types of applications. Limited to the length of this article, it will not
be analyzed detailed.

4.2.2 Hardware overhead

We assume that the MSHR has 18 entries, WANCP needs 5 bits `mshr_allocated` and 5 bits `allocated_Pf` registers. In addition, each entry has one identification bit which totaling 18 bits. So for the MSHR with n entries, the total hardware overhead \( n + 2 \lceil \log_2 n \rceil \) bits. Compared with the state-of-the-art FDP [29], its hardware overhead is less than 1%.

5. Simulation results

5.1 Prefetch energy with WANCP

From Fig. 12 we can see that compared with the Stream prefetcher with P3 configurations, ANCP can reduce the prefetch energy, and the fine-grained WANCP can further reduce the prefetch energy by 9.4% on average, and 20% the most. We also can see in Fig. 13, the average proportion of prefetched data in the L2 cache also decreases with WANCP which is better than ANCP. WANCP not only can reduce the prefetch energy, but also improve the cache efficiency, which is valuable for cache capacity sensitive applications.

5.2 Processor performance with WANCP

It can be seen from Fig. 14, compared with the ANCP, the IPC performance with WANCP is slightly improved on average, and compared with P3, the performance improvement is between −0.9% and 1.5%. For applications sensitive to L2 cache capacity, such as leslie, onnetpp, xalancbmk, etc., we can see that when we replace 256KB SRAM with 1024KB STT-RAM, the IPC performance with P3 increases by 10.9% on average, and 33.2% the most; and the IPC performance with WANCP increases by 11.3% on average, and 34.9% the most as shown in Fig. 15. The above analysis shows that for applications sensitive to L2 cache capacity, WANCP not only reduces the write energy of STT RAM cache, but also increases the CPU IPC by an average of 0.4%.

6. Conclusions

To satisfy the requirements of high performance and low power for IoT device, we introduce STT-RAM into the cache system of processors and evaluate the performance of the processor with STT-RAM L2 cache in the gem5 simulator. Experimental results show that, for the out-of-order processors, if we replace SRAM L2 cache with the same sized STT-RAM L2 cache, the average performance is improved by 10.9% for applications sensitive to cache capacity. We first find that for the STT-RAM cache, although the best-performing stream-based P3 can improve the processor performance, the prefetch energy also increases obviously because of STT-RAM’s high write energy. In addition, a large number of prefetch write increase the congestion L2 cache, which have adverse effects on the processor performance. To make better use of STT-RAM, we propose WANCP to reduce the amount of prefetch write on STT-RAM. Compared with the best-performing P3 prefetcher, WANCP reduces the prefetch energy by 9.4% on average, and obtains up to 1.5% performance improvement of the processor.
This article focuses on the applicability of classic prefetch algorithms to STT-RAM cache. These conclusions are meaningful to the application of traditional prefetch algorithm with coverage priority on the non-volatile cache. Our research work now is only under the single core system. But for out-of-order multi-core processors, the congestion of shared non-volatile cache will be more serious. The next we will focus on the prefetch algorithm in the multi-core processor with STT-RAM cache of the smart terminal.

Acknowledgments

This research was funded by the National Key R&D Program of China under Grant 2019YFB2102400.

