Algorithms and Data Structure for Flash Memories

Kookdo Han, HyunWook Chon
Embedded System Lab.
School of Computer Science
Kookmin University

Introduction

- Flash memory
  - Electrically-erasable programmable read-only memory (EEPROM)의 한 종류
  - 비 휘발성
  - 쓰기 횟수 제한 (10000~100000)
  - 종류
    - NOR, NAND

- 기존의 magnetic disk와 다른 특성으로 인한 flash memory 관리와 저장 기술 개발 필요

- Flash memory 저장 기술을 위한 data structure와 algorithm에 대한 조사
  - Block-mapping techniques
  - Flash-specific file systems
  - Beyond file systems

Flash memory and magnetic disk

- Magnetic disk
  - 임의 접근 시 기계적인 움직임 수반
  - 기록된 데이터 변경 가능
  - 읽기, 쓰기 수행 시간이 거의 동일

- Flash memory
  - 동일한 접근 시간
  - 기록된 데이터를 변경하기 위해 지우기, 복사와 같은 추가 작업 필요
  - 읽기, 쓰기, 지우기 수행 시간에 차이가 있음

Flash memory의 종류

- NOR flash memory
  - 장점
    - Random, direct access interface
    - Fast read performance
  - 단점
    - Slow erase and write

- NAND flash memory
  - 장점
    - Better performance for erase and write
    - Lower cost
      - NOR: requires full address and data bus lines
      - NAND: commands and data are multiplexed onto eight I/O lines
  - 단점
    - Slow random access

Flash memory의 구조 및 특성

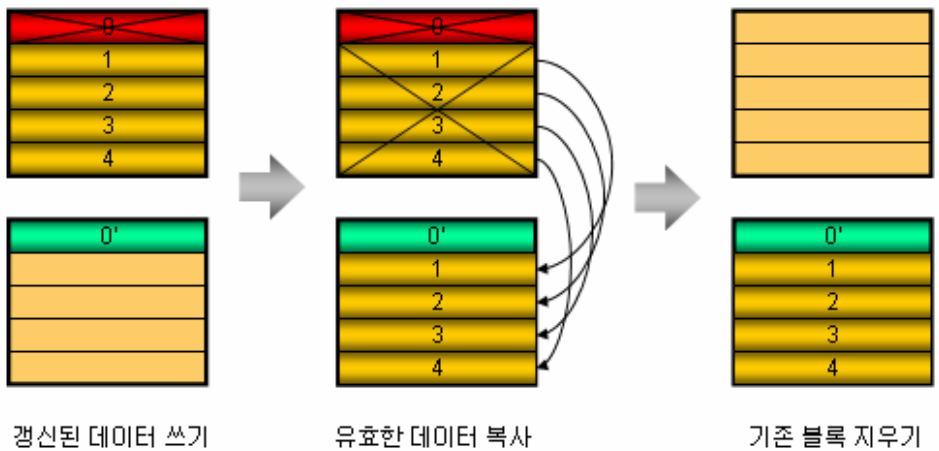
- 읽기(read): 페이지(page) 단위
- 쓰기(write): 페이지(page) 단위
- 지우기(erase): 블록(block) 단위

	Access Time		
	Read	Write	Erase
NOR	23μs	28ms	1.2sec
NAND	156μs	652μs	2ms

< Flash memory 특성 >

Flash memory의 구조 및 특성 (cont.)

