사물인터넷융복합논문지 Vol. 10, No. 2, pp. 109-114, 2024 https://doi.org/10.20465/KIOTS.2024.10.2.109

# 효율적인 자원관리를 위한 SSD 전용 RAID 시스템 설계

**이현섭***
**백석대학교 컴퓨터공학부 교수**

# A Design of SSD Dedicated RAID System for Efficient Resource Management

Hyun-Seob Lee*
Professor, Division of Computer Engineering, Baekseok University

**요 약** 높은 데이터 신뢰성을 요구하는 엔터프라이즈 저장시스템은 데이터 손실 및 장애 시 복구를 위해 RAID (Redundant Array of Independent Disks) 시스템을 적용하고 있다. 특히 RAID 5는 패리티를 여러 저장장치에 분산 저장하여 공간 효율성과 안정성을 보장하였다. 그러나 저장장치의 용량이 서로 다를 경우 가장 작은 용량의 저장장치 기준으로 RAID가 구축되어 저장공간의 낭비가 발생한다. 따라서 이러한 자원관리 문제를 해결하기 위한 연구가 필요하다. 본 논문에서는 SSD(Solid State Disk)로 구성된 RAID에서 각각의 독립적인 NAND 플래시 메모리 블록을 내부뿐만 아니라 외부 SSD와 RAID 그룹을 묶는 방법을 제안한다. 이 방법은 SSD 내부의 블록 정보를 RAID 시스템에 전달하는 정책과 RAID 시스템으로부터 전달된 물리적인 주소를 RAID 그룹으로 묶는 정책으로 구분된다. 이 방법을 통해 서로 다른 용량의 SSD를 RAID로 묶을 때 자원의 낭비가 발생하지 않는 RAID를 유지할 수 있다. 마지막으로 실험을 통해 제안하는 방법의 효과를 증명한다.

**주제어 :** 낸드 플래시 메모리, 반도체 저장장치, 레이드, 저장장치, 내고장성

**Abstract** Enterprise storage systems that require high data reliability are applying RAID (Redundant Array of Independent Disks) systems to recover from data loss and failure. In particular, RAID 5 ensures space efficiency and reliability by distributing parity across multiple storage devices. However, when storage devices have different capacities, RAID is built based on the smallest capacity storage device, resulting in wasted storage space. Therefore, research is needed to solve this resource management problem. In this paper, we propose a method for RAID grouping of each independent NAND flash memory block in a RAID consisting of SSD (Solid State Disk) with external SSDs as well as internal SSDs. This method is divided into a policy for delivering block information inside SSDs to the RAID system and a policy for RAID grouping of physical addresses delivered from the RAID system. This method allows us to maintain a RAID that does not waste resources when SSDs of different capacities are grouped into RAID5. Finally, we demonstrate the effectiveness of the proposed method through experiments.

**Key Words :** Nand Flash Memory, SSD, RAID, Storage, Fault Tolerance

*This paper was supported by 2024 Baekseok University Research Fund
*교신저자 : 이현섭(hyunseob@bu.ac.kr)
접수일 2024년 03월 04일 수정일 2024년 03월 20일 심사완료일 2024년 04월 08일

