운 영 체 제

5장 디스크 스케줄링과 파일시스템

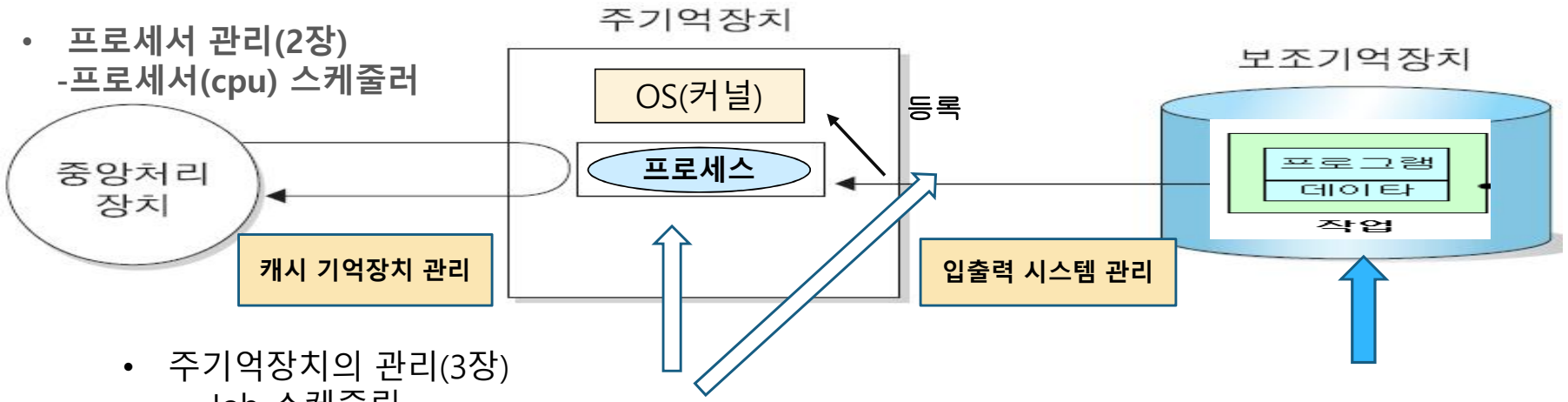
5장 디스크 스케줄링과 파일 시스템

학습 내용

- 5장 디스크 스케줄링과 파일시스템
  - 5.1 개요
  - 5.2 디스크 구조
  - 5.3 디스크 스케줄링
    - FCFS(First Come First Served) 스케줄링
    - SSTF(Shortest Seek Time First) 스케줄링
    - SCAN 스케줄링 및 LOOK 스케줄링
    - C-SCAN 스케줄링 및 C-LOOK 스케줄링

5장 디스크 스케줄링과 파일시스템

5장 디스크 스케줄링과 파일 시스템



- 프로세서 관리(2장)
  -프로세서(cpu) 스케줄러

- 주기억장치의 관리(3장)
  -Job 스케줄링
  -주소 바인딩
  -주기억장치 관리 기법
  =>인출 기법, 배치 기법,교체 기법, 할당 기법(연속)
- 가상 메모리 관리(4장)
  -분산 할당 기법
  =>페이징 기법, 세그먼테이션 기법
  -페이지 교체 기법
  => FIFO(First In First Out),
  OPT(Optimal Replacement), LRU, LFU, NUR, SCR

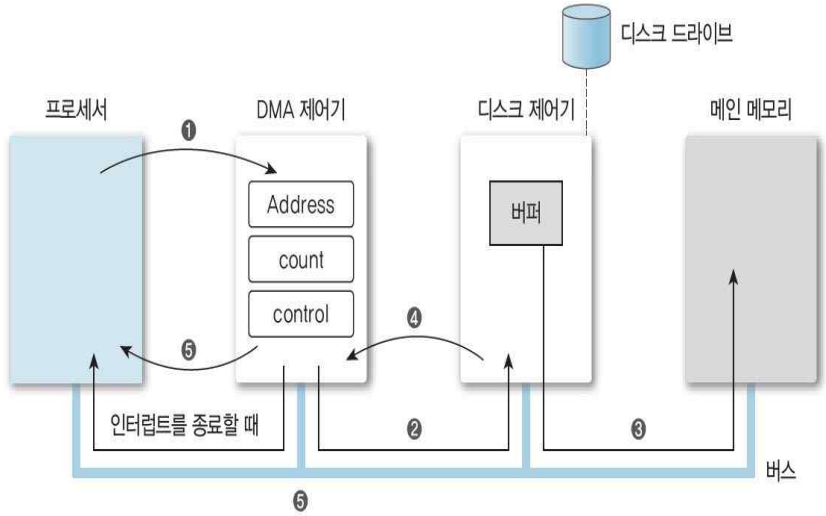
- 디스크 스케줄링 (5장)
  -FCFS, SSTF, SCAN, C-SCAN
  => 탐색시간 최소화
- 파일 시스템(5장)
  -파일 구조
  -디렉토리 구조

개요

- 현재의 컴퓨터와 시스템은 주로 디스크 시스템 중심의 처리
  - 컴퓨터 시스템의 주요 목적은 프로그램을 수행하는 것이고 프로그램을 수행하려면 프로그램과 관련 데이터를 메인 메모리에 올려야 한다.
  - 보조 기억 장치의 목적은 방대한 데이터를 영구히 저장하는 것으로 이용하고 초기에는 자기 테이프를 사용하였지만 지금은 백업용으로 사용하고 현재는 자기 디스크를 사용한다.
- 운영체제는 디스크나 CD-ROM 같은 기억용량이 큰 기억장치를 관리·운영
- 파일 시스템(file system)
  - 전체적인 정보 관리 시스템의 한 부분
- 파일 시스템은 사용자가 운영체제에서 가장 관찰하기 쉬운 부분

디스크 시스템

- DMA 입출력 제어 방식을 사용하는 디스크 시스템 예
  - DMA 제어기, 디스크 제어기, 디스크 드라이버 로 구분
    - DMA 제어기: 기억장치와 입출력 모듈 간의 데이터 전송을 별도의 하드웨어인 DMA 제어기가 처리.
    - 디스크 제어기(입출력 모듈)
      - 디스크 드라이버(입출력 장치)의 인터페이스 역할
      - 프로세서에서 명령을 받아 디스크 드라이버 동작
    - 디스크 드라이버(입출력 장치) : 구동 모터, 액세스 암 이동 장치, 입출력 헤드 부분의 기계적인 부분 담당으로 디스크 드라이버는 탐색seek, 기록, 판독 등 명령 수행
    - **디스크의 정보는 드라이버 번호, 디스크 번호, 표면 번호, 트랙 번호 등으로 나누는 디스크 주소로 참조**
    - **DMA 제어기가 확장되어 입출력 전담 프로세서 즉 입출력 채널을 가진 디스크 시스템도 있음.**

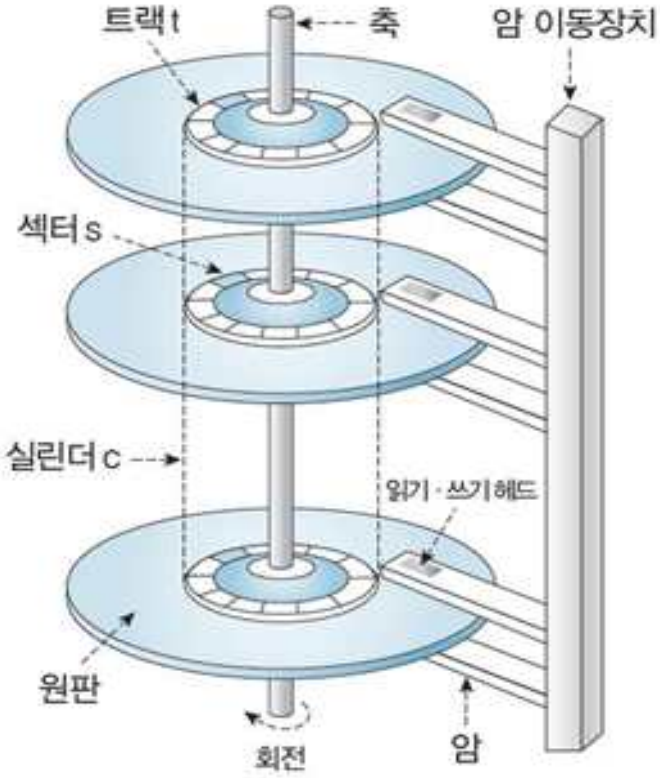


DMA 전송 동작

프로세서는 데이터 전송을 시작하고 종료 할 때만 관여

디스크 구조

- 디스크의 종류: 고정 헤드 디스크, 이동 헤드 디스크
- 원판(platter) : 하드디스크에서 데이터가 저장하기 위해 자성 재료가 입혀진 원형 평판 표면
- 트랙(track)
  - 원형 평판 표면에 데이터를 저장할 수 있는 동심원을 가리킴 또는 중심축에 대해 동심원으로 나누어진 것
- 섹터(sector)
  - 디스크 구조에서 부채꼴 모양으로 자른 것처럼 나누어진 구역
  - 트랙들을 일정한 크기로 구분한 부분
  - 섹터는 데이터 기록이나 전송의 기본 단위로, 일반적으로 512바이트의 데이터 영역으로 구성. 섹터는 고유 번호가 있어 디스크에 저장된 데이터의 위치 식별 가능
- 블록(block)
  - 보통 몇 개의 섹터를 모아 블록 이라 하며 입출력의 기본 단위
- 실린더(cylinder)
  - 동일한 동심원으로 구성된 모든 트랙, 즉 동일한 위치에 있는 모든 트랙의 집합을 의미. 헤드가 한번에 판독/기록 할 수 있는 동일한 인덱스 번호를 가진 트랙의 모임

