쉽게 배우는

운영체제

2판

Chapter 10 입출력 시스템과 저장장치

차례

학습목표

- 입출력 버스의 구조를 파악한다.
- 입출력 과정에서 직접 메모리 접근, 인터럽트, 버퍼링이 어떻게 적용되는지 알아본다.
- 디스크 저장장치의 종류, 각 장치의 구조, 데이터 전송 시간을 알아본다.
- 디스크 저장장치의 여러 가지 관리 기법을 이해한다.
- 디스크 스케줄링 기법을 이해하고 각각의 장단점을 파악한다.
- RAID의 필요성을 이해하고 구성 방식에 따른 종류를 알아본다.

1-1 입출력장치와 채널

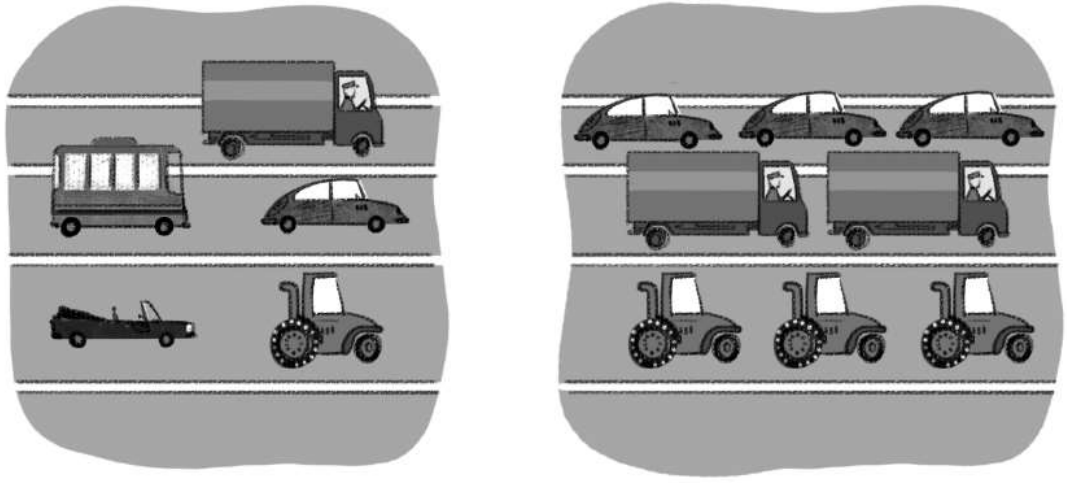
■ 주변장치

- 주변장치는 저속 주변장치(키보드, 마우스 등)와 고속 주변장치(그래픽카드, 하드디스크 등)로 나뉨
- 주변장치는 메인보드 내의 버스로 연결
- 버스에는 많은 종류의 장치가 연결되어 1개만 사용하면 병목 현상 발생하므로 이를 해결하기 위해 여러 개의 버스를 묶어서 사용
- 이때 데이터가 지나다니는 하나의 통로를 채널이라고 부름

1-1 입출력장치와 채널

■ 채널 공유와 채널 분리

- 채널을 모든 주변장치가 공유하면 전체적으로 데이터 전송 속도가 느려짐
- 전송 속도가 비슷한 장치끼리 묶어서 장치별로 채널을 할당하면 전체 데이터 전송 속도를 향상할 수 있음



(a) 채널 공유 (b) 채널 분리

그림 10-1 채널 공유와 채널 분리

1-2 입출력 버스의 구조

■ 초기의 구조

- 모든 장치가 하나의 버스로 연결
- CPU가 작업을 진행하다가 입출력 명령을 만나면 직접 입출력장치에서 데이터를 가져오는 폴링(polling) 방식 이용

그림 10-2 초기 입출력 버스의 구조

1-2 입출력 버스의 구조

■ 입출력 제어기를 사용한 구조

- 버스는 메인버스와 입출력 버스의 2개 채널로 나뉨
  - 메인버스 : 고속으로 작동하는 CPU와 메모리가 사용
  - 입출력 버스 : 주변장치가 사용
- 입출력 제어기를 사용하면 느린 입출력장치로 CPU와 메모리 작업이 느려지는 것을 막을 수 있어 전체 작업 효율 향상

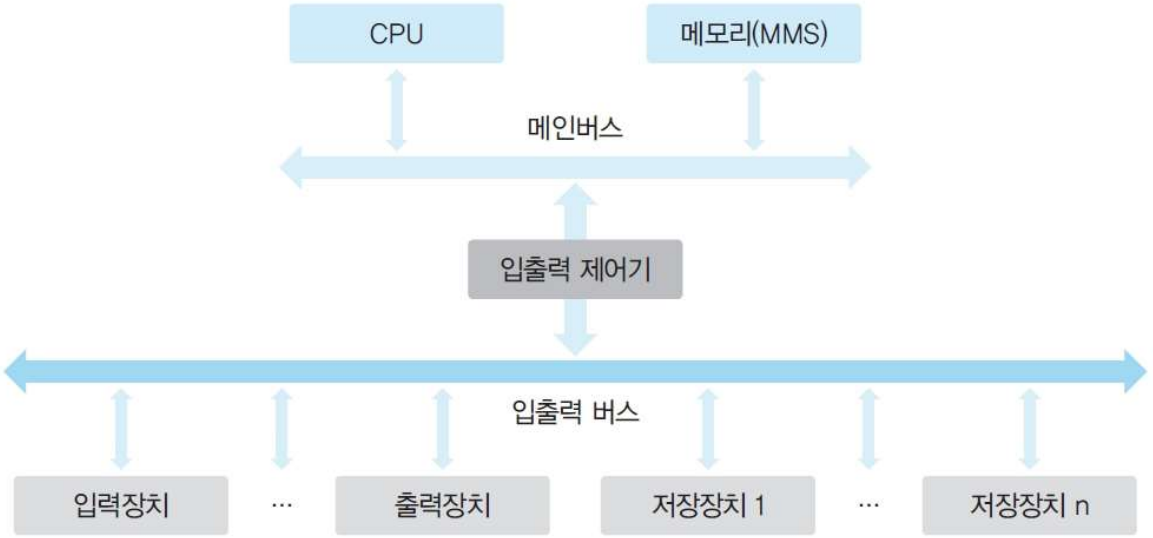


그림 10-3 입출력 제어기를 사용한 입출력 버스의 구조

1-2 입출력 버스의 구조

■ 입출력 버스의 분리

- 입출력 제어기를 사용하면 작업 효율을 높일 수 있지만, 저속 주변장치 때문에 고속 주변장치의 데이터 전송이 느려지는 문제가 있음
- 이를 해결하기 위해 입출력 버스를 고속 입출력 버스와 저속 입출력 버스로 분리하여 운영
- 고속 입출력 버스에는 고속 주변장치, 저속 입출력 버스에는 저속 주변장치 연결
- 두 버스 사이의 데이터 전송은 채널 선택기(channel selector)가 관리
- 입출력 버스로 감당하기 어려워진 그래픽카드는 입출력 버스에서 분리하고 메인 버스에 바로 연결하여 사용
- 결론적으로 현대의 컴퓨터는 CPU와 메모리를 연결하는 메인버스, CPU와 그래픽 카드를 연결하는 그래픽 버스, 고속 입출력 버스와 저속 입출력 버스를 사용

1-2 입출력 버스의 구조

■ 입출력 버스의 분리

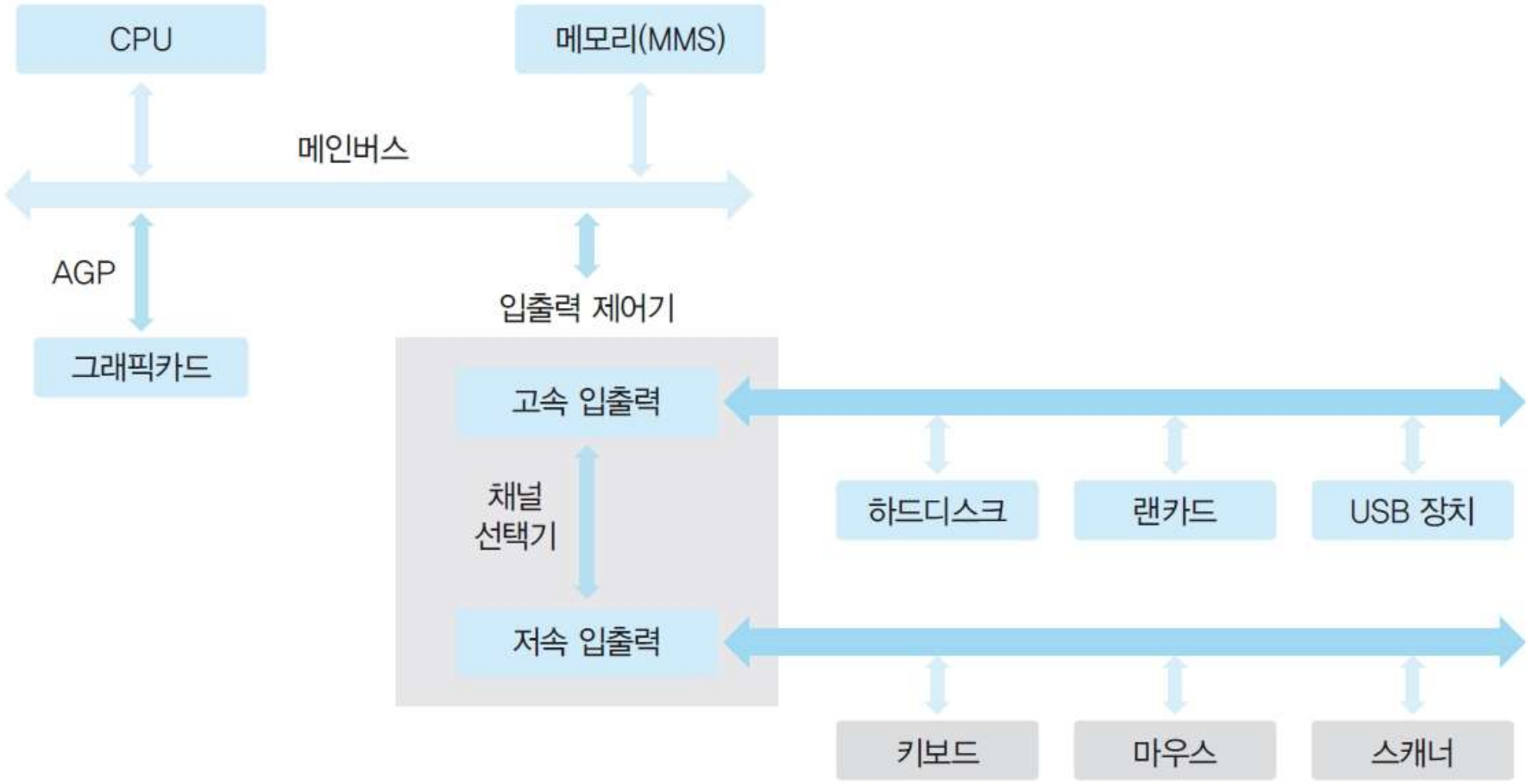


그림 10-4 입출력 버스의 분리

1-3 직접 메모리 접근

■ 직접 메모리 접근(DMA)

- CPU 도움 없이도 메모리에 접근할 수 있도록 입출력 제어기에 부여된 권한
- 입출력 제어기에는 직접 메모리에 접근하기 위한 DMA 제어기가 있음
- 입출력 제어기는 여러 채널에 연결된 주변 장치로부터 전송된 데이터를 적절히 배분하여 하나의 데이터 흐름을 만듦
- 채널 선택기는 여러 채널에서 전송된 데이터 중 어떤 것을 메모리로 보낼지 결정