## 1. 서론

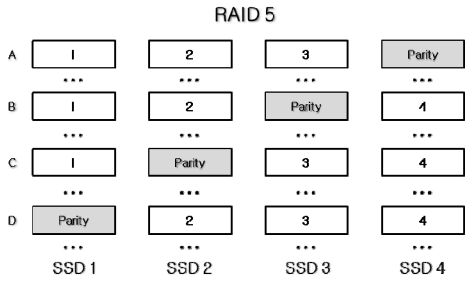
IoT(Internet of Things) 기술은 다양하고 고도화되어 소형 임베디드 시스템에서 대용량의 데이터를 처리하는 임베디드 시스템까지 다양한 분야에서 발전되고 있다. 최근에는 다양한 IoT 장비에서 수많은 데이터를 생성하여 기업의 데이터 수집 및 분석에 새로운 차원의 기술 환경을 제공하고 있다. 엔터프라이즈 환경에서는 이렇게 수집된 대량의 데이터를 안정적으로 저장하고 관리하기 위한 신뢰성 높은 고장회복 시스템을 적용하고 있다. 일반적인 엔터프라이즈 환경에서는 데이터 손실 및 장애로부터 데이터를 안정적으로 복구하기 위해 RAID (Redundant Array of Independent Disks)[1, 2] 시스템을 적용하고 있다. 특히 RAID 5[3-5]는 패리티를 여러 저장장치에 분산 저장하여 공간 효율성과 안정성을 보장하였다. 그러나 저장장치의 용량이 서로 다를 경우 가장 작은 용량의 저장장치 기준으로 RAID가 구축되어 저장공간의 낭비가 발생한다. 따라서 이러한 자원관리 문제를 해결하기 위한 연구가 필요하다. 본 논문에서는 RAID에서 SSD(Solid State Disk)를 구성하고 있는 독립적인 NAND 플래시 메모리 블록을 동일한 RAID 내에 이웃하는 SSD의 메모리 블록과 RAID 그룹을 구성하는 방법을 제안한다. 이 방법은 RAID 시스템이 SSD 내부의 블록 정보를 전달받아서, 이웃하는 SSD 혹은 단일 SSD 내부의 블록들을 RAID 그룹으로 묶는 정책을 적용하였다. 이 방법을 통해 서로 다른 용량의 SSD를 RAID 5로 묶을 때 자원의 낭비가 발생하지 않는 RAID를 유지할 수 있다. 마지막으로 실험을 통해 제안하는 방법의 효과를 증명한다.

논문의 구성은 다음과 같다. 2장에서는 RAID의 구조와 복구 방법을 알아보고 서로 다른 용량의 저장장치가 RAID에 묶였을 때 발생하는 자원 낭비를 설명한다. 3장에서는 RAID의 자원 낭비를 해결하기 위한 관련 연구를 분석한다. 4장에서는 논문에서 제안하는 SSD 전용 RAID 5 방법을 설명한다. 마지막으로, 시뮬레이션을 통해 제안하는 시스템의 성능과 효과를 측정한다.

## 2. 배경

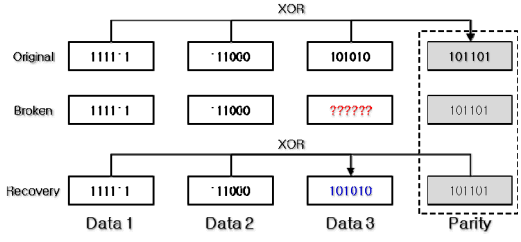
### 2.1 RAID 5의 특징

RAID는 여러 개의 저장장치를 하나의 논리적인 저장장치로 구성하여 데이터를 저장하는 방법이다. 그리고 논리적 저장장치를 구성하는 방법에 따라 여러 RAID의 종류가 있다.




[Fig. 1] RAID 5 Architecture

Fig. 1은 RAID 5의 구조를 보여주고 있다. 그림의 구조에서는 4개의 SSD를 하나의 논리적 저장장치로 묶었다. A 그룹에서는 4번 SSD에 패리티 데이터를 할당하였다. B 그룹에서는 3번 SSD에 패리티 데이터를 할당하였다. 이렇게 RAID 5는 논리적 그룹의 저장장치 중 1대를 할당하여 패리티 데이터를 저장한다.



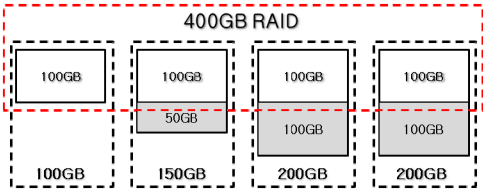

[Fig. 2] Step of RAID Recovery

Fig. 2는 RAID의 패리티 데이터 생성과 고장회복 과정을 보여주고 있다. RAID의 패리티 데이터는 XOR 연산을 통해 생성된다. 그림과 같이 데이터 1, 2, 3의 데이터 111111, 111000, 101010의 패리티는 XOR 연산을 통해 101101이 된다. 그리고 Data 3의 데이터를 손실했을 때 데이터 1, 2와 패리티 데이터를 XOR하여 데이터 3을 복구한다. 그림의 예제에서는 111111, 111000, 101101을 XOR하여 101010을 복구하였다.