❑플래시 메모리는 덮어쓰기(overwrite)가 되지 않음



< 갱신(update) 연산 과정 >

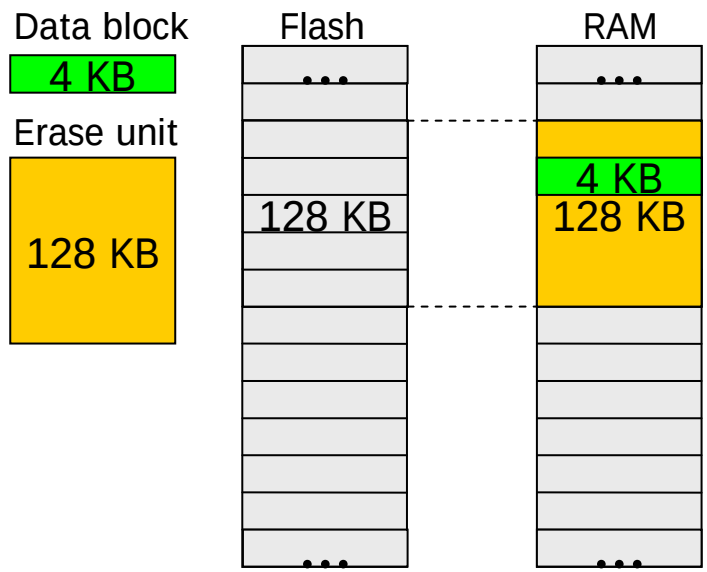
Block-mapping Techniques

- Flash memory를 disk와 같이 읽고 쓰기 위해 고정된 크기의 데이터 블록을 가지는 블록 디바이스로 처리
  - Magnetic disk에서 사용되는 방법
    - 파일 시스템이 디바이스 드라이버에 읽기, 쓰기 요청
    - 디바이스 드라이버는 flash에 블록을 읽고 씀
- Magnetic disk에서와 같이 Flash address와의 다순 순차적인 mapping에 따른 문제점
  - 쓰기 횟수 제한에 따른 성능 저하와 수명 단축
  - 파일 시스템의 데이터 블록 크기와 flash에서의 erase 블록 크기 차이에 따른 데이터 손실 위험

Block-mapping Techniques (cont.)

- 쓰기 횟수 제한에 따른 성능 저하와 수명 단축
  - 몇몇 데이터 블록이 다른 블록보다 더 많이 쓰여졌을 때
    - Magnetic disk에서는 문제가 발생하지 않음
    - Flash에서는 특정 블록의 erase가 자주 발생하여 수명이 줄어들고 erase와 write에 의해 access 시간이 느려짐

- 파일 시스템의 데이터 블록 크기와 flash에서의 erase 블록 크기 차이에 따른 데이터 손실 위험
  - Flash의 erase 블록보다 작은 데이터 블록을 쓸 때
    - 실제로 쓸 데이터 보다 더 많은 양을 erase/write 함
    - Erase/write하기 전에 전원 공급이 중단된 경우 데이터 손실이 큼



- Solution
  - Wear-leveling

Wear Leveling

- Flash는 10K번을 erase/write하게 되면 노화 현상에 의해 bad 블록이 될 가능성이 증가됨

- 전체 Flash를 균일하게 사용하기 위해 다양한 wear leveling 기법이 사용

- 완벽한 wear leveling을 구현 방법
  - Flash를 끝까지 사용하면 항상 처음 블록부터 다시 사용하는 방법
  - 단점: 블록 이동과 지움으로 인해 유효한 데이터의 블록 이동이 빈번해지고 결과적으로 시스템의 성능 저하

- wear leveling과 성능을 모두 만족시키는 방법
  - 블록마다 지움 횟수를 관리하면서 현재 유효한 데이터가 가장 적은 블록을 최우선 지움 대상 블록으로 선택