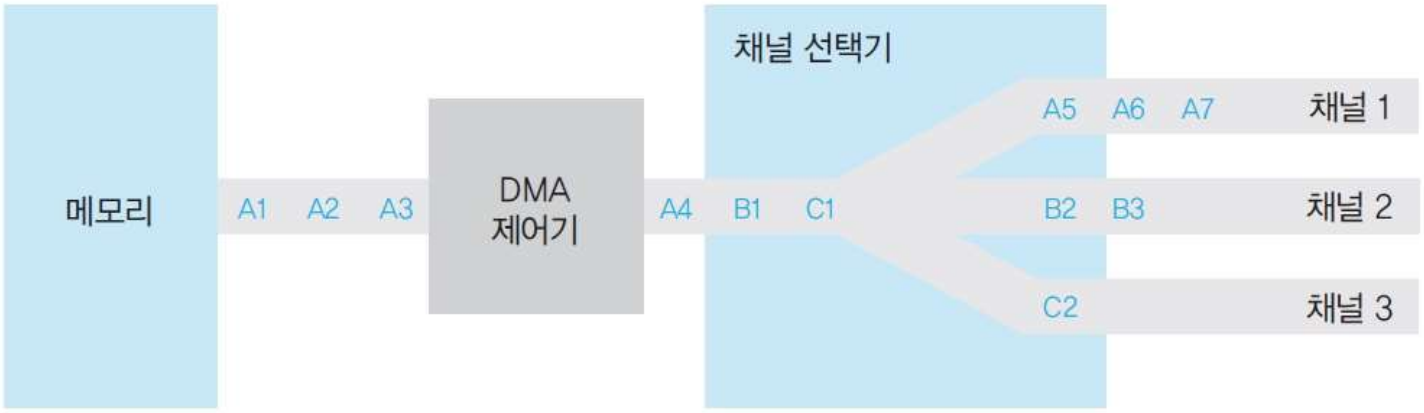
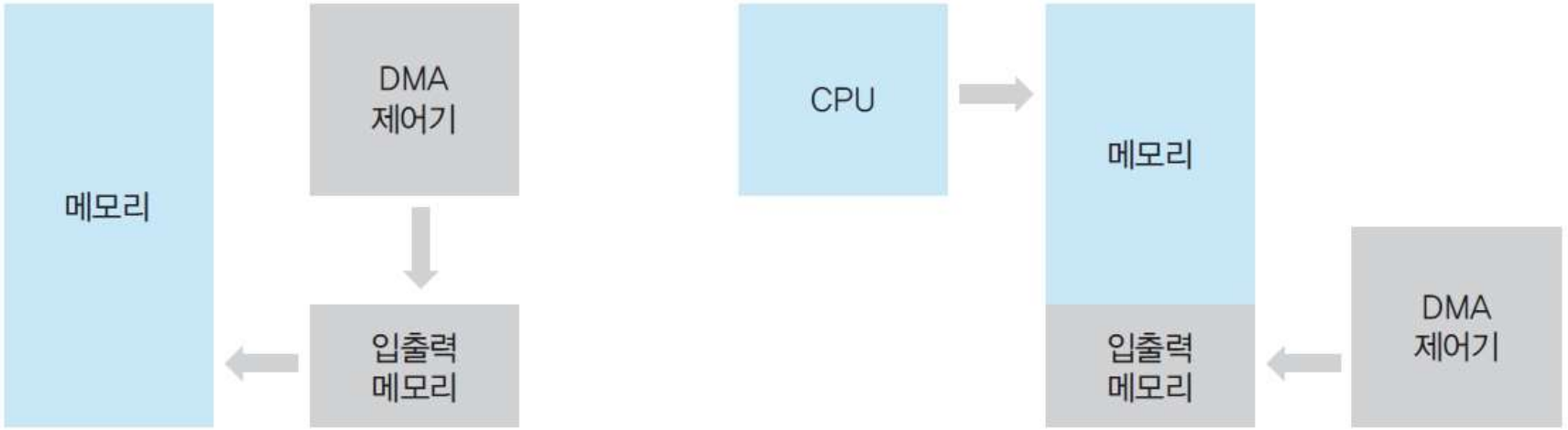


그림 10-5 입출력 제어기의 구조

1-3 직접 메모리 접근

■ 메모리 공간 분할

- CPU와 DMA의 작업 공간이 겹치는 것을 방지하기 위해 과거에는 DMA 제어기가 전송하는 데이터를 '입출력 메모리'라는 별도의 메모리에 보관
- 현재는 CPU가 작업하는 공간과 DMA 제어기가 데이터를 옮기는 공간을 분리하여 메인 메모리를 운영, 이를 메모리 맵 입출력(MMIO; Memory Mapped I/O)이라고 부름



(a) 별도의 입출력 메모리 사용

(b) 메모리 맵 입출력

그림 10-6 메모리 공간 분할

1-4 인터럽트

■ 입출력과 인터럽트

- 인터럽트는 주변장치의 입출력 요구나 하드웨어의 이상 현상을 CPU에 알려주는 신호
- 각 장치에는 IRQ라는 고유의 인터럽트 번호가 부여됨
- 인터럽트가 발생하면 CPU는 IRQ를 보고 어떤 장치에서 인터럽트가 발생했는지 파악

■ 인터럽트의 종류

- 외부 인터럽트: 입출력장치로부터 오는 인터럽트뿐 아니라 전원 이상이나 기계 오류로 발생하는 인터럽트를 포함하므로 하드웨어 인터럽트로도 부름
- 내부 인터럽트: 프로세스 잘못이나 예상치 못한 문제로 발생, 예외 상황(exception) 인터럽트라고도 함
- 시그널(signal): 사용자가 직접 발생시키는 인터럽트

1-4 인터럽트

■ 인터럽트 벡터

- 어떤 인터럽트가 발생했는지 파악하기 위해 사용하는 자료구조
- 인터럽트 벡터의 값이 1이면 인터럽트가 발생했다는 의미

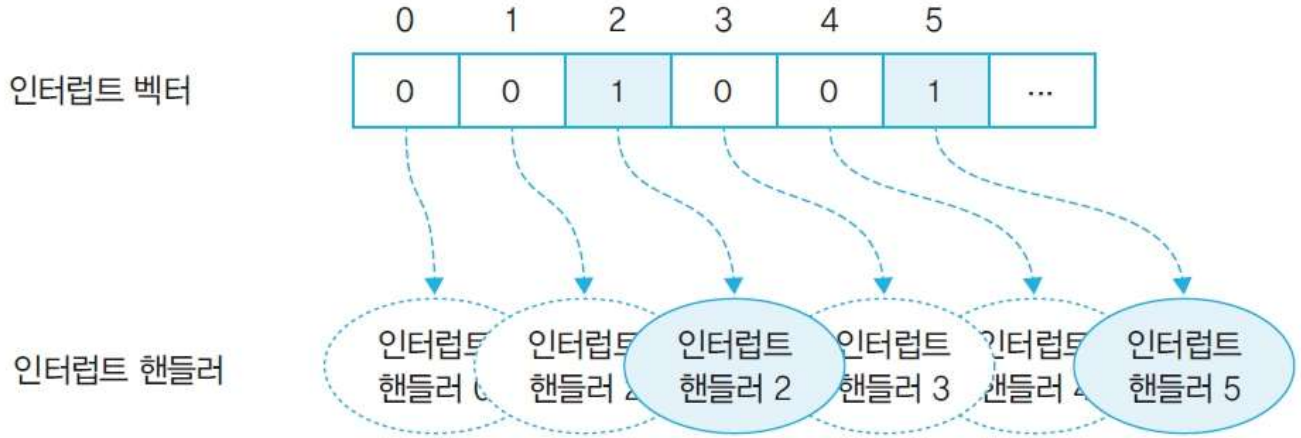


그림 10-7 인터럽트 벡터와 인터럽트 핸들러

■ 인터럽트 핸들러

- 인터럽트의 처리 방법을 함수 형태로 만들어놓은 것
- 운영체제는 인터럽트가 발생하면 인터럽트 핸들러를 호출하여 작업함
- 사용자 인터럽트인 시그널은 자신이 만든 인터럽트 핸들러를 등록할 수도 있음

1-5 단일 버퍼와 이중 버퍼

■ 버퍼

- 단일 버퍼(single buffer) 보다 이중 버퍼(double buffer)가 버퍼 운용에 유리
- 단일 버퍼는 데이터를 담는 작업과 퍼 가는 작업을 동시에 하기 어려움
- 이중 버퍼는 한 버퍼는 데이터 담고 다른 버퍼는 가져가는 용도로 쓸 수 있음