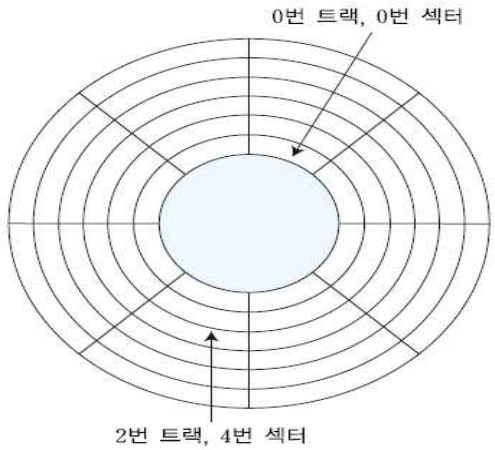


이동헤드 디스크의 구성도

- **디스크의 정보는 드라이버 번호, 디스크 번호, 표면 번호, 트랙 번호 등으로 나누는 디스크 주소로 참조**

디스크 구조(디스크 주소)

- 디스크의 정보는 드라이버 번호, 디스크 번호, 표면 번호, 트랙 번호 등으로 나누는 디스크 주소로 참조
- 각 트랙은 다수의 섹터(sector)로 구성
  - 각 섹터들은 일정한 공간(inter-sector gab)을 두어 구분
  - 각 섹터는 같은 용량을 저장



- 디스크 주소 예제

어떤 자기 디스크 장치에 있는 양쪽 표면이 모두 사용되는 8개의 디스크가 있는데, 각 표면에는 16개 트랙과 8개의 섹터가 있다. 트랙 내의 각 섹터에 하나의 레코드가 있다면 디스크 내의 레코드에 대한 주소 지정에는 몇 비트가 필요한가?

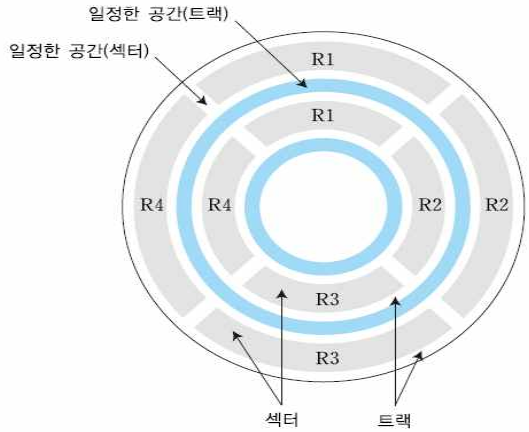
[풀이]

레코드의 총 수 = 디스크 수 x 면 수 x 면당 트랙 수 x 트랙 당 섹터 수 x 섹터당 레코드 수

= 8 x 2 x 16 x 8 x 1 = 2048 개

이 레코드에 대한 주소 지정을 하기 위해서는 2048개의 레코드가 저장 되어 있으므로

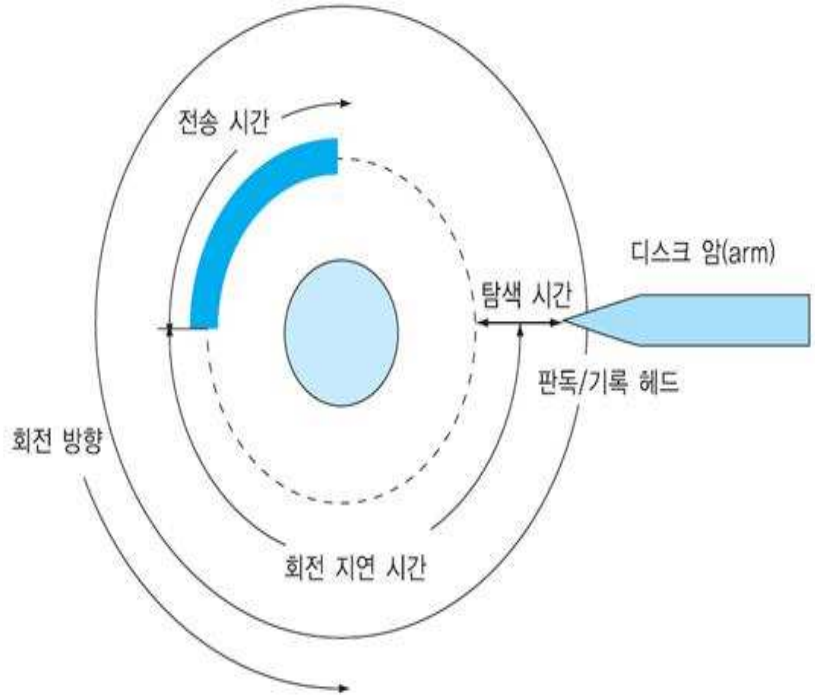
$2048=2^{11}$ => 11개 비트로 레코드 주소 식별



디스크 구조(디스크 접근 시간)

- 탐색시간(seek time)
  - 실제 원하는 트랙위치를 찾는 데 소요되는 시간
- 회전지연시간(latency time)
  - 원판이 회전하면서 처리할 데이터가 있는 위치까지(섹터까지) 오는 시간
- 전송시간(transmission time)
  - 읽은 데이터를 주기억장치에 전달하는 데 소요되는 시간
- 디스크 접근 시간(Access time)
  - 읽은 데이터를 주기억장치에 전달하는데 소요시간
  - 탐색 시간, 회전 지연 시간, 전송 시간의 총합

디스크 접근 시간 = 탐색 시간 + 회전 지연 + 데이터 전송 시간



디스크 접근의 구성 단계

디스크 스케줄링

- 운영체제
  - 디스크 액세스 요청을 스케줄 하여 디스크 처리하는 평균 시간을 향상시키는데 처리량을 최대화하고 평균 반응시간을 최소화하면서 탐색 시간 최소화하도록 해야 함 => 그러나 개별 요청에 지연이 발생할 수 있으므로 시스템 자원을 효과적으로 사용하려면 스케줄링 필요
- 디스크 스케줄링의 전략과 목적
  - 처리량 최대화
  - 탐색 시간 최소화 : 디스크 헤드(암) 이동 시간
  - 반응 시간 최소화 : 요청 후 서비스할 때까지 대기시간

디스크 스케줄링

- 디스크 스케줄링(disk scheduling)
  - 현재의 헤드 위치를 근거로 가장 적은 기계적 이동으로 이러한 요청들을 처리할 수 있도록 대기 큐를 재배열하는 과정
- 디스크 스케줄링의 2가지 방법
  - 탐색 시간을 최적화하는 방법
  - 회전 지연 시간을 최적화하는 방법
- 일반적으로 디스크 스케줄링 방법은 탐색 시간을 최소화 하는 방법 이용
  - 탐색 시간(seek time)은 회전 지연 시간(latency time)에 비하여 훨씬 많은 시간이 소요되기 때문
  - 실제 회전 지연 시간의 감소는 시스템 전체의 생산성을 증진시키는 데 크게 영향을 주지 못함

디스크 스케줄링

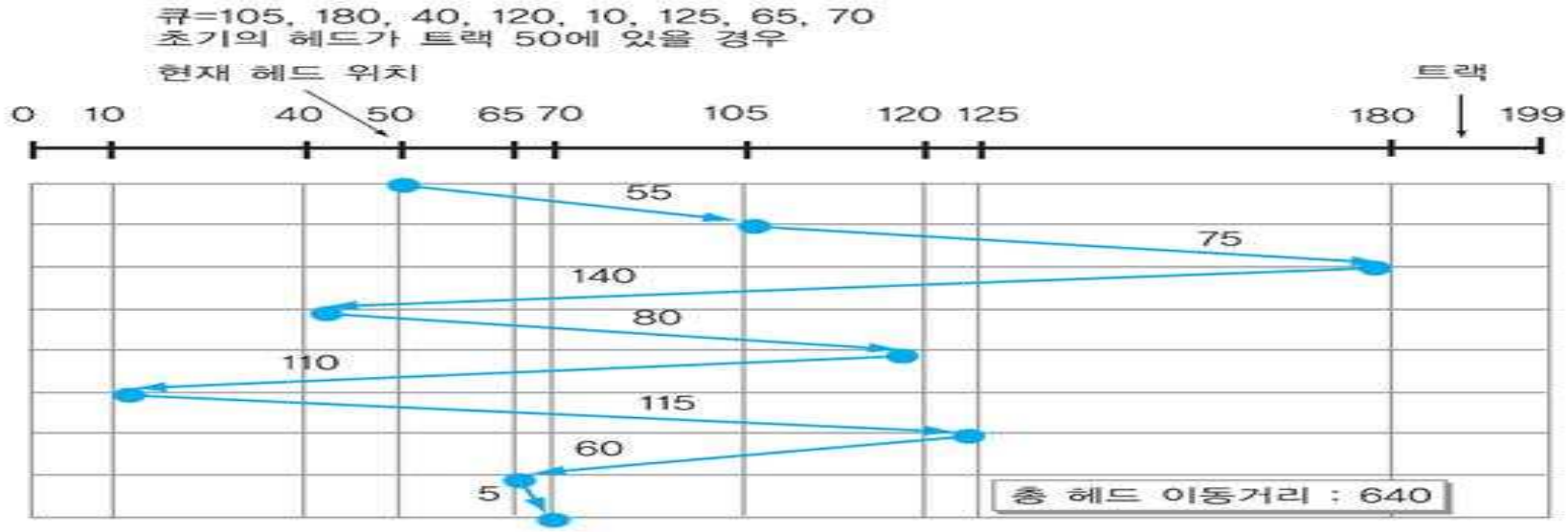
- 디스크 스케줄링(disk scheduling)
  - 현재의 헤드 위치를 근거로 가장 적은 기계적 이동으로 이러한 요청들을 처리할 수 있도록 대기 큐를 재배열하는 과정
- 일반적으로 디스크 스케줄링 방법은 탐색 시간을 최소화 하는 방법 이용
- 디스크 스케줄링(disk scheduling)의 종류
  - FCFS(First Come First Served) 스케줄링
  - SSTF(Shortest Seek Time First) 스케줄링
  - SCAN 스케줄링 및 LOOK 스케줄링
  - C-SCAN 스케줄링 및 C-LOOK 스케줄링

디스크 스케줄링

- FCFS(First Come First Served) 스케줄링
  - 가장 간단한 형태
  - 먼저 도착한 요청을 우선적으로 서비스하는 기법
  - 공평성이 보장되고 프로그래밍하기도 쉽고 스케줄링으로 인한 오버헤드 적음
  - 높은 우선순위를 가진 요청이 도착하여도 실행의 순서가 바뀌지 않음
  - 대기 중인 요청들 간의 위치에 대한 관련성을 무시하기 때문에 가장 좋은 처리 방법이라고는 할 수 없음
  - 탐색 패턴을 최적화하려는 시도가 없는 스케줄링 기법으로, 일반적으로 효율이 낮음

디스크 스케줄링(FCFS 스케줄링)

디스크 시스템은 총 200개의 실린더로 구성 되었다. 현재 디스크 헤드 위치는 50번 트랙이고 서비스하는 동안 다른 추가의 요구가 도착하지 않는 조건일 경우 FCFS 기법으로 아래의 디스크 입출력 요구 큐에 있는 요구들을 모두 서비스하기 위한 디스크 헤드의 총 이동 거리 D와 이동 순서를 구하시오.
(이동 거리는 각 트랙 사이에 이동한 거리의 합을 구한다.)