The Block-Mapping Idea

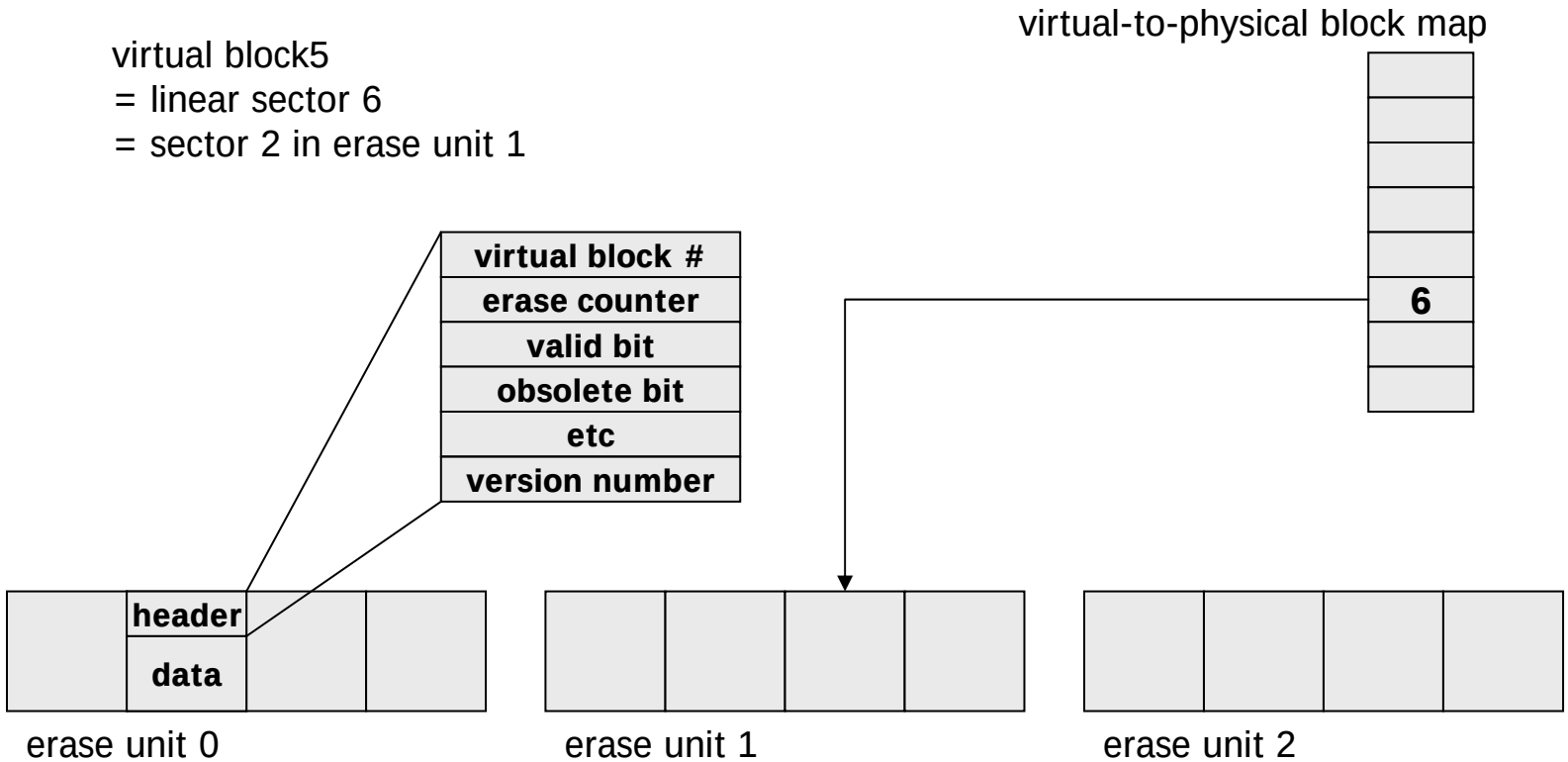
- 데이터의 갱신을 위해 virtual block이 필요할 때 sector (physical flash address)를 덮어쓰지 않고 다른 sector에 저장하고 virtual-block-to-sector map을 갱신
  - Erase 블록의 쓰기 횟수를 균등하게 하기 위해 다른 블록에 씀
  - Erase와 rewriting없이 flash에 씀
  - 쓰기 연산 중에 전원 공급이 중단될 경우 손실 최소화

- Block이 쓰여질 때
  - 소프트웨어는 비어 있거나 지워진 sector를 찾음
  - Sector와 header의 모든 비트는 1
  - Sector의 header의 free/used 비트는 클리어
  - Virtual block number는 header에 쓰여지고 새로운 데이터가 선택된 sector에 쓰여짐
  - Header의 prevalid/valid 비트는 클리어
  - 이전 sector의 header의 valid/obsolete 비트는 클리어

The Block-Mapping Idea (cont.)