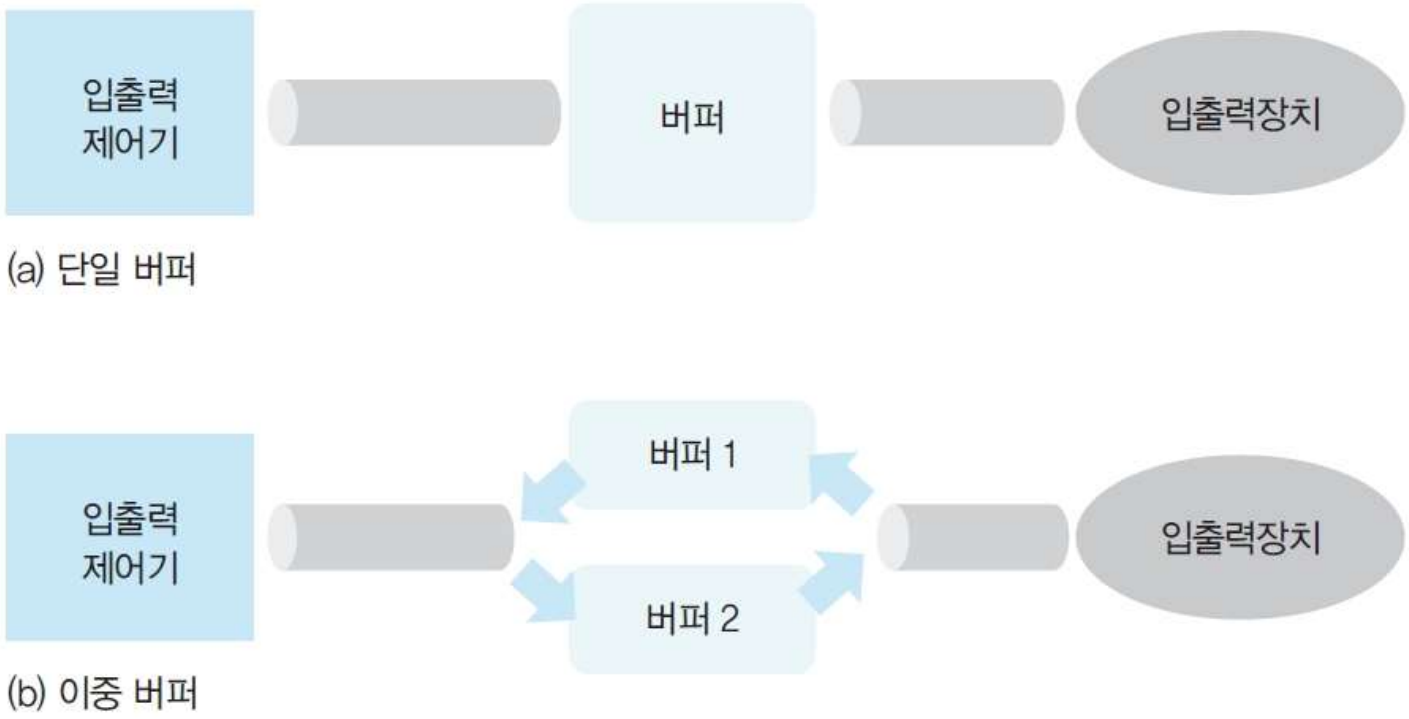


그림 10-8 단일 버퍼와 이중 버퍼

2-1 저장장치의 종류

■ 하드디스크

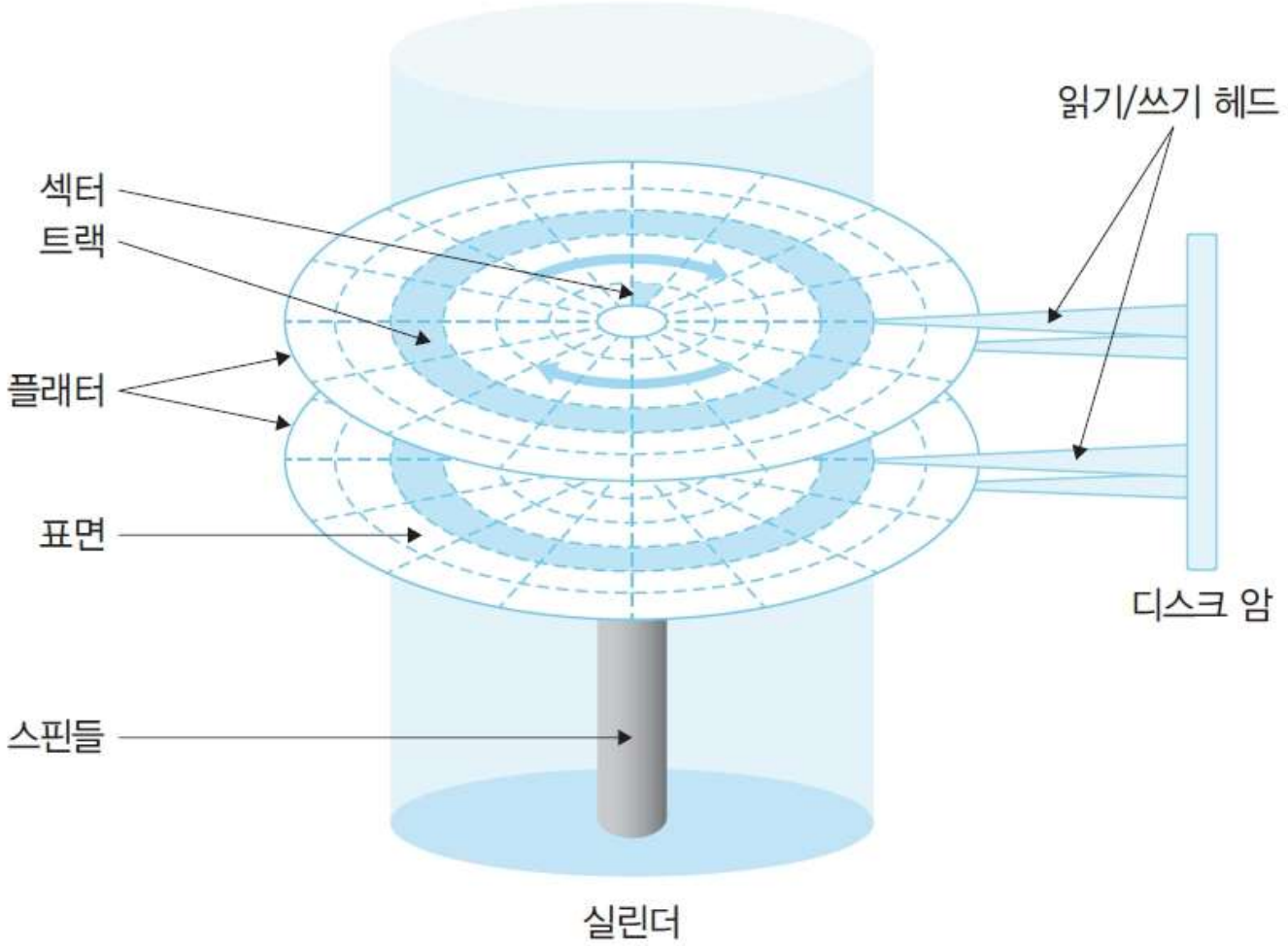


그림 10-9 하드디스크의 구조

2-1 저장장치의 종류

■ 플래터(platter)

- 표면에 자성체가 발려 있어 자기를 이용하여 0과 1의 데이터를 저장
- 플래터의 표면이 N극을 띠면 0, S극을 띠면 1로 인식
- 플래터 수는 보통 2장 이상으로 구성되며 항상 일정 속도로 회전

■ 섹터와 블록

- 섹터(sector)
  - 하드디스크의 가장 작은 저장 단위
  - 하나의 섹터에는 한 덩어리의 데이터가 저장
- 블록
  - 하드디스크와 컴퓨터 사이에 데이터를 전송하는 논리적인 저장 단위 중 가장 작은 단위
  - 여러 개의 섹터로 구성되며, 윈도우 운영체제에서는 블록 대신 클러스터(cluster)라고 표현
- 하드디스크 입장에서는 섹터가 가장 작은 저장 단위, 운영체제 입장에서는 하드디스크에 데이터를 보내거나 받을 때 블록이 가장 작은 저장 단위

2-1 저장장치의 종류

■ 트랙과 실린더

- 트랙(track): 플래터에서 회전축을 중심으로 데이터가 기록되는 동심원, 즉 동일한 동심원상에 있는 섹터의 집합
- 실린더(cylinder): 개념적으로 여러 개의 플래터에 있는 같은 트랙의 집합

■ 헤드와 플래터

- 하드디스크에서 데이터를 읽거나 쓸 때는 읽기/쓰기 헤드 사용
- 플래터가 회전을 시작하면 표면에 약한 바람이 일어나며 이 바람에 의해 헤드는 표면에서 약간 떠 있는 형태로 작동
- 플래터 표면에 생긴 상처는 데이터를 저장할 수 없는 배드 섹터가 됨

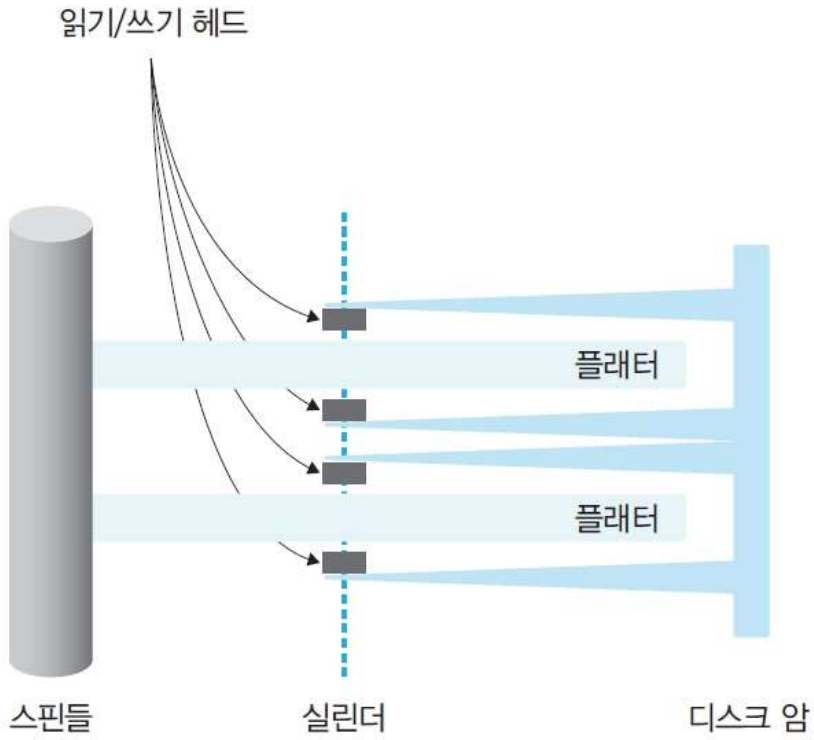


그림 10-10 하드디스크의 헤드와 플래터

2-1 저장장치의 종류

■ SSD(Solid State Disk)

- 하드디스크의 느린 속도를 대체하기 위해 개발된 보조저장장치
- 전원이 사라져도 데이터를 보관할 수 있는 플래시 메모리로 구성
- 메모리를 이용해 속도 빠르고 소음 없음. 크기 작고 외부 충격에 강하며 소모 전력량과 발열 수준이 하드디스크보다 낮음
- 반면, 같은 용량의 하드디스크에 비해 가격이 훨씬 비싼 것이 단점
- SSD의 가장 큰 장점은 빠른 데이터 입출력 속도
- SSD에서는 단편화가 데이터 접근 속도에 영향을 미치지 않아 조각 모음이 필요 없음

그림 10-11 NVMe 인터페이스 방식의 SSD

2-1 저장장치의 종류

■ CD

- 휴대할 수 있는 소형 원반에 데이터 저장
- 하드디스크와 마찬가지로 트랙과 섹터로 구성, 수평으로 움직이는 헤드가 트랙 사이를 움직이며 데이터를 읽음
- 표면에 미세한 홈이 파여 있어 헤드에서 발사된 레이저가 홈에 들어가 반사되지 않으면 0, 반사되어 돌아오면 1로 인식

그림 10-12 CD가 데이터를 읽는 방식

2-1 저장장치의 종류

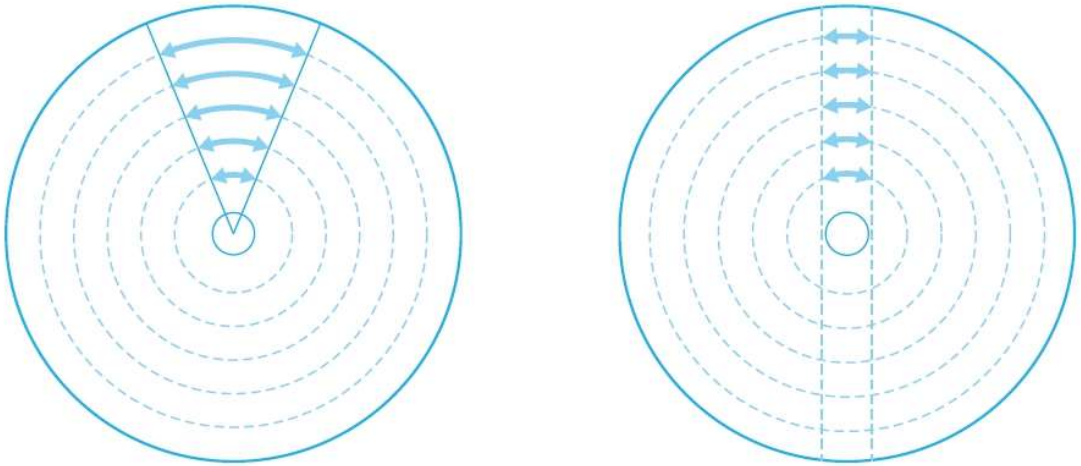
■ 하드디스크와 CD의 회전 비교

▪ 각속도 일정 방식의 회전

하드디스크의 플래터는 항상 일정한 속도로 회전하여 바깥쪽 트랙의 속도가 안쪽 트랙의 속도보다 훨씬 빨라 가장 바깥쪽 섹터가 가장 안쪽 섹터보다 큼. 일정 시간 동안 이동한 각도가 같다는 의미로 각속도 일정(constant angular velocity) 방식이라고 함