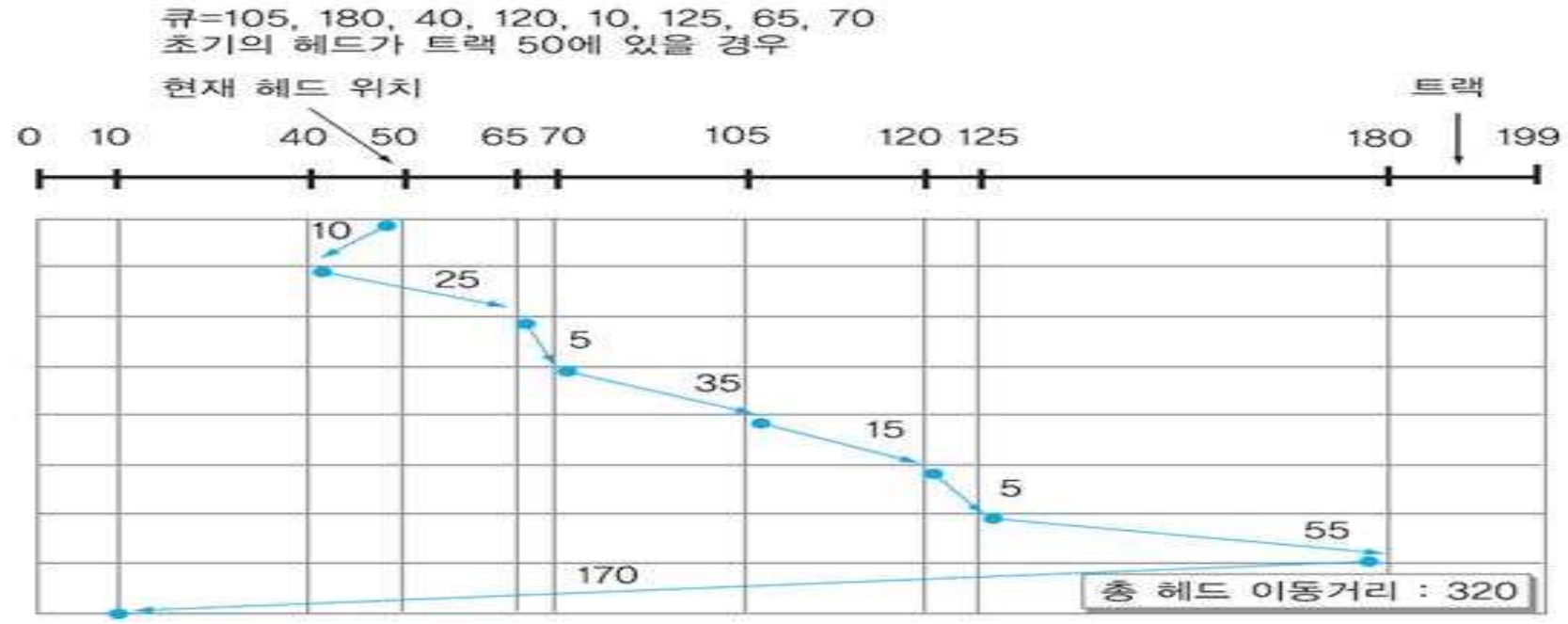


FCFS 스케줄링를 이용한 총 이동 순서 : 50 -> 105 -> 180 -> 40 -> 120 -> 10 -> 125 -> 65 -> 70
FCFS 스케줄링를 이용한 총 이동 거리 : D=55+75+140+80+110+115+60+5=640

디스크 스케줄링(SSTF 스케줄링)

- SSTF(Shortest Seek Time First) 스케줄링
  - 현재 헤드의 위치에 가장 가까운 요청을 먼저 서비스하는 기법
  - 본질적으로 SJF(Shortest Job First) 알고리즘 형태
  - 큐의 요구들을 처리하는 동안 헤드의 이동 거리 극소화
  - 단위 시간당 처리량 극대화 기법
  - 어떤 요청의 경우는 기근(starvation) 현상이 발생
  - 안쪽이나 바깥쪽의 트랙보다는 가운데 트랙이 더 많은 서비스를 받을 수 있기 때문에 응답 시간에 큰 편차가 생길 수 있음
  - FCFS 보다 처리율이 높고 평균 응답시간이 짧음

디스크 스케줄링(SSTF 스케줄링)

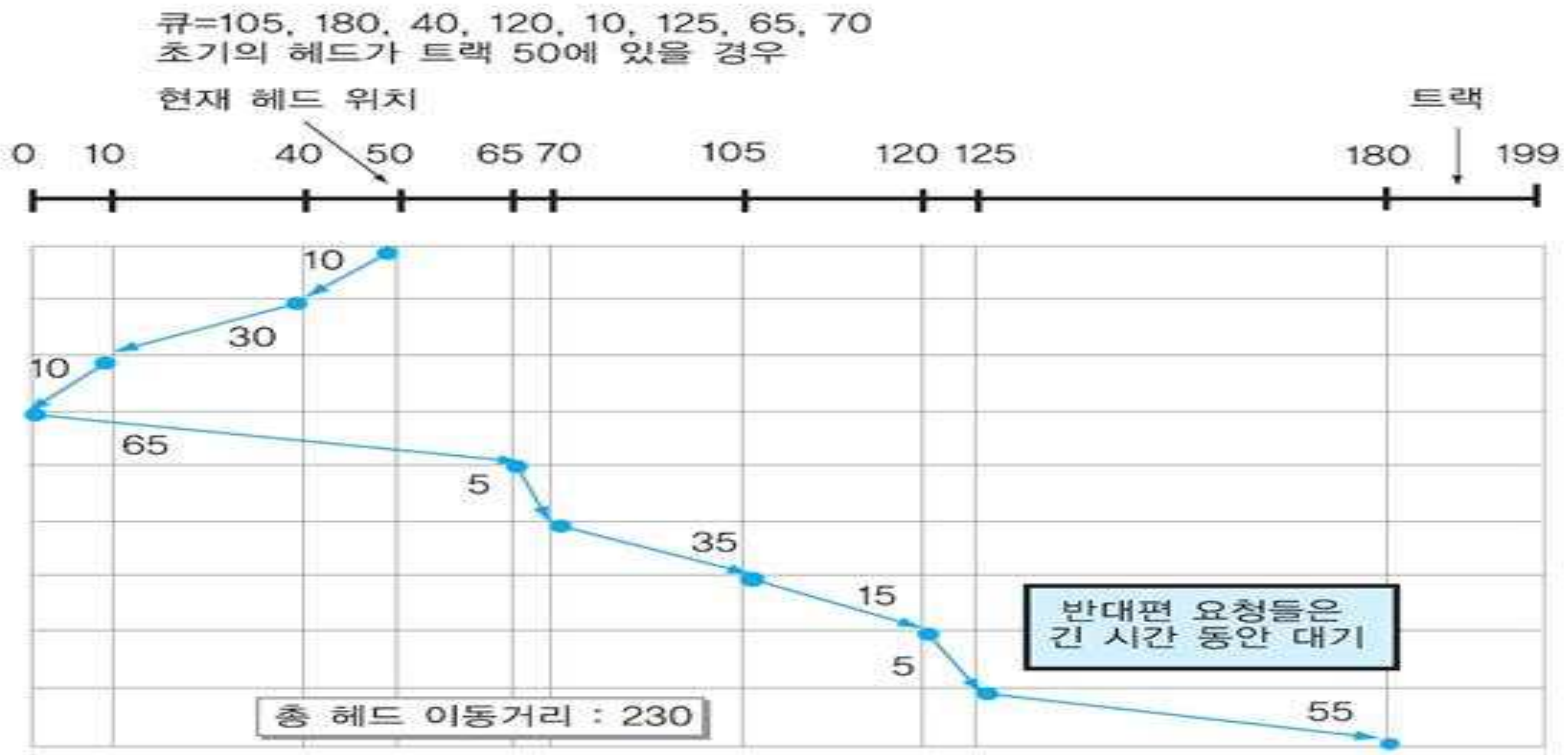


SSTF 스케줄링를 이용한 총 이동 순서 : 50 -> 40 -> 65 -> 70 -> 105 -> 120 -> 125 ->180 -> 10
SSTF 스케줄링를 이용한 총 이동 거리 : D=10+25+5+35+15+5+55+170=320

디스크 스케줄링(SCAN 스케줄링)