### 2.2 RAID 5의 자원 낭비 문제

Fig. 3은 서로 다른 용량의 100GB, 150GB, 200GB, 200GB 저장장치를 RAID로 묶었을 때 발생하는 자원 낭비 문제를 보여주고 있다. RAID는 동일한 데이터 공간을 하나의 논리적인 그룹으로 묶어야 한다. 따라서 그

림과 같이 가장 작은 용량의 저장장치로 RAID 그룹이 구성된다. 그림의 예제에서는 총 650GB의 자원이 RAID로 묶였으나 그룹 내에 가장 작은 용량인 100GB 단위로 RAID가 구성된다. 따라서 400GB의 논리적인 저장장치가 생성되고 250GB의 자원은 활용되지 못하는 문제가 발생한다.

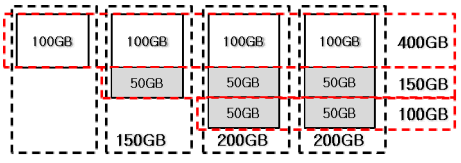



[Fig. 3] Problem of RAID

## 3. 관련연구

### 3.1 Hybrid RAID

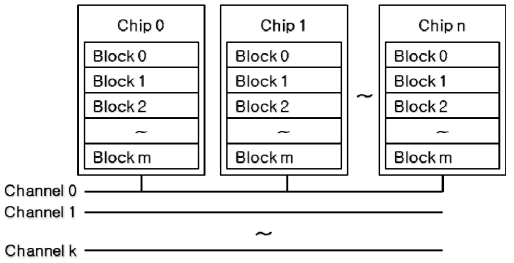
Fig. 4는 RAID의 자원 낭비를 줄이기 위한 Hybrid RAID[6-8] 기법을 보여주고 있다. 이 방법은 그림과 같이 RAID 내의 저장장치 중 최저 용량과 동일한 용량을 할당하여 논리적 저장장치 그룹을 만들고, 남아있는 용량을 별도의 RAID로 구성하는 방법이다. 그림의 예제에서는 먼저 100GB씩 4대를 묶어서 400GB의 RAID를 구성하였다. 그다음, 남은 용량 중 최저 용량인 50GB로 150GB의 RAID를 구성하였다. 마지막으로, 남아있는 50GB의 용량을 할당하여 100GB의 RAID를 구성하였다. 이 방법은 남는 자원을 활용하여 자원을 최적화한다. 그러나 디스크 구성에 따라 RAID 내부에서 서로 다른 타입의 RAID를 구축하여 관리하는 오버헤드가 발생한다. 따라서 저장장치 내부의 자원을 효율적으로 관리하기 위한 연구가 필요하다.




[Fig. 4] Hybrid RAID

### 3.2 Flash Memory Chip의 구성

Fig. 5는 SSD의 플래시 메모리 칩 구조를 보여주고 있다[9-11]. 그림과 같이 SSD는 k개의 채널로 연결되어 있고 각 채널은 n개의 칩으로 구분된다. 그리고 각 칩은 m의 블록들로 구성된다. 각 플래시 메모리 칩은 독립적으로 동작한다. 따라서 병렬처리가 가능하고, 특정 칩에서 고장이 발생해도 이웃하는 칩에 영향을 주지 않는다[12-15]. 이러한 구조적인 특징은 독립적인 여러 대의 저장장치를 RAID로 그룹해 놓은 구조와 유사하다. 따라서 플래시 메모리 내 저장장치 내부적으로 RAID를 구축하는 것 또한 가능하다.