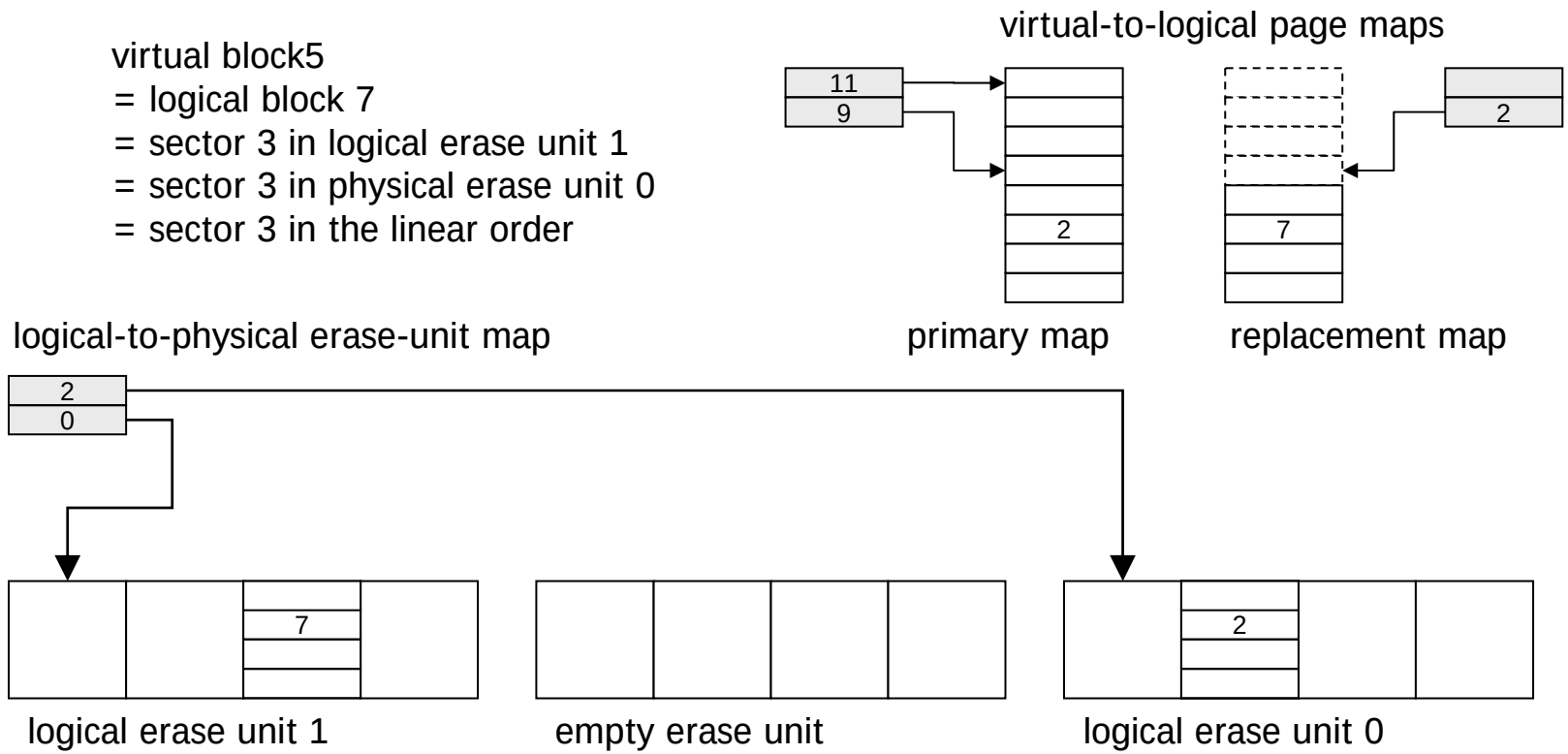
- 쓰기 연산 중에 전원 공급이 중단된 경우
  - 새로운 섹터가 valid로 설정되기 전에 전원 공급이 중단된 경우에는 저장된 내용을 무시
  - 새로운 섹터가 valid로 설정되었지만 이전 섹터가 obsolete가 설정되기 전에 전원 공급이 중단된 경우에는 두 섹터가 모두 유효함
    - 하나의 데이터를 선택하고 나머지 하나는 obsolete로 설정
    - 선택 방법: version number를 이용한 최신 데이터 선택