- SCAN 스케줄링
  - SCAN 알고리즘은 '진행 방향'상의 가장 짧은 거리에 있는 요청을 서비스하는 기법
  - 현재 큐에 대기중인 요구들 중에서, 현재 헤드의 진행 방향으로 현재 헤드의 위치와 가장 가까운 요구를 먼저 서비스하고, 마지막 실린더에 도착했을 때에 방향을 전환하는 기법
  - 헤드가 디스크의 한쪽 끝에서 시작하여 반대편 끝까지 움직이고 끝에서 역으로 하여 다시 계속 처리하는 방식
  - SCAN 알고리즘은 엘리베이터 동작과 유사하기 때문에 '엘리베이터' 알고리즘이라고 부름
  - SSTF 기법과 비슷한 기법
    - SSTF 기법 사용시 응답시간에 대한 예측성이 저하되는 단점 해결
    - 대체적으로 단위 시간당 처리량, 평균 응답 시간의 면에서 우수함
    - 실제 디스크 시스템에서 사용되는 스케줄링 기법들의 근간이 됨
- LOOK 스케줄링
  - 헤드가 각 방향의 트랙 끝까지 이동하지 않고 마지막 요청 트랙까지만 이동

디스크 스케줄링(SCAN 스케줄링)

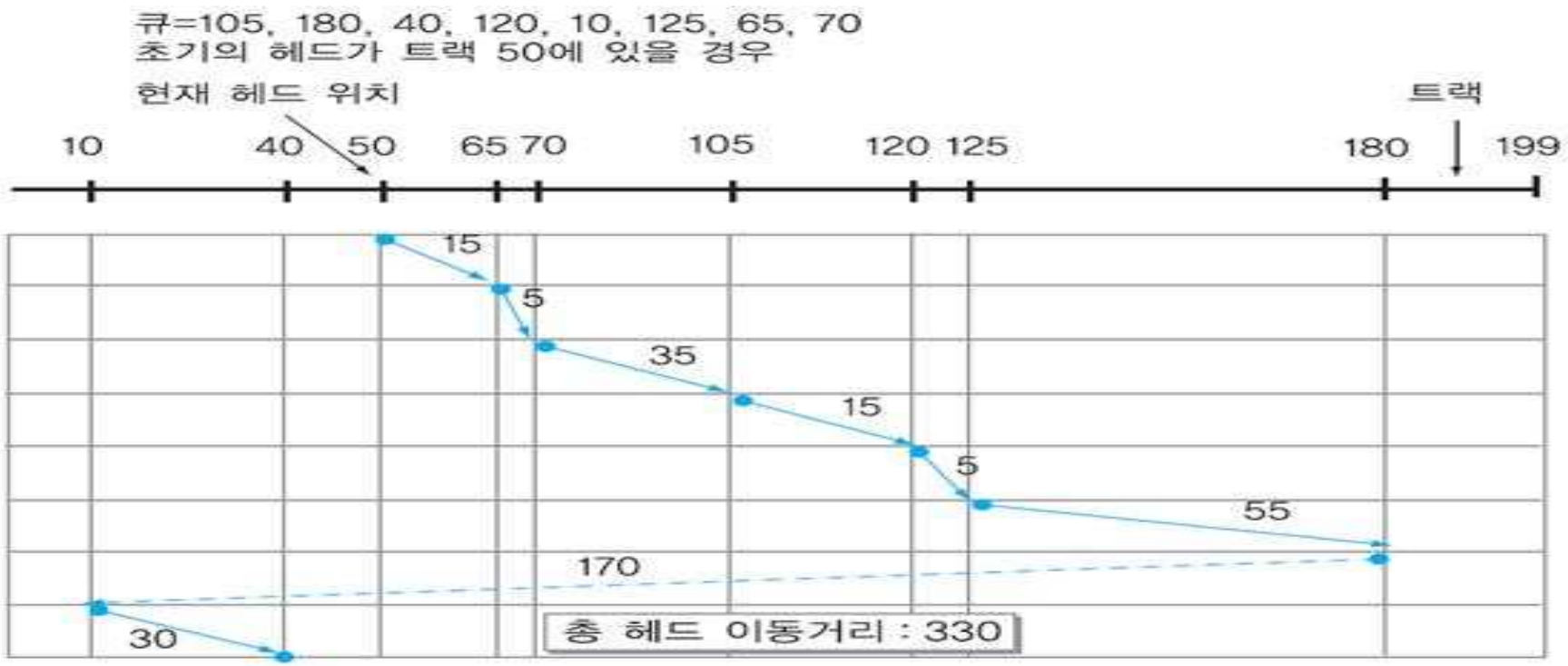


SCAN 스케줄링를 이용한 총 이동 순서 : 50 -> 40 ->10 -> 0 -> 65 -> 70 -> 105 ->120 -> 125->180
SCAN 스케줄링를 이용한 총 이동 거리 : D=10+30+10+65+5+35+15+5+55=230

디스크 스케줄링(C-SCAN 스케줄링)

- C-SCAN 스케줄링
  - C-SCAN(Circular SCAN)은 한쪽 방향으로 헤드를 이동하면서 '진행 방향'상의 가장 짧은 거리에 있는 요청을 처리
  - 한쪽 끝에 다다르면 다시 처음 시작 방향으로 이동하여 서비스하는 기법으로 끝에서 다시 같은 방향으로 처리를 진행하는 방식임
  - 미리 정해진 서비스 방향으로 헤드가 이동할 때에만 큐의 요구들을 처리함
    - 응답 시간의 분산이 매우 작은 기법
    - 응답 시간에 대한 예측성이 매우 높은 기법
  - C-SCAN은 가장 안쪽과 바깥쪽 트랙의 차별 대우를 개선하였고, 대기 시간의 편차가 매우 작음. 환형(circular)처리 모습 임
- C-LOOK 스케줄링
  - 헤드가 각 방향의 트랙 끝까지 이동하지 않고 마지막 요청 트랙까지만 이동

디스크 스케줄링(C-LOOK 스케줄링)



C-LOOK 스케줄링를 이용한 총 이동 순서 : 50 -> 65 ->70 -> 105 ->120 -> 125 -> 105 ->180 -> 10->40
C-LOOK 스케줄링를 이용한 총 이동 거리 : D=15+5+35+15+5+55+170+30=330

퀴즈 문제1

- 디스크에서 헤드가 70트랙을 처리하고 60트랙으로 이동해 왔다. 디스크 스케줄링 기법으로 SCAN 방식을 사용할 때 다음 디스크 대기 큐에서 가장 먼저 처리되는 트랙은?

디스크 대기 큐 : 20, 50, 95, 100

① 20 ② 50 ③ 95 ④ 100

- 디스크의 서비스 요청 대기 큐에 도착한 요청이 다음과 같을때 SSTF 스캐줄링 기법 사용 시 75번 트랙은 몇 번째로 서 비스를 받는가? (단, 현재 헤드위치는 100번 트랙으로 가정한다.)

105, 75, 58, 90, 35, 200, 64, 89

① 두 번째 ② 세 번째 ③ 네 번째 ④ 다섯 번째

학습 내용

- **5장 디스크 스케줄링과 파일시스템**
  - 5.4 파일 시스템
    - 파일 시스템의 개요
    - 파일 시스템 데이터의 계층 구조
    - 파일 시스템의 기능
    - 파일 시스템의 구조
    - 파일 시스템의 관리
    - 파일 시스템의 블로킹
  - 5.5 파일 구조
    - 순차 파일
    - 색인된 순차 파일
    - 직접 파일

5장 디스크 스케줄링과 파일시스템

5.1 개요
5.2 디스크 구조
5.3 디스크 스케줄링
5.4 파일 시스템
5.5 파일 구조
5.6 디렉토리 구조

파일 시스템의 개요

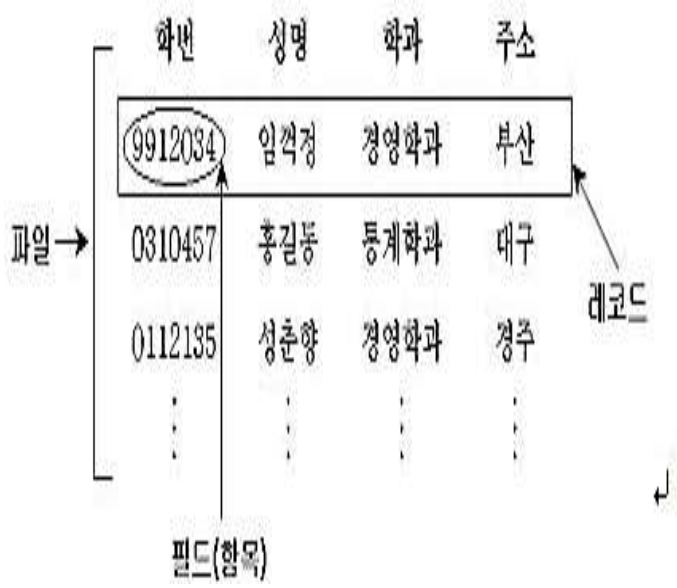
- 운영체제는 정보를 저장하는 논리적인 관점과 저장 장치의 물리적인 특성을 고려하여 논리적 저장 단위인 파일을 정의하고 메모리에 매핑 시키는 파일 시스템(시스템 소프트웨어) 기능 제공
- 파일 시스템은 파일을 구성하고 데이터 액세스를 관리하기 위해 두 부분으로 구성
  - 관련된 정보를 포함하는 실제적인 파일들의 집합체(데이터를 실제로 저장하는 파일들)
  - 시스템 내의 모든 파일에 대한 정보를 제공하고 이를 계층적으로 연결하는 디렉터리
  - 파일이란
    - 일반적으로 작성자와 사용자에 의해 그 의미가 정의된 비트, 바이트, 행(line), 또는 레코드들의 연속체
    - 하나의 파일에는 그 파일의 이름(name), 형태(type), 작성 시기, 작성자, 길이 등의 여러 속성이 있음