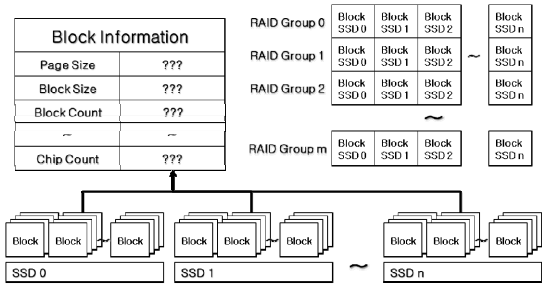



[Fig. 5] Structure of Flash Memory Storage

## 4. 플래시 메모리 전용 RAID 5

### 4.1 핵심 아이디어

본 논문에서는 플래시 메모리의 내부 블록 정보를 이용하여 RAID를 구축하기 위한 FMDR (Flash Memory Dedicated RAID) 기법을 제안한다. 이 기법은 각각의 SSD 블록 정보를 전송받아서 논리적 저장장치를 RAID 기반으로 구축한다. 이때 기본적인 RAID를 구축하고 남는 자원은 플래시 메모리의 독립 칩 간의 Internal RAID를 이용하여 추가적인 저장공간으로 확장한다.



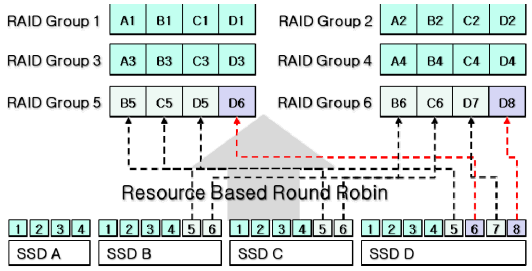

[Fig. 6] Key Idea

Fig. 6은 제안하는 FMDR의 핵심 아이디어를 보여주고 있다. 그림과 같이 n개의 SSD로부터 블록 정보를 전

송 받는다. 그리고 전송 받은 정보를 이용하여 논리적 저장장치를 생성한다. 저장장치는 서로 다른 SSD가 묶여있는 m개의 RAID 그룹으로 나누어 관리한다. 만약 각 SSD의 블록 자원이 동일 한 경우 일반적인 RAID 5와 동일한 구성으로 동작한다. 그러나 블록 자원의 수가 다른 경우 RR(Round Robin) 방식을 블록들을 할당하여 RAID를 구성한다.

## 4.2 Round Robin 기반 RAID 블록 할당 정책

본 절에서는 RAID 그룹에 저장 자원을 할당하기 위한 자원 기반 RR 정책을 소개한다. 이 방법은 먼저 최저 자원의 SSD에서 블록을 모두 소진할 때까지 RAID를 구축한다. 그다음 남은 자원을 기반으로 더 많은 자원을 가지고 있는 SSD로부터 추가 자원을 할당하는 방식으로 RAID를 구축한다.

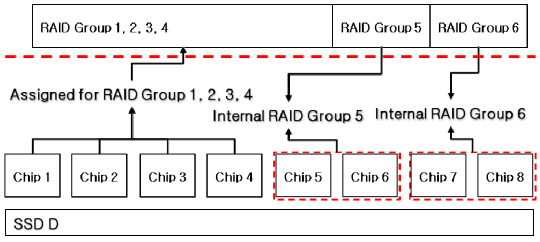



[Fig. 7] Resource Based Round Robin Policy

Fig. 7은 자원 기반 RR 정책을 보여주고 있다. 그림의 예제에서는 4개의 자원을 가지고 있는 SSD A와, 6개의 자원을 가지고 있는 SSD B, SSD C, 8개의 자원을 가지고 있는 SSD D가 있다. 먼저, 각 SSD로부터 자원을 하나씩 할당하여 RAID 그룹 1, 2, 3, 4를 만들었다. 그다음 RAID 그룹 5에서는 SSD B, C, D로부터 자원을 하나씩 할당하였다. 그러나 할당된 B5, C5, D5 만으로는 RAID 그룹을 구축할 수 없다. 제안하는 RR 정책에서는 RAID 그룹을 구성하기 위한 자원이 모두 채워질 때까지 남아있는 자원의 양을 기준으로 각각의 SSD로부터 자원을 하나씩 할당한다. 그림의 예제에서는 가장 많은 자원을 가지고 있는 SSD D로부터 D6를 추가 할당하였다. 이와 동일한 방법으로 RAID 그룹 6에서는 B6, C6, D7, D8을 할당하여 RAID를 구축하였다.