Data Structures for Mapping



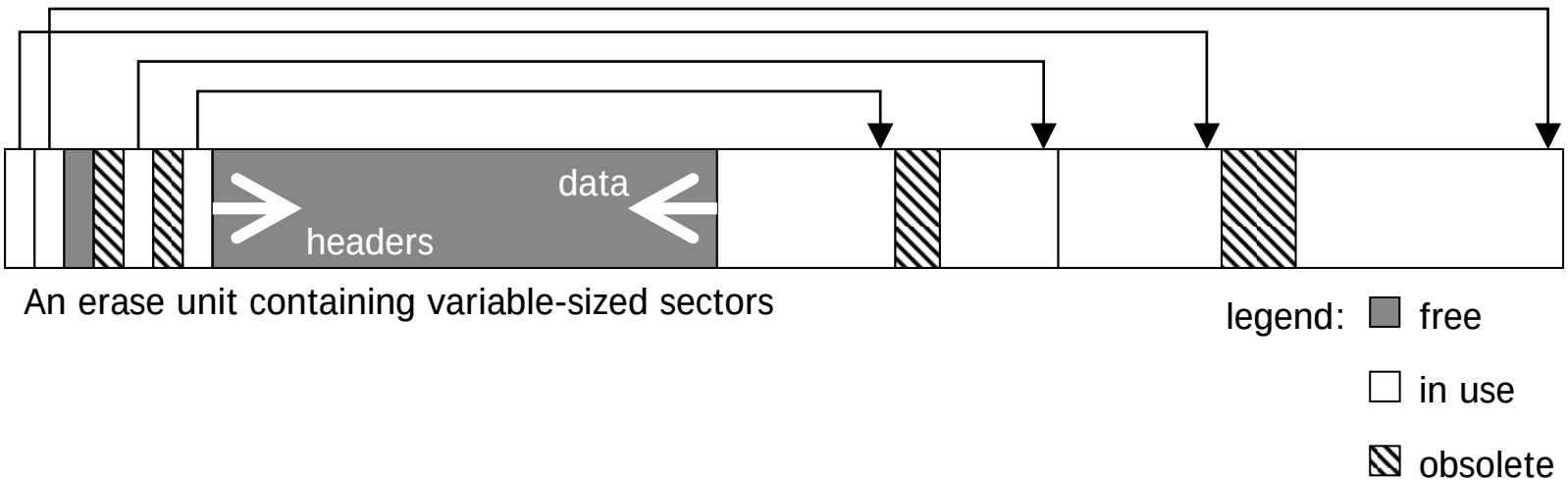
< Block mapping >

Data Structures for Mapping (cont.)



< FTL mapping structure >

Data Structures for Mapping (cont.)



< Block mapping with variable-length sectors >

Erase-Unit Reclamation

- Garbage collection
  - Obsolete 섹터와 빈 섹터의 감소가 축적되면 obsolete 섹터를 재사용
  - CPU가 idle 상태이거나 빈 공간이 부족할 때 수행

- Garbage collection 단계
  1) 재사용할 erase 블록들을 선택
  2) 선택된 블록들의 valid 섹터를 새로 할당한 빈 공간으로 복사
  3) 블록 mapping 정보 수정
  4) 재사용할 블록들을 erase하여 빈 섹터로 만듦

Wear-Centric Reclamation Policies and Wear-Measuring Techniques

- 단순한 wear-leveling 기술
  - Erase블록의 재사용에 의해 처리
  - 각각의 erase블록의 헤더는 erase counter를 포함
  - 많이 쓰여진 블록이 재사용될 때 적게 쓰여진 블록의 counter와 비교하여 차이가 크면 wear-leveling 적용
  - 재사용될 때 많이 쓰여진 블록의 데이터를 적게 쓰여진 블록으로 복사
  - 고정 블록을 알고 있다는 가정 하에 고정 블록을 적게 쓰여진 블록에서 다른 블록으로 복사하여 전체적인 쓰기 횟수가 비슷하도록 쓰기 비율을 맞춤

Wear-Centric Reclamation Policies and Wear-Measuring Techniques (cont.)

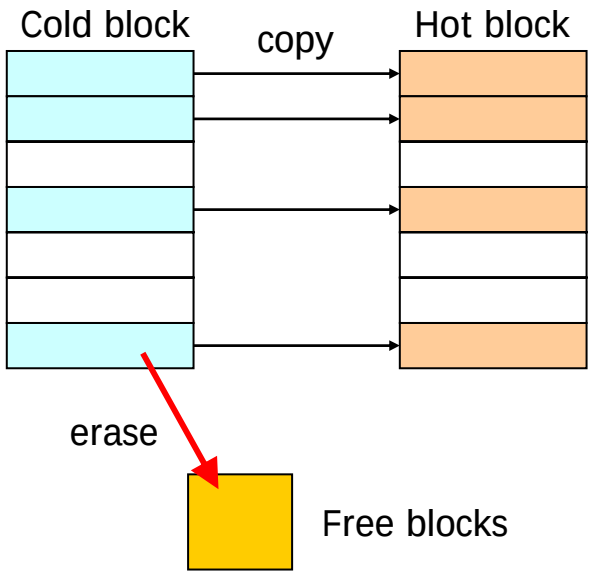
- Erase 횟수의 상한선을 유지하는 기술
  - 재사용을 위해 선택된 erase블록의 valid 데이터는 다른 블록으로 복사하고 즉시 삭제하지 않음
  - 지워질 대상으로 설정하고 큐에 추가함
  - 큐에서 가장 적게 쓰여진 블록이 빈 블록이 필요할 때 erase됨

- Erase 블록의 순위를 이용한 기술
  - Erase 블록의 순위에 따라 erase 횟수를 비교
  - Erase counter를 저장할 필요가 없음

Combination Wear-Leveling with Efficient Reclamation

- Wear-leveling에 따른 블록 교체 기술
  - 블록의 레벨이 최대이면서 비어있는 블록 – hot 블록
  - 레벨이 최소이며 할당된 블록 – cold 블록
  - Hot 블록은 소거 연산이 많이 이루어져 높은 레벨이므로 빈 블록으로 할당하지 않고, cold 블록과 교체
  - Cold 블록은 할당된 블록 중에서 레벨이 최소인 블록으로써 erase가 드문 블록이고 대기큐에 쌓이게 되며, hot 블록을 만나게 되면 교체되어 hot 블록의 erase 가능성이 낮아짐
  - 블록에 대한 교체 작업은 특정 블록에 대한 erase의 집중을 막아 erase 횟수를 평준화함

Combination Wear-Leveling with Efficient Reclamation (cont.)