파일 시스템 개념

- 파일 시스템은 논리적인 저장 단위인 파일 자원을 관리하며, 파일의 생성과 삭제 등 파일을 액세스하고 제어하는 책임이 있는 소프트웨어
  - 파일의 저장 위치, 저장 방법, 저장 공간 활용, 파일 사용 할당, 파일 접근 횟수 등 복잡한 작업 수행하고 이러한 작업은 각 파일 정보를 기록한 디렉토리로 해당 파일을 관리 수행
  - 디스크 저장소를 관리(저장 공간 할당)하는 것과 메인 메모리와 다른 매체에 저장된 파일 데이터 액세스도 포함
  - 사용자가 파일을 생성·수정·삭제할 수 있도록 지원하며, 파일을 각 응용 프로그램에 가장 적합한 구조로 구성할 수 있도록 지원
  - 파일 읽기, 쓰기, 실행 등을 다양하게 액세스하도록 제어된 방법도 제공
  - 파일 시스템 설계
    - 사용자 수, 사용자당 평균 파일 수와 크기 등 사용자 정보가 필요하고, 해당 시스템에서 실행할 응용 프로그램의 특성을 이해하여 적합한 파일 구성과 디렉터리 구조 결정해야 함

파일 시스템 데이터의 계층 구조

- 데이터의 계층 구조
  - 필드(field) : 상호 관련 있는 문자들(숫자, 영문자, 특수 기호 등)의 집합
  - 레코드(record) : 서로 관련이 있는 필드들의 집합
    - 레코드 키(record key)란
      - 어떤 레코드를 다른 레코드로부터 식별하는 데 사용되는 제어 필드(control field)
  - 파일(file) :상호 관련 있는 레코드들의 집합
  - 데이터베이스(database) : 상호 관련 있는 파일들로 구성됨



데이터의 계층 구조의 예

파일 시스템의 기능

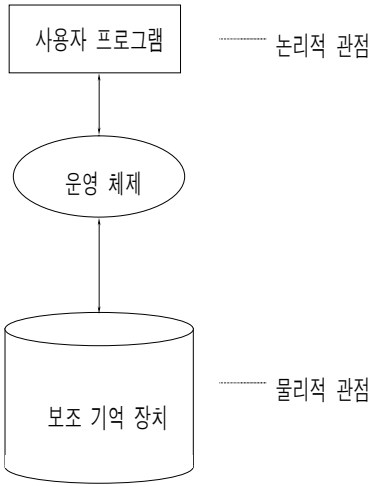
- **파일 시스템의 기능**
  - 사용자가 파일을 생성(create), 수정(modify), 삭제(delete) 가능
  - 파일 공유를 위한 적절한 제어 방법 제공
  - 여러 가지 접근 제어 방법을 제공
  - 다양한 형태로 파일을 재구성 방법 제공
  - 백업(backup)과 복구(recovery)를 위한 기능

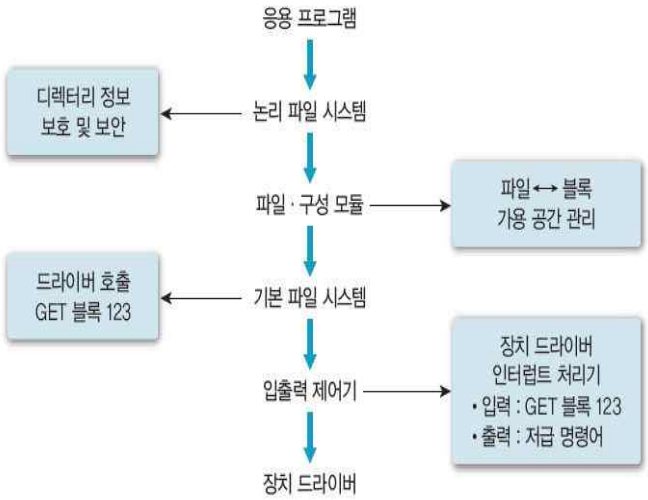
파일 시스템의 기능

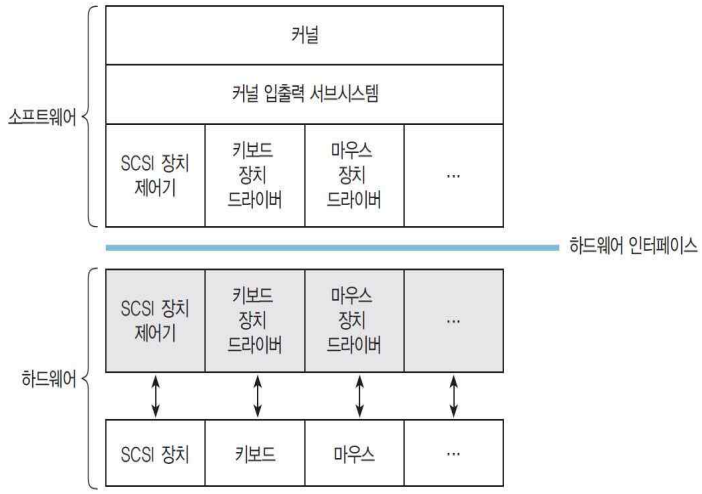
- **파일 시스템의 기능**
  - 사용자와 장치 간의 독립성(device independence) 유지하기 위하여 기호화된 이름(symbolic name) 제공
  - 정보가 안전하게 보호되고 비밀이 보장될 수 있도록
    정보의 암호화(encryption)와 복호화(decryption) 제공
  - 사용자에게 친숙한 인터페이스(user friendly interface) 제공

파일 시스템의 구조

파일 시스템 :정보를 저장하는 논리적인 관점과 저장 장치의 물리적인 특성을 고려하여 논리적 저장 단위인 파일을 정의하고 메모리에 매핑 시키는 시스템 소프트웨어 => 여러 계층(수준)으로 구성







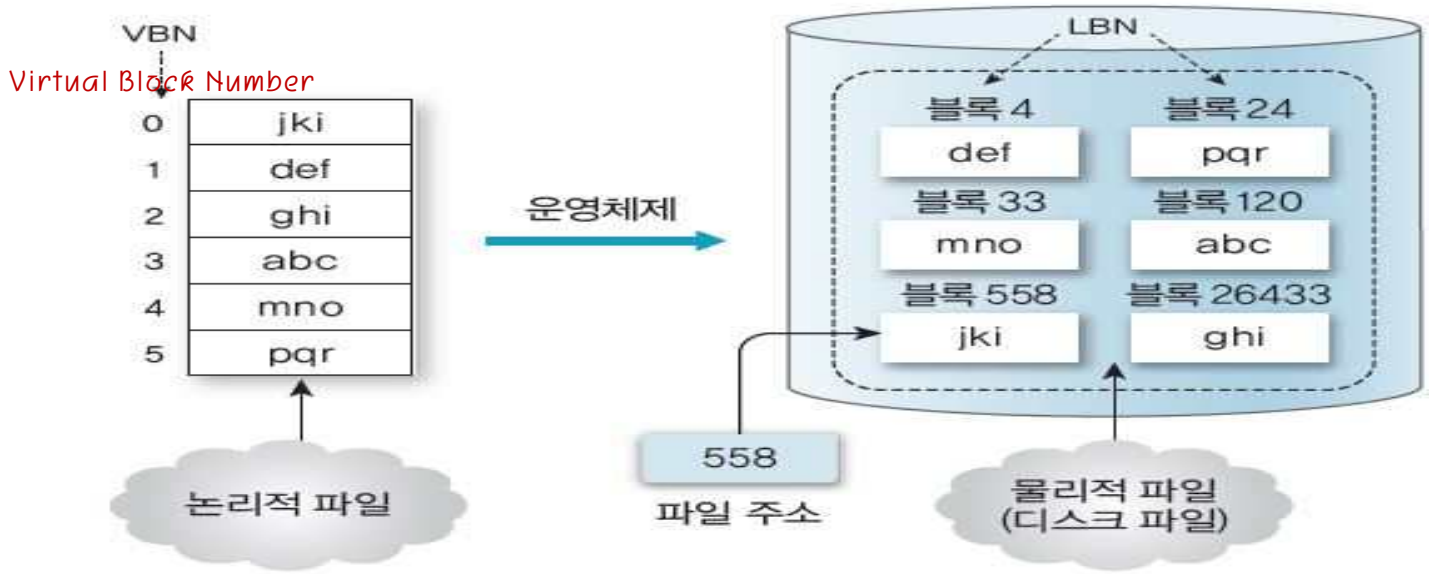
응용프로그램
fopen,
fgetc,
fputc,
fclose,
...

계층적 파일 시스템

커널 입출력 구조

파일 시스템의 관리

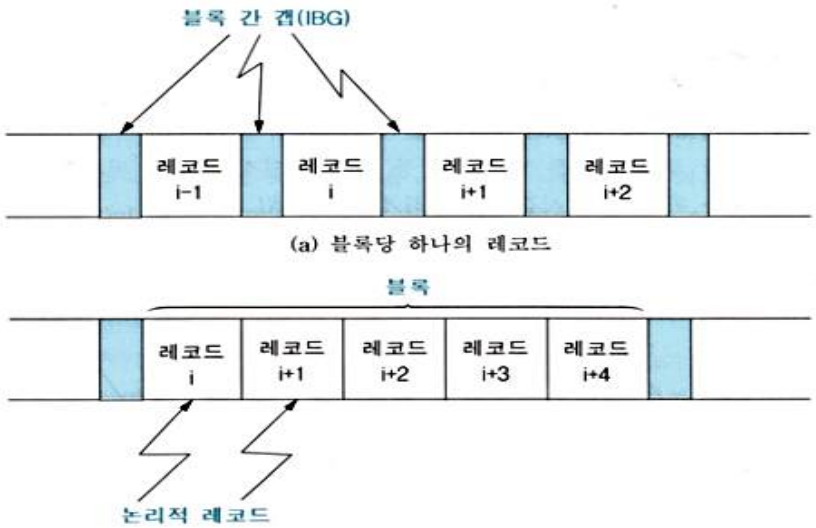
- **블록 : 메모리와 디스크 간 전송 단위**
  - 섹터 하나 이상으로 구성되며, 블록을 할당하는 방법은 사용하는 운영체제에 따라 다름
  - 운영체제는 파일에 속하는 정보를 블록에 유지하며, 0에서 시작하는 연속 정수로 일부 최대 수에 달하는 논리 블록 번호 사용
    - 논리 블록 번호는 물리적 디스크 주소(트랙, 실린더, 표면, 섹터)로 변환
  - 논리적 파일을 물리적 파일로 매핑하는 과정