## 4.3 Internal RAID 정책

플래시 메모리는 읽기/쓰기 단위와 지우기 단위가 다르고, 데이터를 쓰기 전 지워야 하는 특징이 있다. 이러한 특징을 감추기 위해 논리적인 주소와 물리적인 주소를 매핑하여 별도로 관리한다. 따라서 SSD 내부의 칩을 묶어서 RAID 그룹을 만들기 위한 방법이 필요하다.

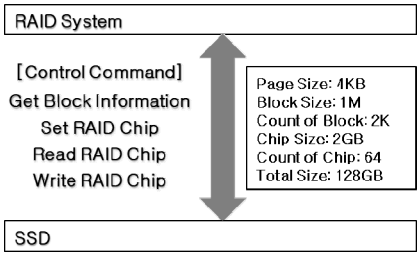



[Fig. 8] Internal RAID Policy

Fig. 8은 Internal RAID 정책을 보여주고 있다. 그림에서는 SSD D는 독립적으로 동작 가능한 플래시 메모리 칩을 8개 가지고 있다. 이 중, 칩 1, 2, 3, 4는 일반적인 RAID를 구축하기 위해 할당되었다. 그리고, 칩 5, 6과 칩 7, 8은 각각 RAID 그룹 5와 6을 위해 할당되었다. SSD는 논리적 주소와 물리적 주소가 분리되어 있고 각각의 주소에 할당된 메모리 블록을 관리하기 위한 별도의 관리 방법이 사용되고 있다. 따라서, 이렇게 SSD의 물리적인 주소를 가지고 있는 칩을 RAID로 묶기 위해서는 선택된 플래시 메모리 칩을 호스트에서 선택하거나 RAID로 묶기 위한 기능이 필요하다. 본 논문에서 제안하는 FMDR은 호스트로부터 추가적인 명령을 통해 SSD 내부의 칩 정보를 전송받거나 데이터를 읽고 쓸 때 어떤 칩으로부터 데이터를 읽을 것인지에 대한 명령을 포함한다.

## 4.4 플래시 메모리 블록 정보 및 명령 체계

RAID 그룹을 구축하는 기본적인 방법은 각각의 SSD로부터 자원을 하나씩 할당하는 방법이다. 따라서 서로 다른 용량의 SSD로부터 자원을 할당하기 위해서는 자원을 할당하는 정책뿐만 아니라 각각의 SSD로부터 전달받기 위한 자원의 정보 구조와 이를 기반으로 논리적인 저장장치를 구성하는 방법이 필요하다. 또한, 데이터를 쓸 때 단순히 논리적인 주소로 데이터를 쓰는 것뿐만 아니라 SSD 내부의 칩을 선택하여 데이터를 읽거나 쓰기 위한 방법이 필요하다. 본 절에서는 플래시 메모리의 내부 정보와 전달하기 위한 정보 구조와 명령 체계를 설계하였다.




[Fig. 9] Control Command Sets

Fig. 9는 SSD로부터 자원의 정보를 전달받거나 internal RAID를 컨트롤하기 위한 명령어 체계를 보여주고 있다. 제안하는 레이드 시스템은 Get Block Information 명령어를 통해 그림과 같은 칩과 플래시 메모리 정보를 전송받는다. 그다음 Internal RAID가 필요한 경우 Set RAID Chip 명령을 이용하여 각각의 SSD에 Internal RAID를 설정한다. Internal RAID가 설정된 경우, RAID로부터 데이터를 읽거나 쓰기 위해서는 칩 정보를 포함한 Read/Write RAID Chip 명령을 사용한다.

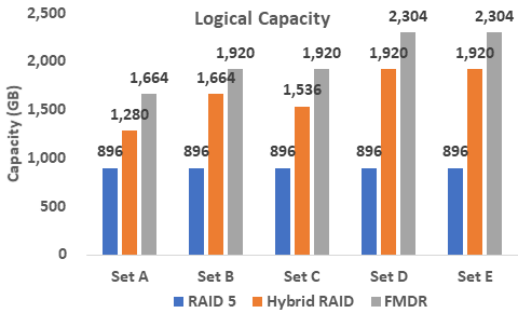
## 5. 실험 및 평가

본 절에서는 시뮬레이션을 통해 제안하는 FMDR의 효과를 증명한다. 실험을 위해 플래시 메모리의 페이지 크기를 4KB로 설정하였다. 블록당 페이지는 256개로 하였고, 플래시 메모리 칩 당 블록의 수는 2,048개로 가정하였다.