References

[1] K. Namba and F. Lombardi: “On coding for endurance enhancement and error protection of phase change memories with write latency reduction,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 26 (2018) 230 (DOI: 10.1109/tvlsi.2017.2766362).
[2] P.K.D.Pramanik, et al.: “A cross-layer adaptive approach for performance and power optimization in STT-MRAM,” DATE (2018) 10.23919/ date.2018.8342114.
[3] K. Kuan and T. Adegbiye: “A study of runtime adaptive prefetching for STT-RAM L1 caches,” ICCD (2020) (DOI: 10.1109/ICCD50377.2020.00051).
[4] J. Ahn, et al.: “DASCA: Dead write prediction assisted stramcache architecture,” HPCA (2014) (DOI: 10.1109/hpca.2014.6835944).
[5] N. Sayed, et al.: “An energy-efficient and fast scheme for hybrid STT-RAM caches,” Proc. Great Lakes Symp. VLSI (2020) 487 (DOI: 10.1145/3386263.3406951).
[6] M.Shihab, et al.: “Adaptive placement and migration policy for high-performance many-core processors,” PACT (2018) 1 (DOI: 10.1109/pact.2018.8241516).
[7] H. Noguchi, et al.: “An energy-efficient and fast scheme for hybrid STT-RAM caches,” ACM J. Emerging Technologies in Computing Systems 17 (2021) 1 (DOI: 10.1145/3423135).
[8] N. Sayed, et al.: “Dynamic behavior predictions for fast and efficient hybrid STT-RAM caches,” ACM J. Emerging Technologies in Computing Systems 17 (2021) 1 (DOI: 10.1145/3423135).
[9] Z. Wang, et al.: “Feedback directed prefetching: improving the performance and bandwidth-efficiency of hardware prefetchers,” (2006).
[10] K. Kuan, et al.: “Energy-efficient runtime adaptable L1 STT-RAM cache design,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. 2021 (DOI: 10.1109/access.2019.2958684).
[11] N. Sayed, et al.: “A novel prefetching scheme for non-volatile cache in the AIoT processor,” 2020 5th Intern. Conf. Univ. Village (UV) (2020) (DOI: 10.1109/U50937.2020.9426214).
[12] K. Kuan, et al.: “Dimming hybrid caches to assist in temperature control of chip multiprocessors,” Proc. Great Lakes Symp. VLSI (2020) 487 (DOI: 10.1145/3386263.3406951).
[13] N. Sayed, et al.: “Dynamic behavior predictions for fast and efficient hybrid STT-RAM caches,” ACM J. Emerging Technologies in Computing Systems 17 (2021) 1 (DOI: 10.1145/3423135).
[14] K. Kuan, et al.: “Energy-efficient runtime adaptable L1 STT-RAM cache design,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. 39 (2020) 1328 (DOI: 10.1109/TCAD.2019.2912920).
[15] D. Gajaria and T. Adegbiye: “ARC: DVFS-aware asymmetric-retention STT-RAM caches for energy-efficient multicore processors,” Proc. Internat. Symp. Memory. Memory. (2019) 439 (DOI: 10.1145/3357526.3357553).
[16] M.P. Komalan, et al.: “System level exploration of a STT-RAM based level 1 data cache,” DATE (2015) (DOI: 10.7873/date.2015.0551).
[17] A. Lai, et al.: “Dead-block prediction & dead-block correlating prefetchers,” Proc. 28th Annual Internat. Symp. Comp. Archit. 29 (2001) (DOI: 10.1109/isca.2001.973445).
[18] E. Bhatia, et al.: “Perception-based prefetch filtering,” ISCA (2019) 1 (DOI: 10.1145/3307650.3322207).
[19] S. Palacharla and R.E. Kessler: “Evaluating stream buffers as a secondary cache replacement,” Proc. 21 Internat. Symp. Comp. Archit. (1994) (DOI: 10.1109/ISCA.1994.288164).
[20] P. Michaud, “Best-offset hardware prefetching,” HPCA (2016) (DOI: 10.1109/HPCA.2016.7446087).
[21] N. Binkert, et al.: “The gem5 simulator,” SIG ARCH Comput. Archit. News 39 (2011) 1 (DOI: 10.1145/2024716.2024718).
[22] SPEC CPU 2006 Benchmarks, http://www.spec.org/cpu2006.
[23] S. Srinath, et al.: “Feedback directed prefetching: improving the performance and bandwidth-efficiency of hardware prefetchers,” (2006).
[24] S. Srinath, et al.: “Dead-block prediction & dead-block correlating prefetchers,” Proc. Great Lakes Symp. VLSI (2020) 487 (DOI: 10.1145/3322207).
[25] K. Kuan, et al.: “Near-side prefetch throttling: adaptive prefetching for high-performance many-core processors,” PACT (2018) 1 DOI: doi.org/10.1109/pact.2018.8241516.
[26] K. David: “Lockup-free instruction fetch/prefetch cache organization,” 25 years of the international symposia on Computer architecture (selected papers) (1998) 20 (DOI: 10.1145/285930.285939).
[27] N. Binkert, et al.: “Adaptive placement and migration policy for an STT-RAM based hybrid cache,” HPCA (2014) (DOI: 10.1109/hpca.2014.6835933).
[28] SPEC CPU 2006 Benchmarks, http://www.spec.org/cpu2006.
[29] N. Binkert, et al.: “A cross-layer adaptive approach for performance and power optimization in STT-MRAM,” DATE (2018) 10.23919/ date.2018.8342114.
[30] M. Shihab, et al.: “Costure: tailoring STT-RAM for persistent main memory,” 4th Workshop Interact. NVM/Flash Operat. Sys. Workloads (2016).
[31] H. Noguchi, et al.: “3.2 Mb STT-MRAM-based cache with memory-access-aware power optimization and write-verify-write/read-modify-write scheme,” ISSCC (2016) (DOI: 10.1109/isscc.2016.7417942).
[32] Q. Chen: “Design and optimization of hybrid cache based on SRAM and STT-MRAM,” [D].
[33] Z. Wang, et al.: “Adaptive placement and migration policy for an STT-RAM based hybrid cache,” HPCA (2014) (DOI: 10.1109/hpca.2014.6835933).
[34] J. Park, et al.: “M2H cache: a multi-retention STT-RAM-based low-power last-level cache for mobile hardware rendering systems,” ACM Trans. Arch. Code Optimization 16 (2019) 1 (DOI: 10.1145/3328520).
[35] N. Sayed, et al.: “Compiler-assisted and profiling-based analysis for fast and efficient STT-MRAM on-chip cache design,” ACM Trans. Des. Auto. Electron. Sys. 24 (2019) 1 (DOI: 10.1145/33221693).
[36] K. Korgaonkar, et al.: “Density tradeoffs of non-volatile memory as a replacement for SRAM based last level cache,” ISCA (2018) (DOI: 10.1109/ISCA.2018.00035).
[37] R. Bera, et al.: “DSPatch: dual spatial pattern prefetcher,” Proceedings of the 52nd Annual IEEE/ACM Internat. Symp. Microarchitecture (2019) 531 (DOI: 10.1145/3352460.3358325).
[38] Y. Ishii, et al.: “Access map pattern matching for high performance data cache prefetch,” J. Instruction-Level Parallelism (2011).
[39] L. Wang, et al.: “Fine-grained data management for DRAM/SSD hybrid main memory architecture,”IEICE Trans. Inf. & Syst. E99-D (2016) 3172 (DOI: 10.1587/transinf.2016ed8105).
[40] H. Sun, et al.: “An energy-efficient and fast scheme for hybrid storage class memory in an AIoT terminal system,” Electronics 9 (2020) 1013 (DOI: 10.3390/electronics9061013).
[41] C. Liu, et al.: “Fast cacheline-based data replacement for hybrid DRAM and STT-RAM main memory,” IEICE Electron. Express 17 (2020) 20200090 (DOI: 10.1587/elex.17.20200090).