▪ 선속도 일정 방식의 회전

CD에서 사용하는 선속도 일정(constant linear velocity) 방식은 어느 트랙에서나 단위 시간당 디스크 이동 거리 같음. 이를 구현하려면 헤드가 안쪽 트랙에 있을 때는 디스크 회전 속도를 빠르게, 바깥쪽 트랙으로 이동하면 느리게 해야 함



(a) 하드디스크의 각속도 일정 방식　　(b) CD의 선속도 일정 방식

그림 10-13 디스크 장치의 회전 비교

2-1 저장장치의 종류

■ 하드디스크와 CD의 섹터

▪ 각속도 일정 방식의 섹터

- 각속도 일정 방식의 하드디스크는 트랙마다 속도 달라 섹터 크기도 다름
- 장점: 디스크가 일정 속도로 회전하므로 구동 장치가 단순하고 조용하게 작동
- 단점: 모든 트랙의 섹터 수가 같고 바깥쪽 섹터 크기가 안쪽 섹터보다 커서 바깥쪽 트랙으로 갈수록 낭비 공간 생김

▪ 선속도 일정 방식의 섹터

- 선속도 일정 방식의 CD는 모든 트랙의 움직이는 속도와 섹터 크기 같아서 안쪽보다 바깥쪽 트랙에 더 많은 섹터 존재(모든 트랙의 섹터 수 다름)
- 장점: CD는 한정된 공간에 많은 데이터를 담을 수 있고 하드디스크처럼 바깥쪽 트랙의 섹터 공간이 낭비되는 문제가 없음
- 단점: 모터 제어가 복잡하고 소음이 발생

2-2 디스크 장치의 데이터 전송 시간

■ 하드디스크에서 데이터를 전송하는 과정

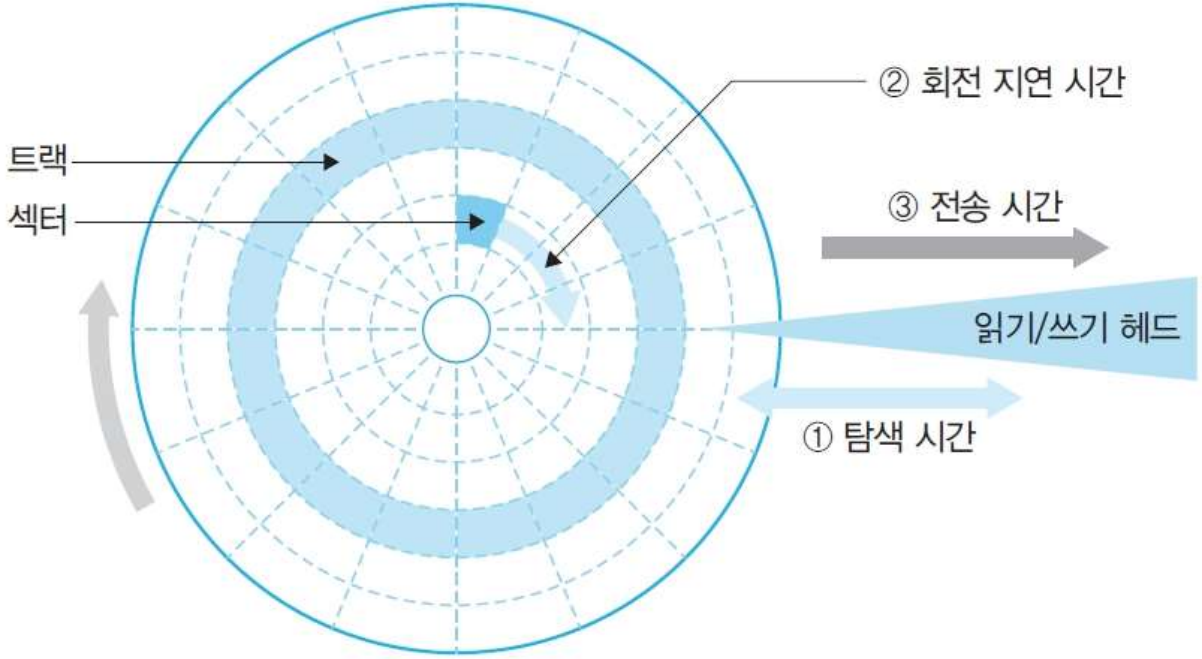


그림 10-15 하드디스크의 데이터 전송 과정

정의 10-1 디스크의 데이터 전송 시간

데이터 전송 시간 = 탐색 시간 + 회전 지연 시간 + 전송 시간

3-1 디스크 스케줄링 개요

■ 디스크 스케줄링(disk scheduling)

- 트랙 이동을 최소화하여 탐색 시간을 줄이는 것이 목적

■ 공통으로 이용할 트랙 접근 순서

순번	1	2	3	4	5	6	7	8	9
트랙 번호	15	8	17	11	3	23	19	14	20

그림 10-16 트랙 접근 순서

3-1 FCFS 디스크 스케줄링

■ FCFS 디스크 스케줄링(First Come, First Service disk scheduling)

- 요청 들어온 순서대로 서비스
- 헤드가 이동한 총거리: 7+9+6+8+20+4+5+6=65

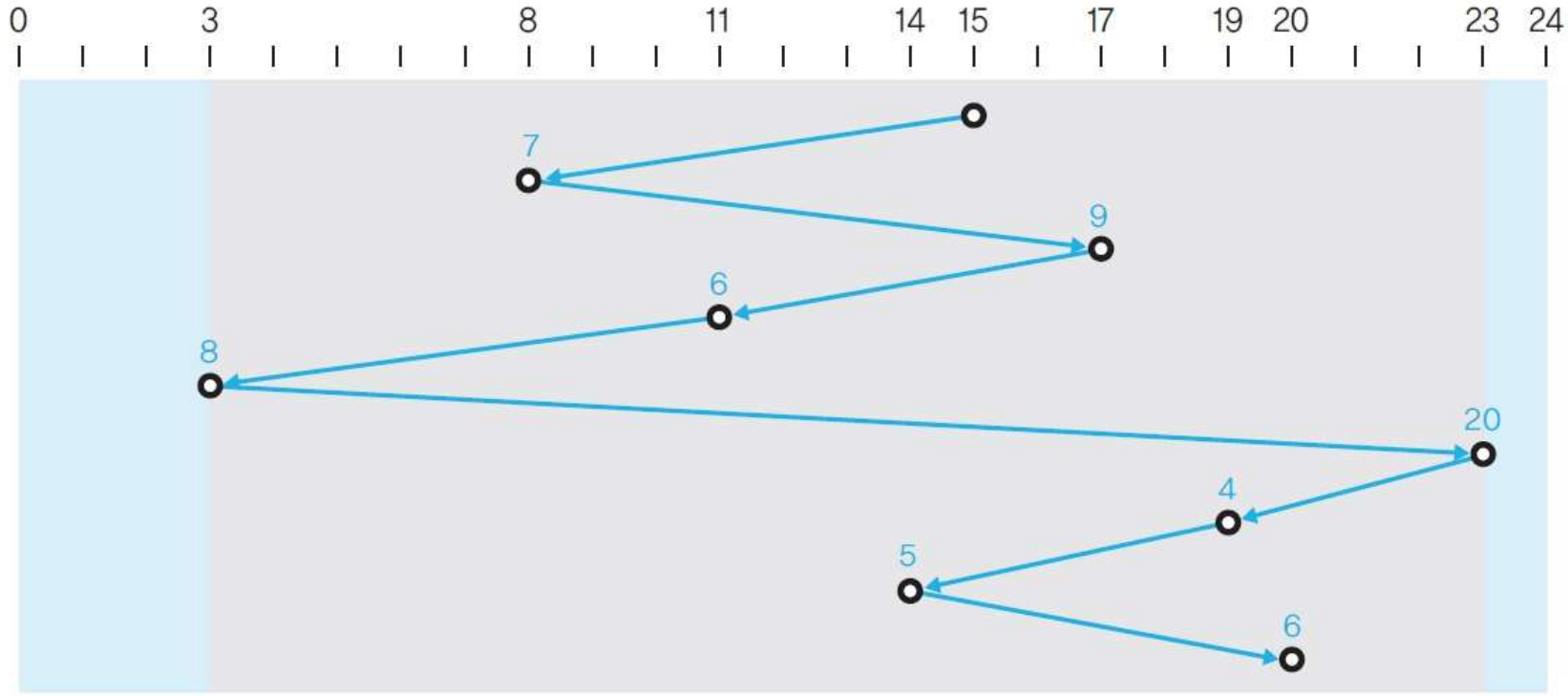


그림 10-17 FCFS 디스크 스케줄링의 동작

3-2 SSTF 디스크 스케줄링

■ SSTF 디스크 스케줄링(Shortest Seek Time First disk scheduling)

- 현재 헤드가 있는 위치에서 가장 가까운 트랙부터 서비스
- 다음에 서비스할 두 트랙의 거리가 같다면 먼저 요청받은 트랙을 서비스
- 헤드가 이동한 총거리: 1+3+3+1+3+12+3+5=31
- 효율성은 좋지만 아사 현상을 일으킬 수 있음

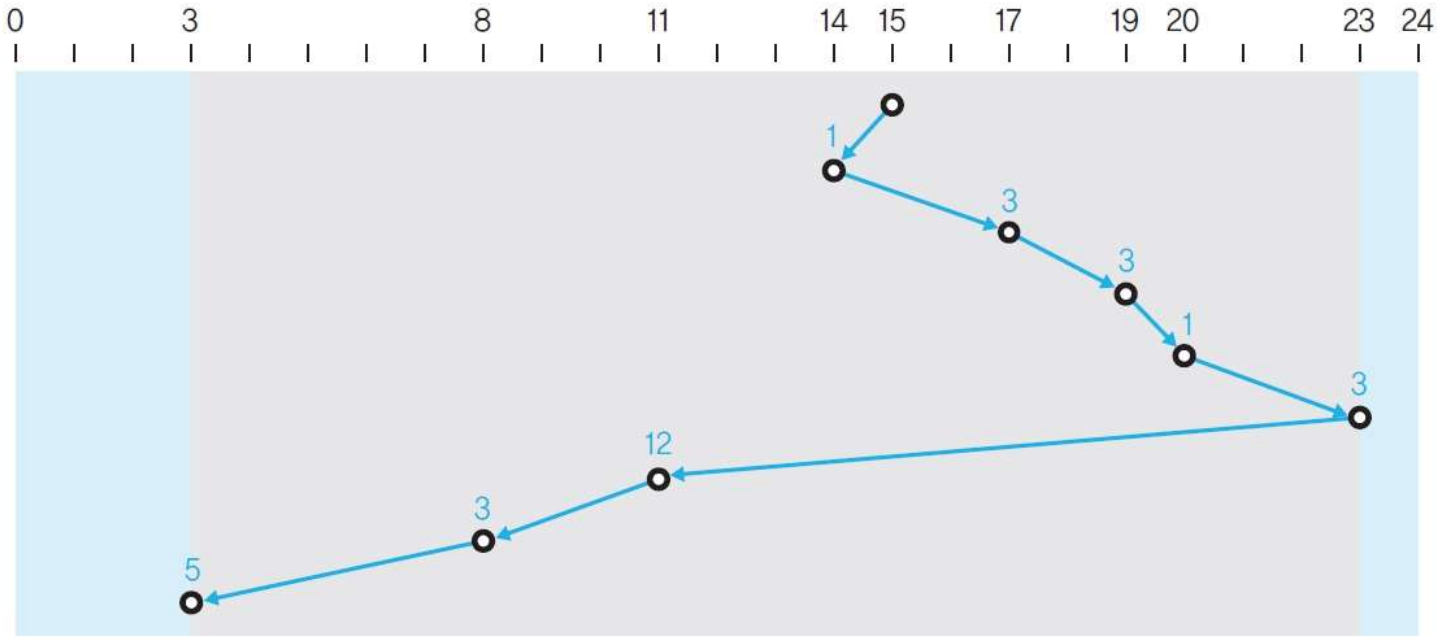


그림 10-18 SSTF 디스크 스케줄링의 동작

3-3 블록 SSTF 디스크 스케줄링

■ 블록 SSTF 디스크 스케줄링(block SSTF disk scheduling)