〈Table 1〉 RAID Test Sets

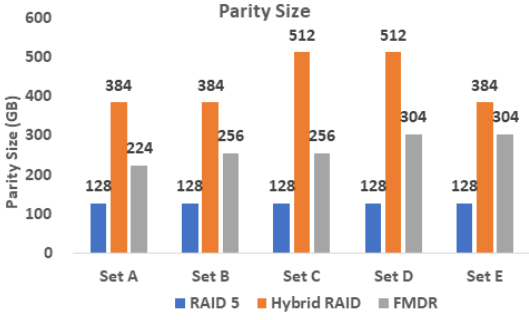
| Test Set | Set A | Set B | Set C | Set D | Set E |
|---|---|---|---|---|---|
| 1 | 128 | 128 | 128 | 128 | 128 |
| 2 | 128 | 128 | 128 | 128 | 128 |
| 3 | 128 | 128 | 128 | 128 | 256 |
| 4 | 128 | 256 | 128 | 128 | 256 |
| 5 | 128 | 256 | 256 | 384 | 384 |
| 6 | 256 | 384 | 256 | 512 | 384 |
| 7 | 384 | 384 | 512 | 512 | 384 |
| 8 | 512 | 384 | 512 | 512 | 512 |

Table. 1은 실험을 위해 랜덤으로 생성한 RAID 테스트 셋이다. 이 테스트 셋에서는 RAID 당 8대의 저장장치를 128, 256, 384, 512GB로 랜덤하게 할당하였다.




[Fig. 10] Total Capacity

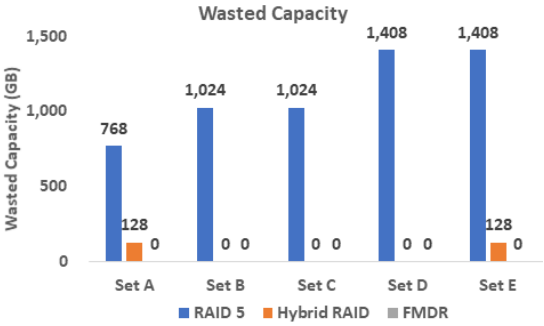
Fig. 10은 RAID를 구축할 때 생성된 논리적 저장장치의 전체 용량이다. 전체적으로 FMDR이 RAID 5보다 약 85.71 ~ 157.14% 용량이 증가하였고 Hybrid RAID와 비교하여 약 15.38 ~ 30% 증가하였다. 이러한 원인은 기본적인 RAID를 할당하고 남는 공간을 Internal RAID를 이용하여 효율적으로 관리하기 때문이다.




[Fig. 11] Generated Parity Size

Fig. 11은 생성된 패리티의 크기를 보여주고 있다. 전체적으로 FMDR의 패리티가 RAID 5보다 약 75 ~ 137.5% 증가하였고, Hybrid RAID보다 약 20.83 ~ 50% 감소하였다. RAID 5보다 패리티가 증가한 원인은 추가적인 RAID를 구축하기 때문이다. 반면 FMDR이 칩 단위의 효율적인 패리티를 생성하기 때문에 미러링하는 Hybrid RAID와 비교하여 패리티 데이터가 감소하는 효과가 있다.

Fig. 12는 FMDR을 적용했을 때 RAID 5와 비교하여 활용되는 자원을 보여주고 있다. FMDR은 추가적인 RAID를 구성하여 낭비된 공간의 용량이 0이고, RAID 5와 비교하여 전체적으로 768GB에서 1,408GB의 자원을 절약하는 효과를 보여주었다.




[Fig. 12] Wasted Capacity

## 6. 결론

본 논문에서는 SSD로 구성된 RAID에서 플래시 메모리 칩을 Internal RAID로 그룹화하는 방법을 제안하였다. 또한, 이 방법을 통해 증가하는 자원의 용량과 감소하는 패리티 데이터 효과를 시뮬레이션을 통해 증명하였다. 향후에는 Internal RAID를 더욱 효율적으로 관리하기 위한 방법을 연구할 예정이다.