< Wear-leveling에 따른 블록 교체 >

FLASH-SPECIFIC FILE SYSTEMS

- Background : Log-Structured File System
- The Journaling Flash Filing System
- YAFFS : Yet Another Flash Filing System
- The Trimble File System
- The Microsoft Flash File System
- Norris Flash File System
- Other Commercial Embedded File System

Log-Structured File System

- In-place modification 의 문제점
  - 탐색 시간 오래 걸림 (long seek),
    회전 지연 시간 (rotational delay)
  - 데이터를 디스크에 저장 도중 시스템 다운이나 문제가 발생할 경우 시스템 손상

- Journaling file system
  - 데이터를 디스크에 쓰기 전에 Log 에 데이터를 저장
  - NTFS (Windows), JFS (AIX, Linux), XFS (IRIX, Linux), ext3, reiserfs, HFS+

Log-Structured File System (cont.)

- 순차적인 추가
  - 전용 로그를 유일한 디스크 구조로 사용
- 다수의 파일 시스템 변경을 하나의 큰 로그 항으로 모아 단일 연산으로 디스크에 씀
- 파일 시스템의 디스크상의 구조와 커널 루틴을 크게 고쳐야 함
- 쓰기는 전부 로그의 꼬리에 행해짐
  - 전부 순차적, 디스크 탐색은 없어짐

Log-Structured File System (cont.)

- 장점
  - 회전 인터리빙도 제거 (트랙 단위로 씀)
  - 데이터와 메타 데이터를 한번의 원자적 쓰기로 쓸 수 있음
  - 크래쉬 회복이 신속

- 문제점
  - 데이터 검색시 필요한 데이터의 로그를 찾아야 함
    - 인-메모리 캐쉬로 해결

The Journaling Flash Filing System

- Axis Communication
- JFFS2
- NOR Flash Memory 에 적합
- Log-structured file system 기반

The Journaling Flash Filing System (cont.)

- Write
  - LFS 와 유사하게 데이터를 로그형태로 순차적으로 기록
- Read
  - 로그를 역순으로 검색해 가장 최신의 데이터 읽음
- Garbage Collection
  - 앞부분 블록부터 차례로 삭제 수행해 빈 공간 확보
- JFFS2
  - Garbage collection 을 좀 더 효율적으로 수행하기 위해, 유효한 데이터의 비율 등을 고려해 최적의 블록을 선택적으로 삭제
  - 단위공간 당 단가가 높은 플래시 메모리의 공간을 효율적으로 활용하기 위해 압축기능 사용
  - 보다 다양한 종류의 inode 구조 지원

The Journaling Flash Filing System (cont.)

- Wear-leveling
  - 쓸모 없는 데이터가 많은 블록을 삭제
  - cleaning operation 이 100번 발생 할 때 마다 중요한 데이터를 많이 삭제된 블록으로 이동

YAFFS : Yet Another Flash Filing System

- NAND 플래시 메모리 전용 파일 시스템
  - 마운트 속도와 NAND 유형 플래시 메모리의 입출력 속도에서 JFFS2 보다 성능면에서 우수
  - LFS 방식을 사용하므로 마운트 시 플래시 메모리 전체를 스캔해야 하므로 플래시 메모리가 클수록 마운팅 시간 길어진다
  - 파일 수정시 해당 파일의 모든 데이터를 다시 플래시 메모리에 쓰게 되므로 메모리 낭비 발생

YAFFS : Yet Another Flash Filing System (cont.)

- YAFFS
  - 512 byte chunks
- YAFFS2
  - 1 ~ 2 Kbyte chunks
  - chunks header
    - file ID, position, sequence number
  - 파일 삭제시 trash directory 이동
  - write : 1.5x ~ 5x
  - delete : 4x
  - garbage collection : 2x
  - 25 ~ 50 % lower RAM footprint

YAFFS : Yet Another Flash Filing System (cont.)