- 큐에 있는 트랙 요청을 일정한 블록 형태로 묶음
- 모든 트랙은 블록 안에서만 움직임
- 헤드가 이동한 총거리: 2+9+3+8+20+3+1+5=51
- 에이징을 사용하여 공평성을 보장하지만 성능은 FCFS 디스크 스케줄링만큼 좋지 않음

순번	1	2	3	4	5	6	7	8	9
트랙 번호	15	8	17	11	3	23	19	14	20
블록 SSTF	15	17	8	11	3	23	20	19	14

그림 10-19 블록 SSTF 디스크 스케줄링의 서비스 순서 변경

3-3 블록 SSTF 디스크 스케줄링

■ 블록 SSTF 디스크 스케줄링(block SSTF disk scheduling)

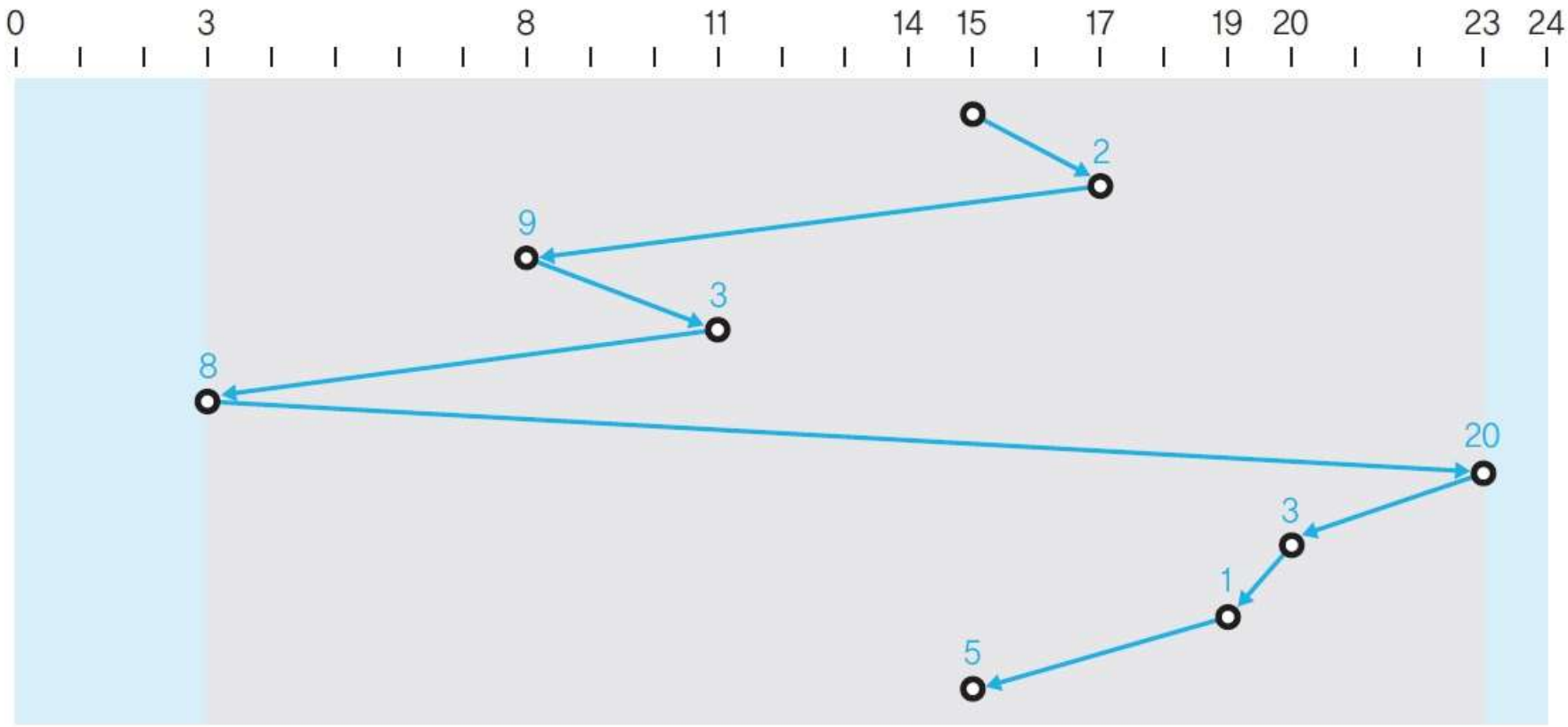


그림 10-20 블록 SSTF 디스크 스케줄링의 동작

3-4 SCAN 디스크 스케줄링

■ SCAN 디스크 스케줄링(SCAN disk scheduling)

- 헤드가 움직이면 맨 마지막 트랙에 도착할 때까지 뒤돌아가지 않고 계속 전진하면서 요청받은 트랙을 서비스
- 헤드가 이동한 총거리: 1+3+3+5+3+17+2+1+3=38
- 동일 트랙이나 실린더 요청이 연속 발생하면 헤드가 더 이상 전진하지 못하고 제자리에 머물러 바깥쪽 트랙이 아사 현상을 겪는 문제 발생

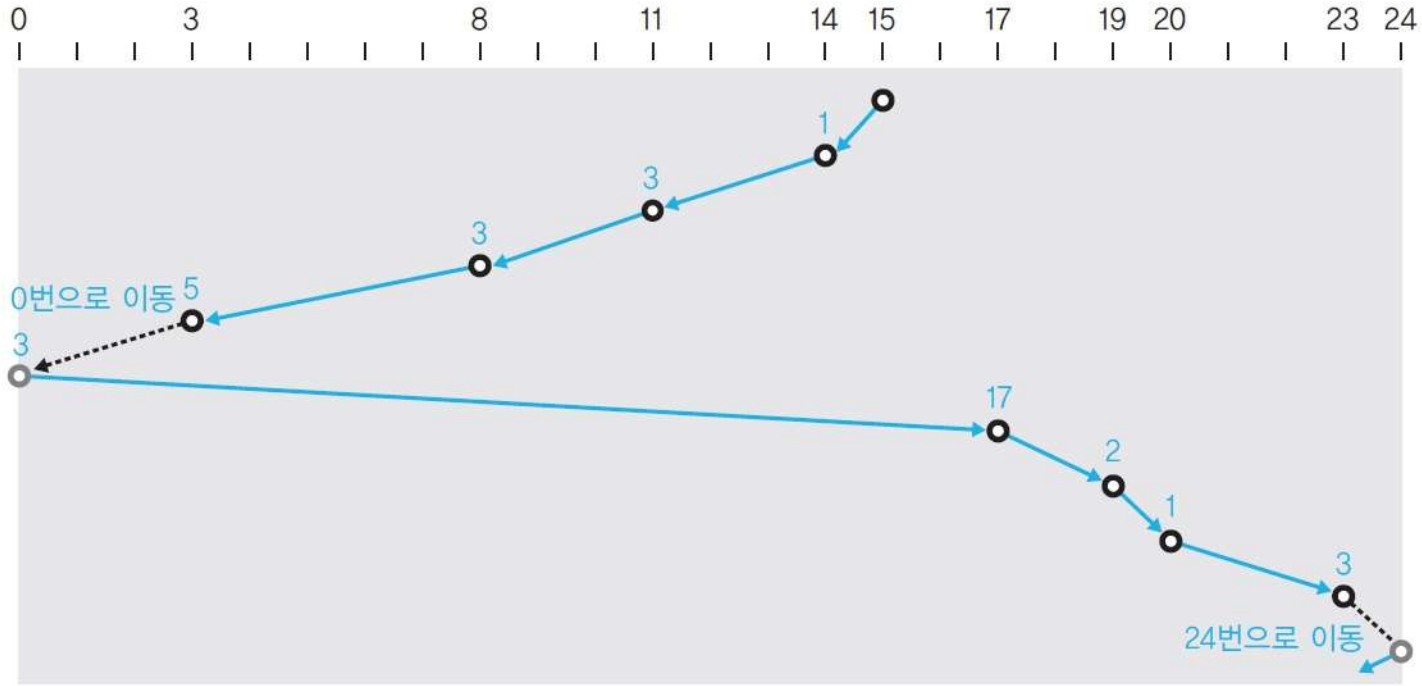


그림 10-21 SCAN 디스크 스케줄링의 동작

3-5 C-SCAN 디스크 스케줄링

■ C-SCAN 디스크 스케줄링(Circular SCAN disk scheduling)

- SCAN 디스크 스케줄링을 변형한 것으로 헤드가 한쪽 방향으로 움직일 때는 요청받은 트랙을 서비스하고 반대 방향으로 돌아올 때는 서비스하지 않고 이동만 함
- 헤드가 이동한 총거리: 1+3+3+5+3+24+1+3+1+2=46
- 작업 없이 헤드를 이동하는 것은 매우 비효율적

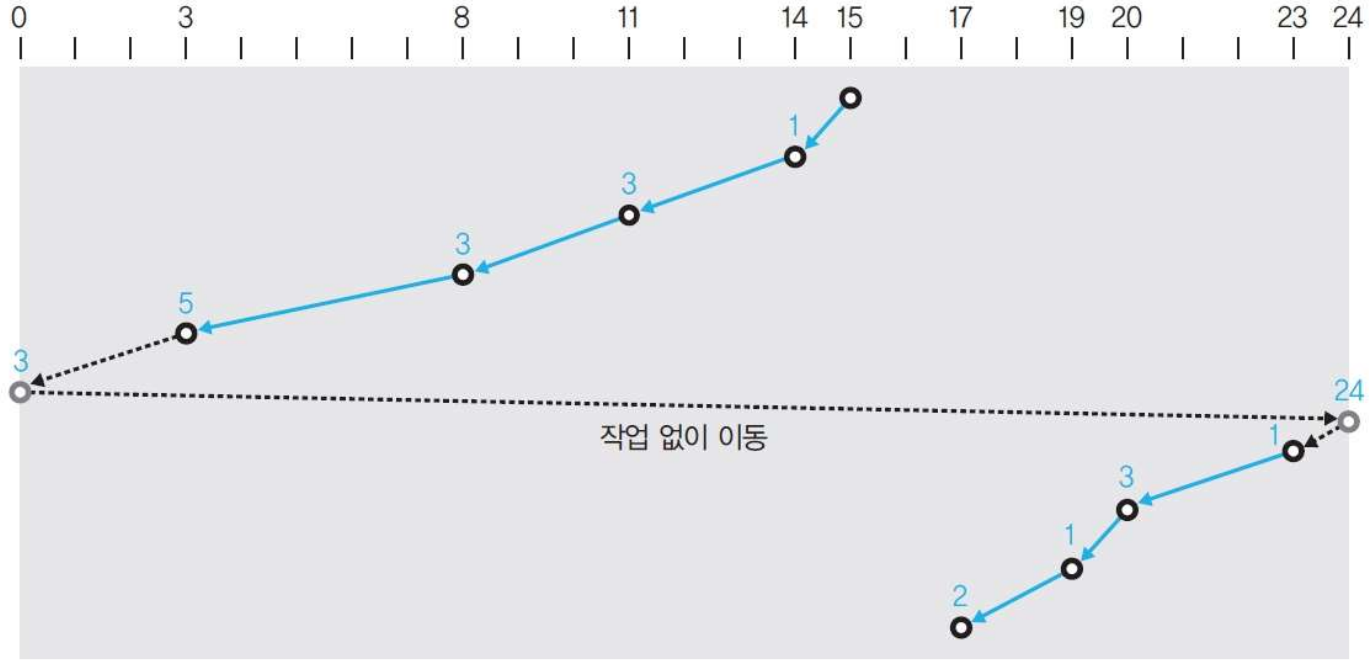


그림 10-22 C-SCAN 디스크 스케줄링의 동작

3-6 LOOK 디스크 스케줄링

■ LOOK 디스크 스케줄링(LOOK disk scheduling)