논리적 파일을 물리적 파일로 매핑

파일 시스템의 블로킹

- 블로킹(Blocking)
  - 블록(Block) : 메모리와 디스크 간 전송 단위
  - 블로킹(Blocking): 파일 시스템에서 메모리(RAM)와 I/O효율을 위해 여러 개의 논리적 레코드를 하나의 물리적 레코드(블록)에 저장 시키는 것
    - 물리적 레코드(physical record)나 블록(block)
      - 기억매체에 출력되거나 기억매체로부터 입력되는 실제 정보의 단위
    - 논리적 레코드(logical record)
      - 사용자 관점에서 취급되는 자료 집단의 단위
    - 블로킹되지 않은 레코드(unblocked record)
      - 물리적 레코드가 단 하나의 논리적 레코드로 구성
    - 블로킹된 레코드(blocked record)
      - 여러 개의 논리적 레코드가 하나의 물리적 레코드를 구성

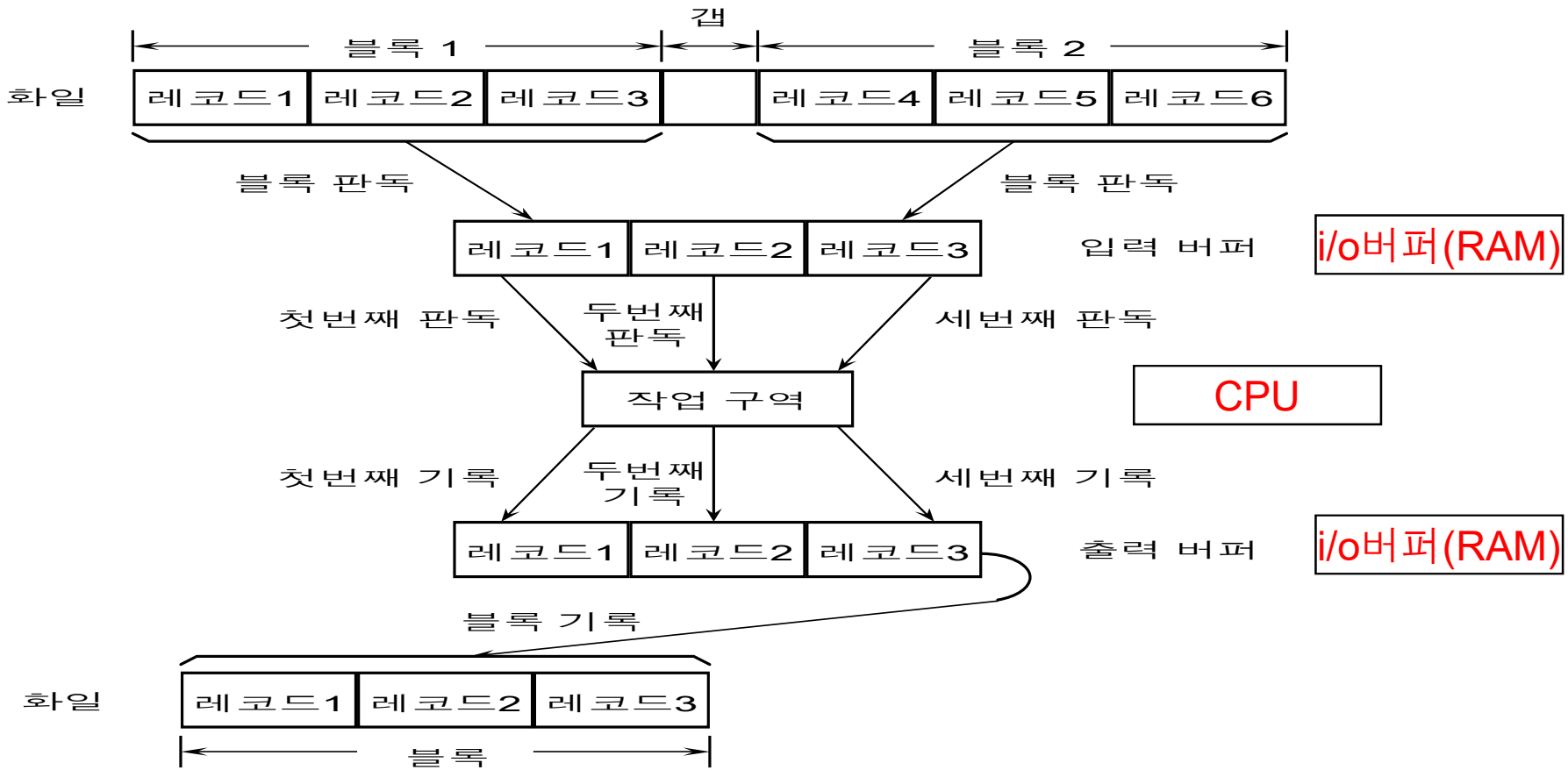


파일 시스템의 블로킹

- **블로킹(Blocking)**
  - 고정길이 레코드(fixed-length record)
    - 구성된 파일에서의 레코드 길이는 모두 같음
  - 가변길이 레코드(variable length record)
    - 구성된 파일에서의 레코드 길이는 다양
    - 최대 크기는 블록의 크기와 동일

파일 시스템의 블로킹

- 블로킹을 이용한 레코드 판독과 기록



파일 구조

- 파일의 구조
  - 파일을 구성하는 레코드들이 보조기억장치에 배치되는 방법
  - 파일에 대한 접근 방법과도 밀접한 관계
  - 파일의 구조의 종류
    - 순차 파일, 색인된 순차 파일, 직접 파일

- 순차 파일(sequential file)
  - 가장 단순한 방법
  - 논리적인 레코드를 물리적인 순서에 따라 순차적으로 저장하고 검색하도록 저장
  - 디스크, 자기 테이프, 프린터로 된 출력 등에 주로 사용
  - 순차접근 기억장치(SASD : Sequential Access Storage Device)에 저장
  - 일괄처리에서 많이 사용
  - 장점은 다음 레코드에의 빨리 접근할 수 있으며, 단점은 순차 파일에 대한 접근 방식이 파일에 저장된 레코드 순서와 다를 때 프로그램의 접근 속도가 저하 된다는 것임

파일 구조

- 파일의 구조
  - 색인된 순차 파일(indexed sequential file)
    - 순차 및 직접 접근을 모두 처리할 수 있는 파일 구조
    - 레코드는 각 레코드의 키 값에 따라 논리적 순으로 배열
    - 레코드는 키 값에 순차적으로 접근될 수도 있고, 시스템에 의해 관리되는 인덱스 블록의 검색을 통하여 직접 접근도 가능
    - 보통 디스크와 같은 직접 접근 기억장치(DASD : Direct Access Storage Device)에 저장
    - 일괄처리 및 대화형 처리를 목적으로 하는 파일을 지원
    - 장점은 융통성이 많고 검색 성능도 우수하며, 단점은 설계 시 고려할 사항이 많음

파일 구조

- 파일의 구조
  - 직접 파일(direct file)
    - 다른 레코드를 참조하지 않고 임의 레코드를 직접 접근할 수 있는 파일 구조임
    - 특정 응용 분야에 적합한 순으로 레코드를 직접 접근 기억장치에 저장
    - 키 값에서 보조기억장치의 주소로 사상시키는 사상함수가 필요
    - 특정 레코드를 저장하고 있는 기억장소의 주소를 신속하게 계산해야
    - 대화형 처리에 유용
    - 디스크와 같은 직접 접근 기억장치(DASD : Direct Access Storage Device)에 저장
    - 장점은 특정 레코드의 검색, 삽입, 수정, 삭제가 쉬우며, 단점은 키 값의 순서에 의한 순차 검색이 어려움

퀴즈 문제2

- 파일 시스템의 기능으로 옳지 않은 것은?

① 여러 종류의 점근 제어 방법을 제공
② 파일의 생성, 변경, 제거
③ 네트워크 제어
④ 파일의 무결성과 보안을 유지 할 수 있는 방안 제공

- 파일 시스템에서 메모리(RAM)와 I/O효율을 위해 여러 개의 논리적 레코드를 하나의 물리적 레코드(블록)에 저장 시키는 것을 무엇이라 하는가?

① 링킹(linking) ② 블로킹(Blocking) ③ 스와핑(Swapping) ④ 통합(Coalescing)

- 파일 구조는 파일을 구성하는 레코드들이 보조기억장치에 배치되는 방식을 말한다. 이에 관한 설명 중 틀린 것은?

① 순차 파일의 레코드들은 반드시 연속된 물리적 저장 공간에 저장될 필요는 없다.
② 인덱스된 순차 파일에서 레코드는 각 레코드의 키 값에 따라 논리적 순서대로 배열 되어 있다.
③ 직접 파일은 레코드가 직접 액세스 기억장치의 물리적 주소를 통해 직접 액세스한다.
④ 분할된 파일은 여러 개의 순차 서브파일로 구성된 파일이다.