## REFERENCES

[1] T.Ma, Z.Li and N.Liu, "Log-ROC: Log Structured RAID on Open-Channel SSD," *IEEE 40th International Conference on Computer Design*, pp.332-335, 2022.

[2] Z.Gu, J.Li, Y.Peng, Y.Liu and T.Zhang, "Elastic RAID: Implementing RAID over SSDs with Built-in Transparent Compression," *Proceedings of the 16th ACM International Conference on Systems and Storage*, pp.83-93, 2023.

[3] G.Si, W.Chen, X.Xu, Y.Sun and R.Gao, "Data Back-Up Algorithm Based on Blockchain with Network Erasure Codes," *IEEE 11th Joint International Information Technology and Artificial Intelligence Conference*, pp.232-235, 2023.

[4] S.Wang, Q.Cao, Hong.Jiang, Z.Lu, J.Yao, Y.Chen and A.Pan, "Explorations and Exploitation for Parity-based RAIDs with Ultra-fast SSDs," *ACM Transactions on Storage*, Vol.20, No.6, pp.1-32, 2024.

[5] S.Pashazadeh, L.N.Tazehkand and R.Soltani, "RSS_RAID a Novel Replicated Storage Schema for RAID System," *Data Science: From Research to Application*, Vol.45, pp.36-43, 2020.

[6] J.H.Choi, J.H.Park and S.J.Lee, "Reassembling Linux-based Hybrid RAID," *Journal of Forensic Sciences*, pp.966-973, 2019.

[7] R.Pitchumani and Y.S.Kee, "Hybrid Data Reliability for Emerging Key-Value Storage Devices," *FAST '20 Open Access Sponsored by NetApp*, 2020.

[8] Y.Cao, "Future Development of Redundant Array of Independent Disks," *International Conference on Applied Physics and Computing*, pp.8-10, 2022.

[9] H.S.Lee, "Performance analysis and prediction through various over-provision on NAND flash memory based storage," *Journal of Digital Convergence*, Vol.20, No.3, pp.343-348, 2022.

[10] H.S.Lee, "A Memory Mapping Technique to Reduce Data Retrieval Cost in the Storage Consisting of Multi Memories," *Journal of Internet of Things and Convergence,* Vol.9, No.1, pp.19-24, 2023.

[11] H.S.Lee, "A Study on Characteristics and Techniques that Affect Data Integrity for Digital Forensic on Flash Memory-Based Storage Devices," *Journal of Internet of Things and Convergence,* Vol.9, No.3, pp.7-12, 2023.

[12] H.S.Lee, "High Efficiency Life Prediction and Exception Processing Method of NAND Flash Memory-based Storage using Gradient Descent Method," *Journal of Convergence for Information Technology*, Vol.11, No.11, pp.44-50, 2021

[13] H.S.Lee, "A Safety IO Throttling Method Inducting Differential End of Life to Improving the Reliability of Big Data Maintenance in the SSD based RAID," *Journal of Digital Convergence,* Vol.20, No.5, pp.593-598, 2022.

[14] H.S.Lee, "An Efficient SLC Transition Method for Improving Defect Rate and Longer Lifetime on Flash Memory," *Journal of Internet of Things and Convergence,* Vol.9, No.3, pp.81-86, 2023.

[15] H.S.Lee, "An Efficient SLC Transition Method for Improving Defect Rate and Longer Lifetime on Flash Memory," *Journal of Internet of Things and Convergence,* Vol.9, No.3, pp.81-86, 2023.

이 현 섭(Hyun-Seob Lee) [종신회원]

- 2013년 2월 : 한양대학교 컴퓨터공학과 (공학 박사)
- 2012년 3월 ~ 2021년 2월 : 삼성전자 책임연구원
- 2021년 3월 ~ 현재 : 백석대학교 컴퓨터공학부 조교수

〈관심분야〉
인공지능, 저장시스템, 임베디드 시스템