- 더 이상 서비스할 트랙 없으면 헤드가 끝까지 가지 않고 중간에서 방향 바꿈
- 헤드가 이동한 총거리: 1+3+3+5+17+2+1+3=35

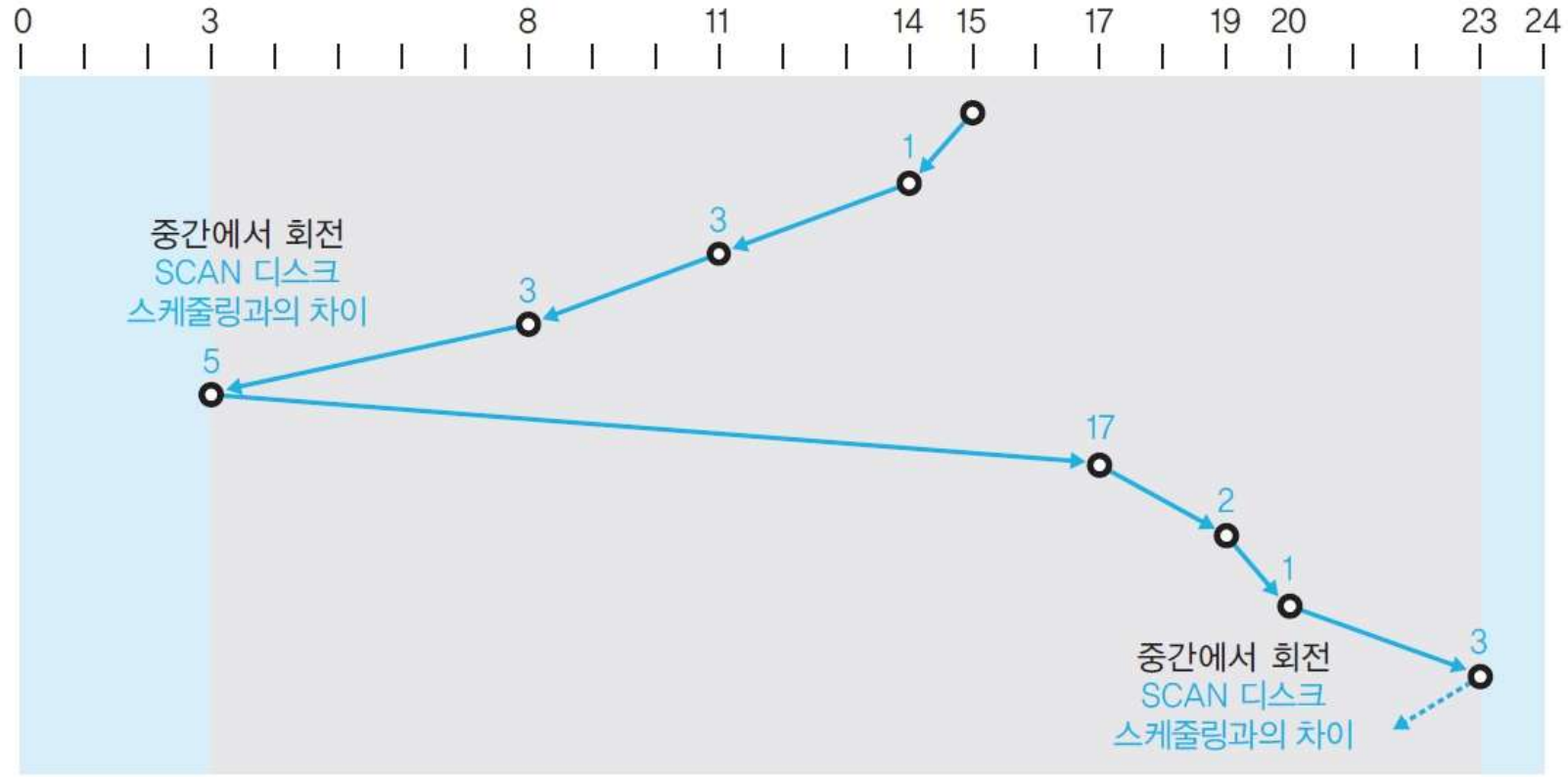


그림 10-23 LOOK 디스크 스케줄링의 동작

3-7 C-LOOK 디스크 스케줄링

■ C-LOOK 디스크 스케줄링(Circular LOOK disk scheduling)

- C-SCAN 디스크 스케줄링의 LOOK 버전
- 한쪽 방향으로만 서비스하는 C-SCAN 디스크 스케줄링과 유사, 차이점은 더 이상 서비스할 트랙이 없으면 헤드가 중간에서 방향을 바꿀 수 있음
- 헤드가 이동한 총거리: 1+3+3+5+20+3+1+2=38

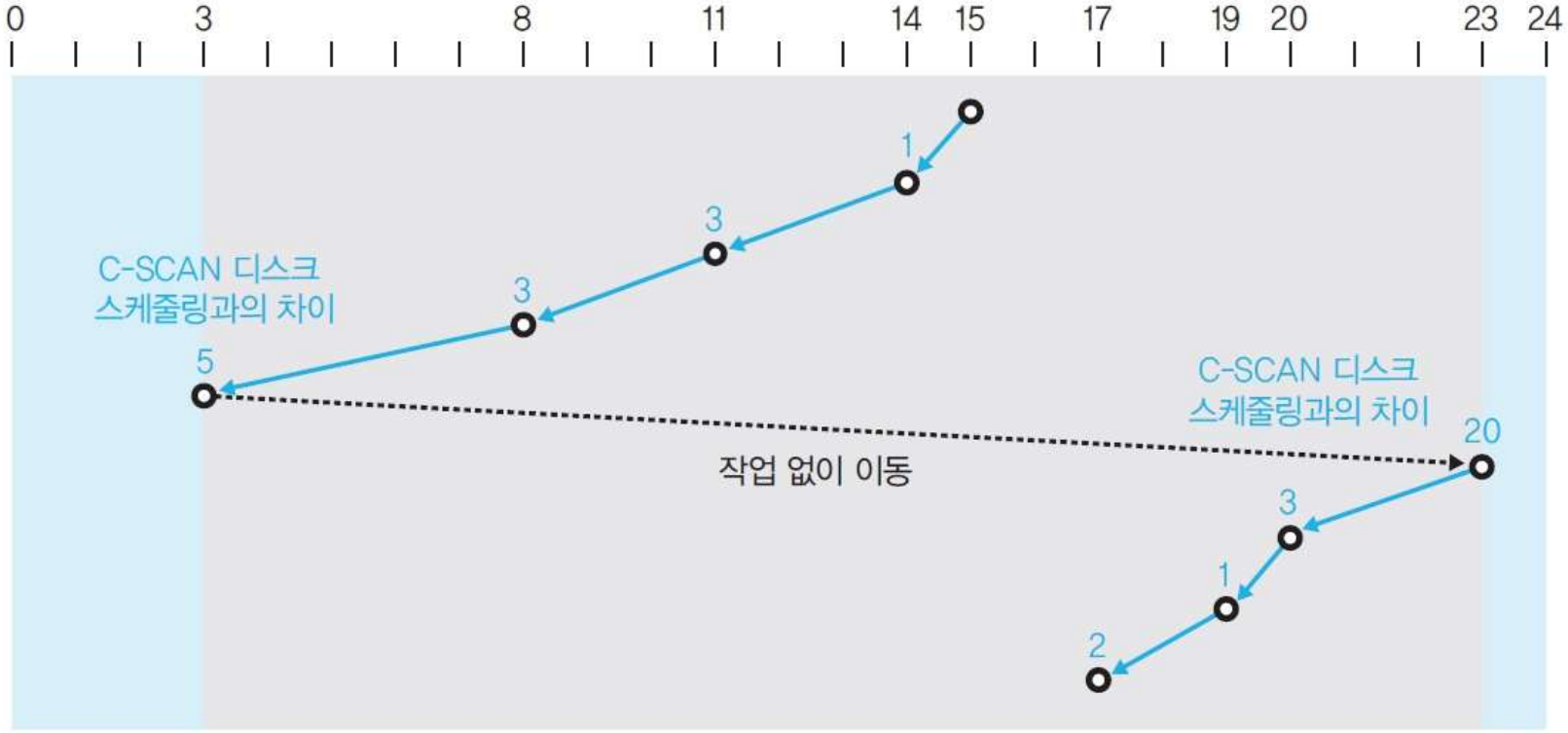


그림 10-24 C-LOOK 디스크 스케줄링의 동작

3-8 SLTF 디스크 스케줄링

■ SLTF 디스크 스케줄링(Shortest Latency Time First disk scheduling)

- 큐에 들어온 요청을 디스크 회전 방향에 맞추어 재정렬한 후 서비스

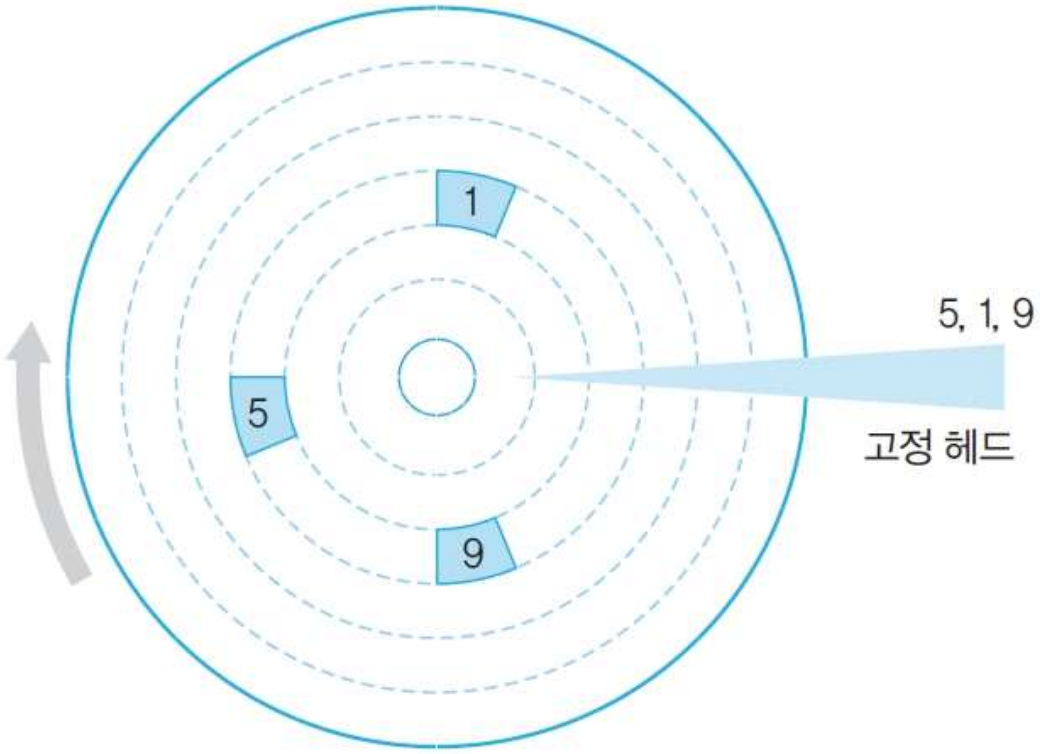


그림 10-25 SLTF 디스크 스케줄링의 동작

4-1 RAID의 개요

■ RAID(Redundant Array of Independent Disks)