학습 내용

- **5장 디스크 스케줄링과 파일시스템**
  - 5.5 파일 구조
    - 파일의 유형
    - 파일의 연산
    - 파일 디스크립터
  - 5.6 디렉토리 구조
    - 디렉터리의 개념
    - 디렉터리 구조

5장 디스크 스케줄링과 파일시스템

5.1 개요
5.2 디스크 구조
5.3 디스크 스케줄링
5.4 파일 시스템
5.5 파일 구조
5.6 디렉토리 구조

파일 시스템 개념

- 파일 시스템은 논리적인 저장 단위인 파일 자원을 관리하며, 파일의 생성과 삭제 등 파일을 액세스하고 제어하는 책임이 있는 소프트웨어
- 파일 시스템의 기능
  - **파일 구성** : 사용자가 각 응용 업무에 적합하게 파일을 구성할 수 있게 함
  - **파일 관리** : 파일의 생성, 수정, 저장, 참조, 제거, 보호 기능을 제공하며, 파일을 공유하는 다양한 액세스 제어 방법을 제공
  - **보조 메모리 관리** : 다양한 형태의 저장장치에 입출력 지원, 2차 저장장치에 파일 저장할 공간 할당
  - **파일 무결성 보장** : 파일에 저장된 정보를 손상하지 않도록 보장
  - **파일 액세스 방법 제공** : 저장된 데이터(파일) 읽기, 쓰기, 실행, 이들을 여러 형태로 조합한 다양한 액세스 제어 방법을 제공
  - **장치 독립성 유지** : 물리적 장치 이름을 사용하는 대신 기호화된 이름(예를 들어m, yDirectory:myFile.txt)으로 자신의 파일을 참조하여 장치와 독립성 유지
  - **파일 백업과 복구** : 사고로 정보를 손실하거나 고의로 손상하는 일을 방지하려고 데이터 사본(중복)을 생성하는 백업backup과 손상된 데이터를 복구recovery할 수 있는 기능
  - **파일 보호** : 정보를 암호화하고 해독할 수 있는 기능을 제공하여 정보의 안전과 비밀 보장
  - **정보 전송** : 사용자가 파일 간에 정보 전송 명령을 내릴 수 있게 한다.

파일의 유형

파일의 유형

- 파일의 유형으로 파일의 내부 구조 형태 짐작 가능

- 파일의 세 가지 유형
  - 일반(정규) 파일
    - 가장 일반적인 파일과 데이터를 포함하는 데 사용, 텍스트나 이진 형태
  - 디렉터리 파일
    - 모든 유형의 파일에 액세스할 수 있는 정보 포함하나, 실제 파일 데이터는 포함하지 않음
  - 특수 파일
    - 시스템 장치를 정의하거나 프로세스로 생성한 임시 파일로 파이프라고 하는 FIFO(선입선출), 블록, 문자 이에 해당

- 운영체제가 여러 파일의 구조를 지원하면 크기가 커져 복잡해진다는 단점
  - 너무 적은 파일구조를 지원하면 프로그래밍 어려움
- 파일 이름에 마침표를 넣어서 구분하여 파일 유형 표현
  - 파일 이름은 크게 순수 이름과 확장자로 구성

파일의 유형

- 파일 유형과 확장자 예

파일 유형과 확장자 예

파일 유형	확장자	기능
실행 가능	exe, com, bin 등	이진 수행 가능 프로그램
소스 코드	c, p, pas, f77, asm, a	다양한 언어로 된 소스 코드
배치	bat, sh	명령어 해석기에서 명령
문서	txt, doc	텍스트 데이터, 서류
워드 프로세서	wod, hwp, doc, 기타	다양한 워드 프로세서 형식
라이브러리	lib, a, DLL	프로그래머들이 사용하는 라이브러리 루틴
백업, 보관	arc, zip, tar	관련된 파일을 하나로 묶어서 보관하는 것으로 저장용 압축도 가능

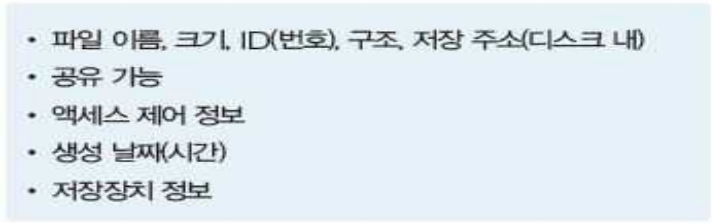
파일의 연산

파일의 연산

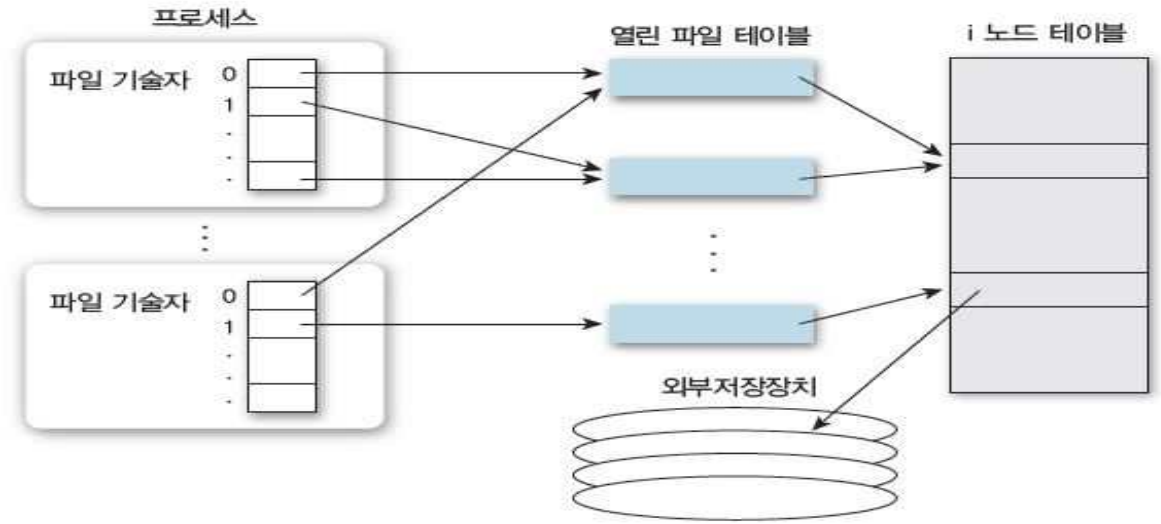
- 운영체제는 파일에서 다양한 연산 지원하여 컴퓨터에서 파일을 사용하도록 함
  - 파일 생성하기
  - 파일 열기
  - 파일 쓰기
  - 파일 읽기
  - 파일 재설정
  - 파일 삭제
  - 파일 크기 조절
  - 속성 설정
  - 파일 이름 바꾸기
  - 파일 삭제

파일 디스크립터

- **파일 디스크립터(descriptor(기술자))**
  - 파일을 액세스하는 동안 운영체제에 필요한 정보를 모아 놓은 자료구조



(a) 파일 디스크립터의 내용



(b) 파일 디스크립터의 역할

파일 디스크립터

-파일마다 독립적으로 존재

-(b)와 같이 파일을 열 때 프로세스가 생성

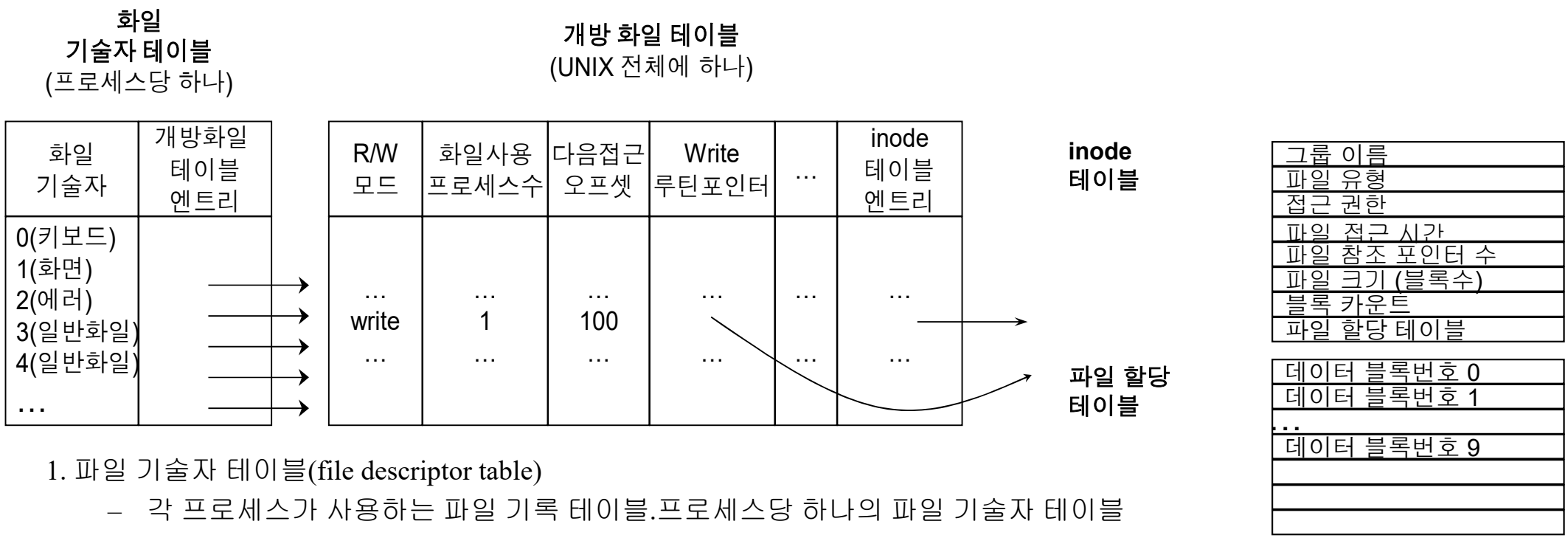
-파일 디스크립터는 음이 아닌 고유의 정수로, 파일을 액세스하려고 열린 파일(테이블)을 식별하는 데 사용

-시스템에 따라 구조가 다를 수 있으며, 파일 시스템이 관리하여 사용자 직접 참조 불가

-디스크의 모든 파일이나 디렉터리는 디스크에 저장하므로, 열린 각 파일이나 디렉터리의 파일 디스크립터는 디스크에 저장했다가 파일을 열면 메모리에 복사