- Wear leveling
  - erase unit 을 random 하게 선택
  - NOR 디바이스보다 중요도가 적다
  - storage 의 용량은 줄지만 file system 에는 크게 영향 없다

The Trimble File System (cont.)

- NOR (Trimble Navigation)
- YAFFS 의 간단한 파일 시스템 구조

- 구성
  - 4 byte header
    - file number, chunk number
  - 252 byte chunks

- File
  - header sector, file number, valid/invalid word, file name, 14 file record 로 구성

The Trimble File System

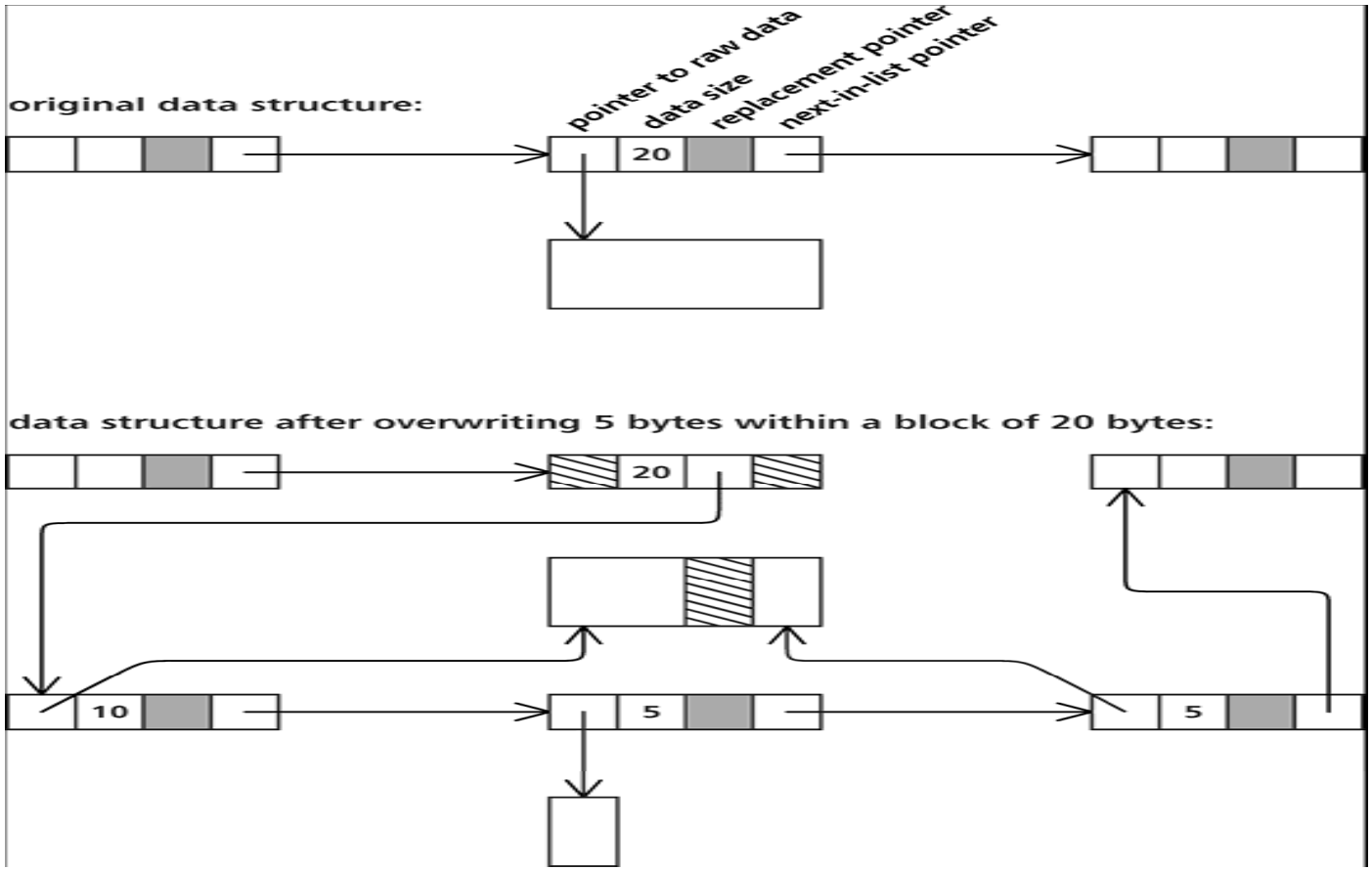
- erase count
  - erase block 을 삭제 전 erase count 를 업데이트

- wear-leveling 에 사용
  - erase count 가 작은 block 를 erase unit

The Microsoft Flash File System

- FFS2 라 불리는 이동식 플래시 메모리의 일반적인 파일 시스템
- 불충분한 write performance
- NOR Flash
  - one large erase unit
  - write-once device

The Microsoft Flash File System (cont.)