- 자동으로 백업하고 장애가 발생하면 복구하는 시스템, '레이드'로 읽음
- 미러링(mirroring): 하나의 원본 디스크와 같은 크기의 백업 디스크에 같은 내용을 동시 저장하고, 하나의 디스크가 고장나면 다른 디스크를 사용하여 데이터를 복구하는 방식
- 스트라이핑(striping): 여러 디스크에 데이터를 동시에 저장해 데이터 입출력 속도를 높이는 방식

4-2 RAID 0(스트라이핑)

■ RAID 0

- 병렬로 연결된 여러 개의 디스크에 데이터를 동시에 입출력할 수 있도록 구성
- 데이터를 여러 갈래로 찢어서 저장하므로 스트라이핑이라고 부름
- 4개 디스크로 구성된 RAID 0은 1개 디스크로 구성된 일반 시스템보다 이론상 입출력 속도가 4배 빠름
- 장애 발생 시 복구 기능이 없어 장애가 발생하면 데이터를 잃지만 입출력이 빨라 기업용 제품, 개인용 컴퓨터와 노트북의 고급 기종에 많이 사용

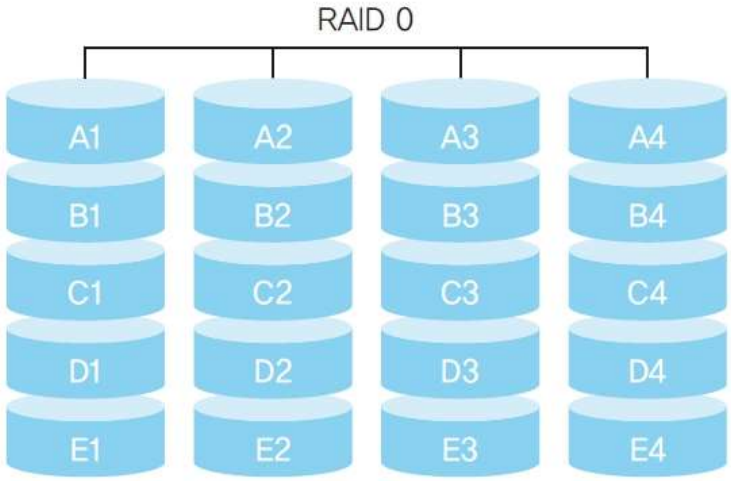


그림 10-26 RAID 0의 구조

4-3 RAID 1(미러링)

■ RAID 1

- 하나의 데이터를 2개 디스크에 나누어 저장하여 장애 시 백업 디스크로 활용
- 데이터가 똑같이 여러 디스크에 복사되어 미러링이라고 부름
- 같은 크기의 디스크가 최소 2개 이상 필요하며 짝수 개 디스크로 구성
- 단점: 저장하는 데이터와 같은 크기의 디스크가 하나 더 필요해 비용이 증가
- 단점: 같은 내용을 두 번 저장하기 때문에 속도가 느려질 수 있음

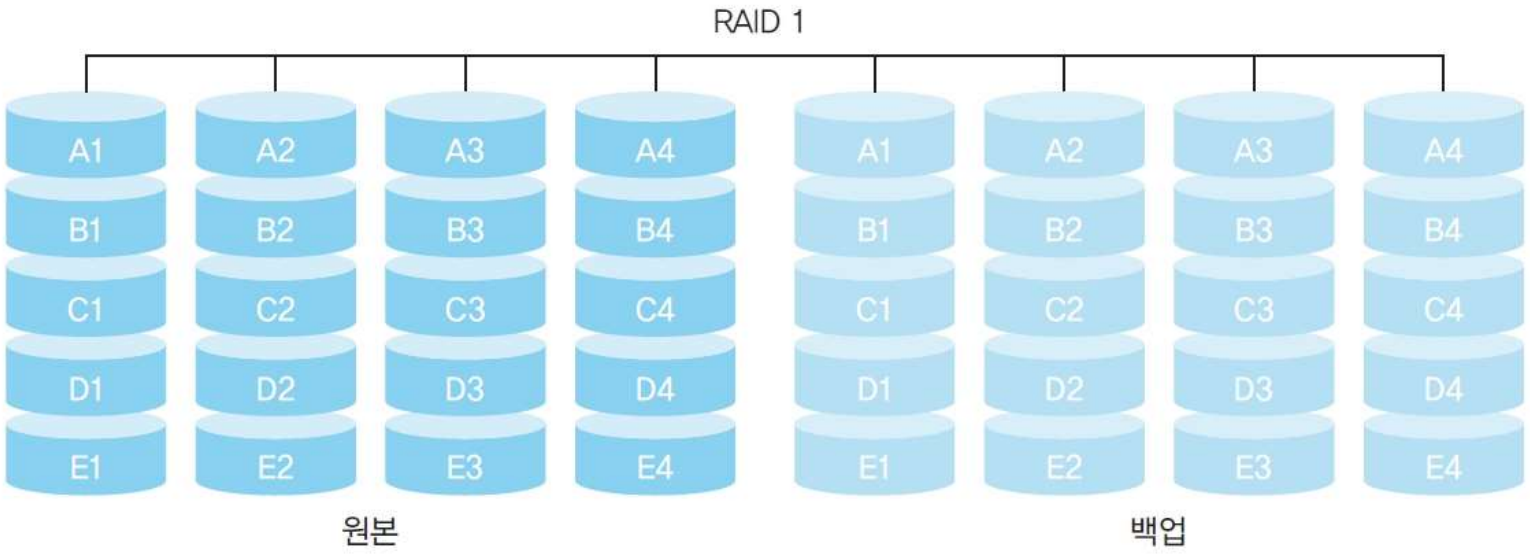


그림 10-27 RAID 1의 구조

4-4 RAID 2

■ RAID 2

- 오류 검출 기능이 없는 디스크에 대해 오류 교정 코드(ECC)를 따로 관리, 오류 발생하면 이 코드로 디스크 복구
- 비트별로 만들어진 오류 교정 코드는 별도 디스크에 저장, 장애 발생 시 이 코드로 데이터 복구
- n개 디스크의 오류 교정 코드를 저장하려면 n-1개의 추가 디스크 필요. RAID 1보다는 작은 저장 공간 필요하지만 오류 교정 코드를 계산하는 데 많은 시간이 소비되어 잘 사용하지 않음

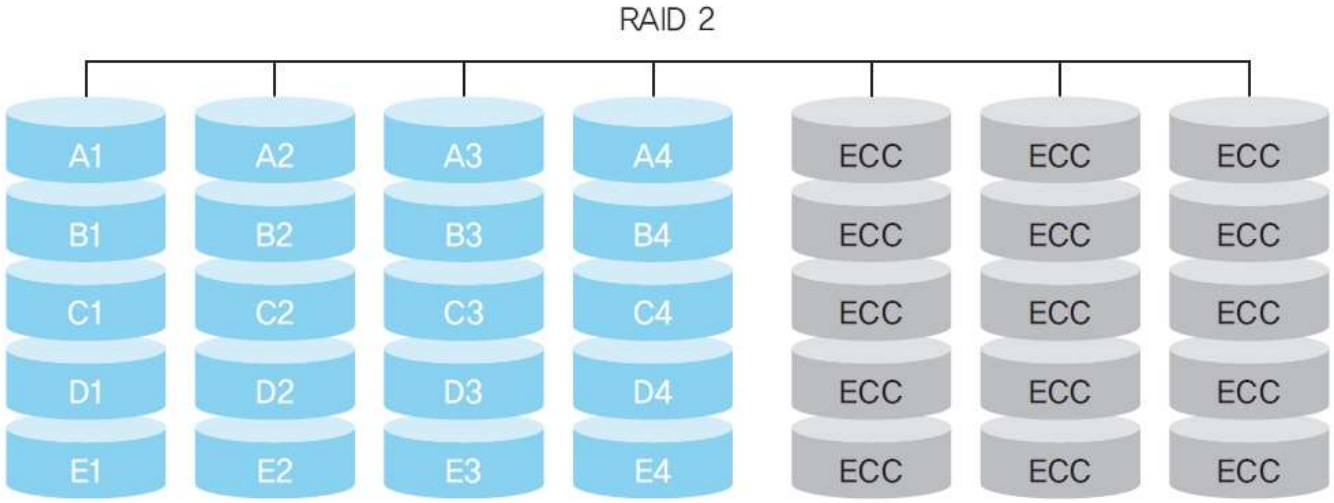


그림 10-28 RAID 2의 구조

4-5 RAID 3

■ RAID 3

- 섹터 단위로 데이터를 나누어 저장(RAID 2는 비트 단위로 데이터를 나누어 저장)
- N-way 패리티 비트 방식: 오류가 없는 섹터로 오류가 있는 섹터의 데이터를 복원
- N-way 패리티 비트를 구성 후 데이터 디스크가 아닌 별도 디스크에 보관해 장애 발생 시 오류 복구
- 단점: 추가되는 디스크 양 적지만 N-way 패리티 비트를 구성하는 데 필요한 계산량 많음

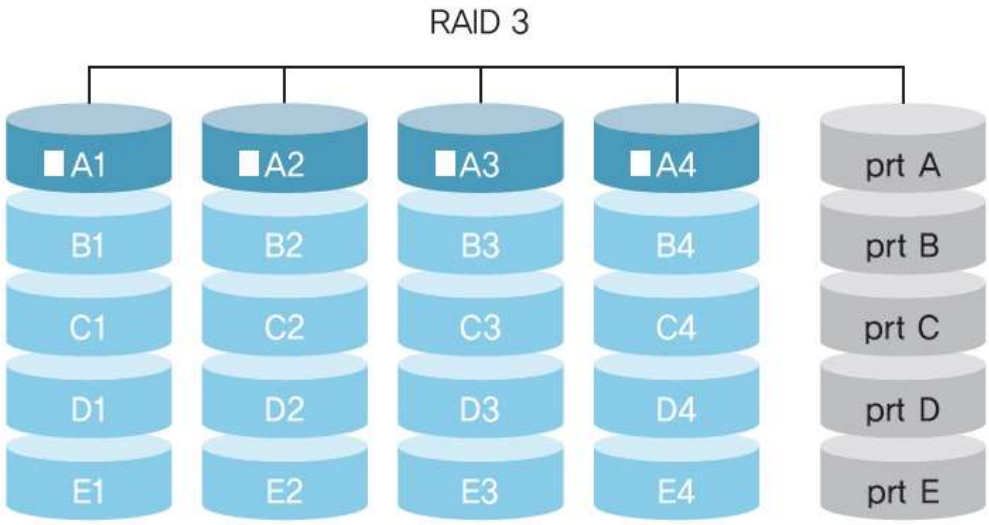


그림 10-30 RAID 3의 구조

4-6 RAID 4

■ RAID 4

- 데이터를 하나의 디스크에 블록 단위로 저장, 패리티 비트를 블록과 연결하여 구성
- 장점: 데이터가 저장되는 디스크와 패리티 비트가 저장되는 디스크만 동작
- 패리티 비트를 추가하기 위한 계산량이 많지만 추가되는 디스크의 양은 적음.
- RAID 0은 4개 디스크에 4개 디스크 추가, RAID 4는 4개 디스크에 1개 디스크 추가