-파일을 닫거나 프로세스 종료할 때 폐기

Unix I/O를 위한 파일 기술자



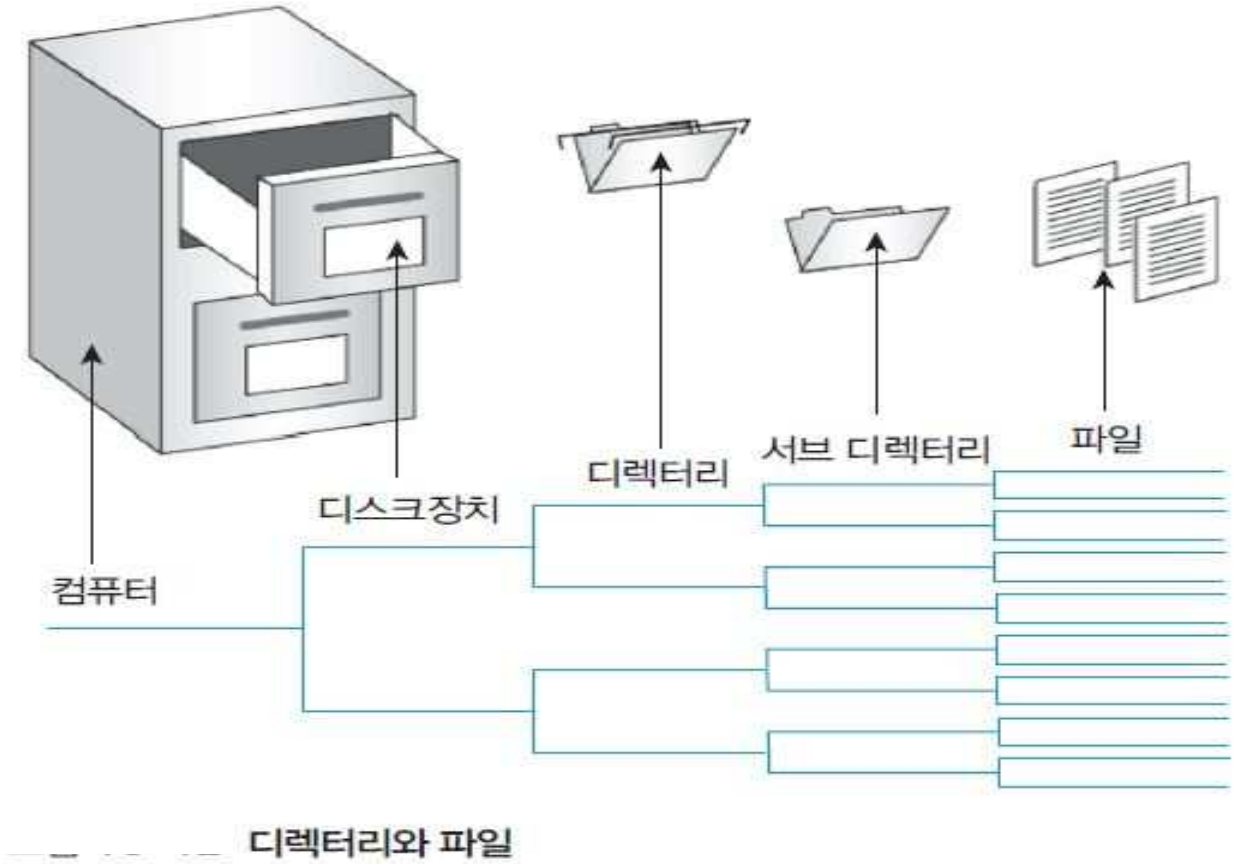
1. 파일 기술자 테이블(file descriptor table)
   - 각 프로세스가 사용하는 파일 기록 테이블.프로세스당 하나의 파일 기술자 테이블
2. 개방 파일 테이블 (open file table)
   - 현재 시스템이 개방하여 사용중인 모든 파일에 대한 엔트리로 구성
   - Unix 시스템 전체에 하나
3. 파일 할당 테이블(file allocation table)
   - 파일에 할당된 디스크 블록 리스트를 포함
4. 인덱스 노드 테이블(index node table)
   - 현재 사용되고 있는 파일 당 하나의 엔트리(inode)로 구성

파일을 관리하는 디렉터리 개념

- **디렉터리의 개념**
  - 디렉터리를 유지하고 관리하여 디스크 등에 저장된 파일을 관리
  - 디렉터리의 구분
    - 장치 디렉터리 : 각 실제 장치에 저장. 장치에 있는 파일의 물리적 속성, 파일의 위치, 파일의 크기와 할당 과정 등을 나타냄
    - 파일 디렉터리 : 모든 파일의 논리적 구성으로 이름, 파일 유형, 소유한 사용자, 계정 정보, 보호 액세스 코드 등을 기술
  - 파일 시스템에서 다른 파일들의 이름과 위치 정보(파일 인덱스)를 담은 파일로, 다른 파일들과 달리 사용자 데이터 저장하지 않음
  - 장치의 범위를 확장할 수 있고 다른 디스크 장치들을 포함 가능
  - 디렉터리에 있는 정보 중 일부는 사용자나 응용 프로그램이 이용 가능하나, 대부분 시스템 루틴이 사용자에게 간접적으로 제공

디렉터리의 개념

- 디렉터리와 파일의 관계



디렉터리와 파일

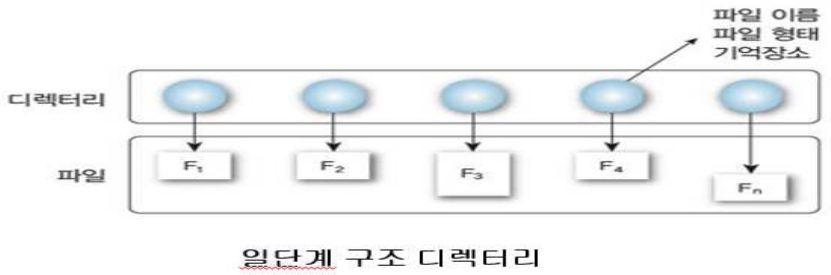
- 디렉터리의 파일 정보
  - 파일 이름
  - 파일 형태
  - 위치
  - 크기
  - 현재 위치
  - 보호
  - 사용 수
  - 시간, 날짜, 처리 식별

디렉터리 구조

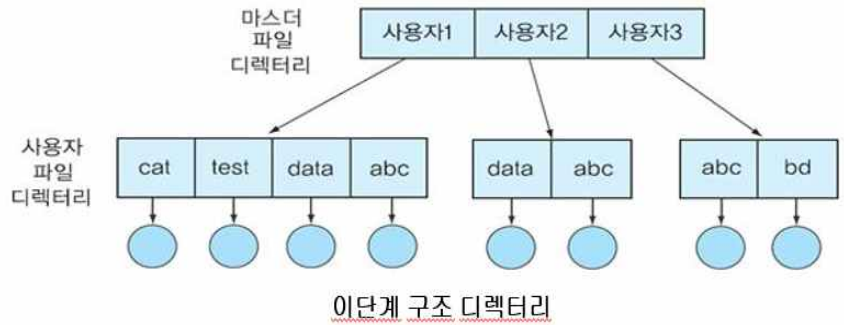
- 파일 시스템 내부에 있는 많은 파일들을 조직화하는 메커니즘
- 디렉터리에서 실행되어야 할 기능
  - 탐색(search)
    - 특정 파일에 대한 항목을 발견하기 위해서는 디렉터리를 찾는 기능
  - 파일 생성(file create)
    - 새로운 파일들을 생성해야 할 필요가 있으면 이를 디렉터리에 첨가하는 기능
  - 파일 삭제(file delete)
    - 파일이 더 이상 필요 없을 때 디렉터리로부터 삭제 기능
  - 디렉터리 리스트(directory list)
    - 디렉터리의 내용들을 보여줄 수 있어야 하고, 그 리스트 내의 각 파일에 대한 디렉터리 항목의 내용을 보여줄 수 있어야 하는 기능
  - 백업(back up)
    - 만약의 경우에 대비하여 일반적으로 파일들을 자기 테이프 등에 복사하는 기능

디렉터리 구조

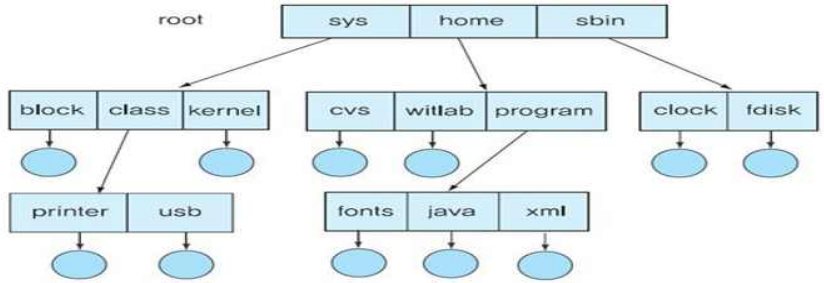
- 디렉토리 구조의 종류
  - 일단계 구조 디렉터리
    - 가장 간단한 구조
    - 모든 파일들을 같은 디렉터리 내에 위치
    - 같은 디렉터리 내에 모두 상이한 이름을 가져야 함
  - 이단계 구조 디렉터리
    - 2개의 단계인 마스터 파일 디렉터리(MFD : Master File Directory), 사용자 파일 디렉터리(UFD : User File Directory)로 구성
    - 일 단계 디렉터리에서의 단점인 서로 다른 사용자들 간의 파일명의 혼란을 해결
  - 트리구조 디렉터리
    - 이단계 디렉터리 구조의 확장을 위한 일반화된 방법
    - 디렉터리 , 또는 서브디렉터리는 일단의 파일이나 또 다른 서브디렉터리들을 가짐
    - UNIX파일 시스템은 트리로 구성
    - 경로 이름은 절대 경로와 상대 경로 방법으로 이용
  - 비 순환 구조 디렉터리
  - 일반적 그래프 구조 디렉터리



일단계 구조 디렉터리



이단계 구조 디렉터리



트리 구조 디렉터리