Norris Flash File System

- Micorsoft Flash File System 기반
  - Handheld audio recorder 에 사용하기 위한 파일 시스템

Other Commercial Embedded File System

- TargetFFS
  - NAND, NOR
- smxFFS
  - nonremovable NAND
  - block-mapping device driver
  - simple FAT-like file system

Other Commercial Embedded File System

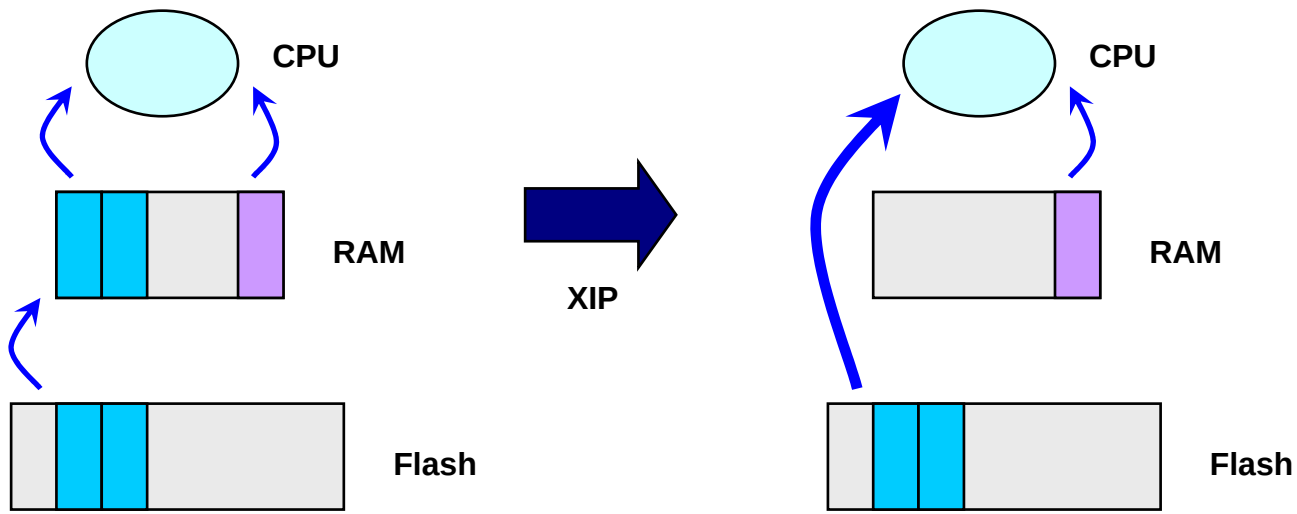
- EFFS
  - Failsafe File Systems
    - EFFS-STD
      - NOR, NAND 에 사용
  - FAT Compatible File System
    - EFFS-FAT
      - PC
      - Comapct Flash, MMC, SD, HDD
    - EFFS-THIN
      - 8-bit processor
- FLite
  - 일반적인 FAT 파일 시스템과 FTL-compatible 블록 맵핑 디바이스 드라이버를 결합 시킨 파일 시스템

Beyond File Systems

- Flash-Aware Application-Specific Data Structures
  - B-trees, R-trees
  - time에 의해 정렬
  - 문제점
    - application-specific data structure 를 위한 파티션과 다른 data (file) 를 위한 파티션 요구
    - 파티션을 동적으로 조절 할 수 없다면 wasting space 발생

Beyond File Systems (cont.)

- Execute-in-place (XIP)
  - Flash file system에 저장되어 있는 실행 코드를 램에 적재하지 않고 그대로 실행
  - 램에 대한 메모리 요구량 감소



Beyond File Systems (cont.)

- Flash-Based Main Memories
  - magnetic disk 와 DRAM 대체
    - single-word access 빠르다
    - nonvolatile
    - 값이 싸다
  - wide bus

Summary

- Flash memory를 모바일 기기의 저장 장치로 사용
- 새로운 형태의 디바이스를 위해 새로운 소프트웨어 기술과 시스템 구조 필요
- flash memory 기술들의 조사를 통한 현재의 기술 파악
- 기반 기술 파악을 통한 새로운 flash memory 기술 개발 촉진