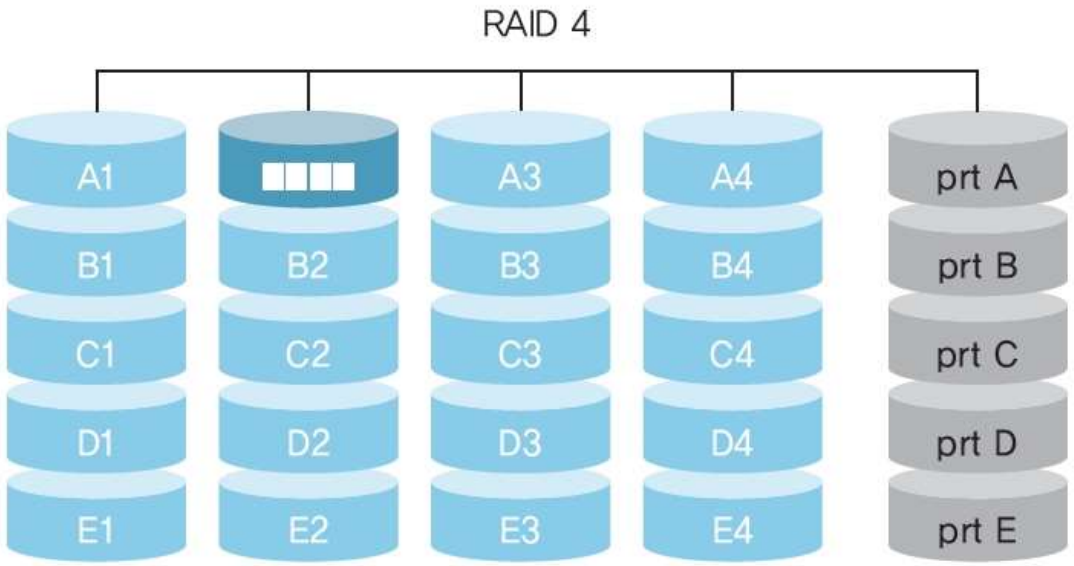


그림 10-31 RAID 4의 구조

4-7 RAID 5

■ RAID 5

- 패리티 비트를 여러 디스크에 분산, 보관해 패리티 비트 디스크의 병목 현상을 완화
- 패리티 비트를 해당 데이터가 없는 디스크에 보관, 한 디스크에 장애가 발생하면 다른 디스크에 있는 패리티 비트로 데이터 복구 가능

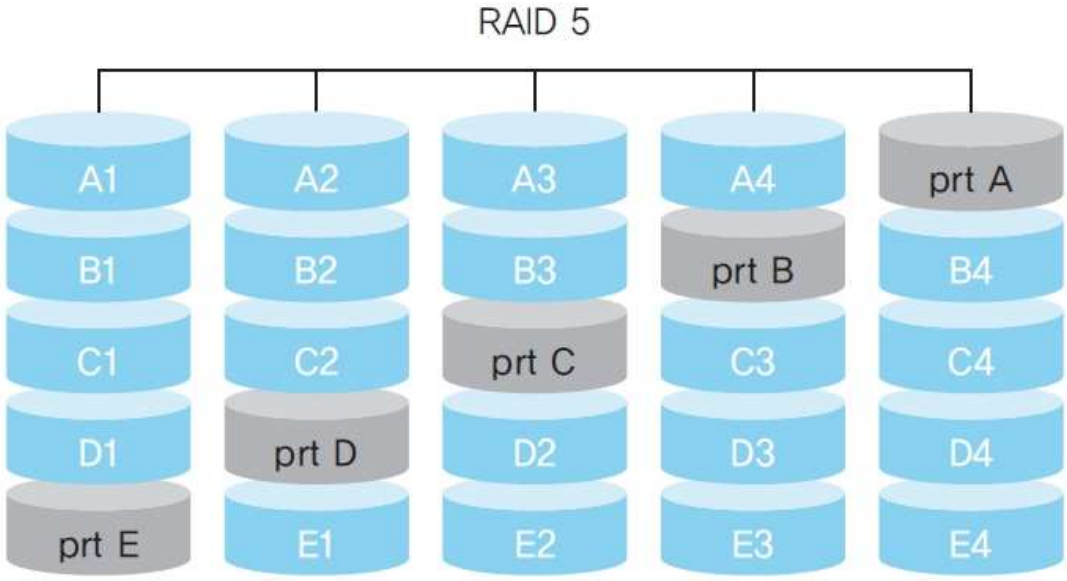


그림 10-32 RAID 5의 구조

4-8 RAID 6

■ RAID 6

- 패리티 비트를 2개로 구성하여 디스크 2개의 장애 복구 가능
- 단점: 패리티 비트를 2개씩 운영하기 때문에 RAID 5보다 계산량이 많고 4개 디스크에 2개의 추가 디스크가 필요

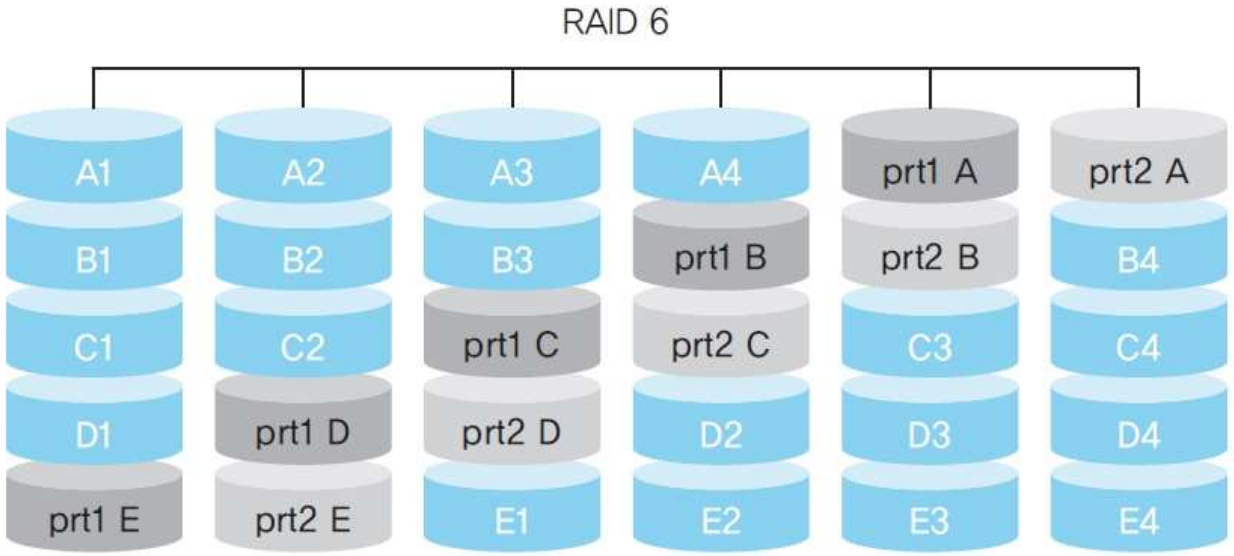


그림 10-33 RAID 6의 구조

4-9 RAID 10

■ RAID 10

- RAID 1+0을 의미, 미러링 기능이 있는 RAID 1과 빠른 데이터 전송 가능한 RAID 0 결합
- 디스크를 RAID 0으로 먼저 묶으면 RAID 0+1, RAID 1로 먼저 묶으면 RAID 10
- 장애 발생시, RAID 0+1은 모든 디스크 중단, RAID 10은 일부 디스크만 중단해 복구 가능
- RAID 10이 더 많이 사용되지만 최소 4개의 디스크 필요

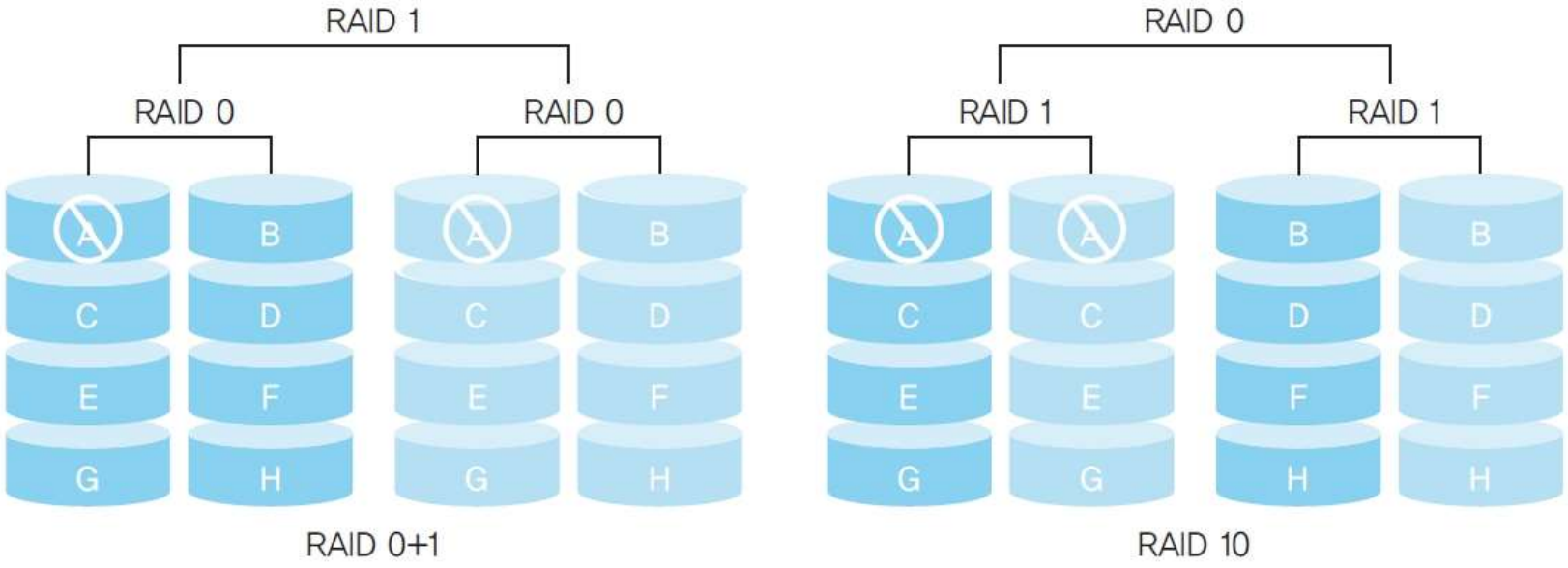


그림 10-34 RAID 0+1과 RAID 10의 구조

4-10 RAID 50과 RAID 60

■ RAID 50과 RAID 60

- RAID 50: RAID 5로 묶은 두 쌍을 다시 RAID 0으로 묶어 사용
- RAID 60: RAID 6으로 묶은 두 쌍을 다시 RAID 0으로 묶어 사용
- RAID 50과 RAID 60은 RAID 10에 비해 추가되는 디스크 수 적지만 입출력 시 계산량 증가

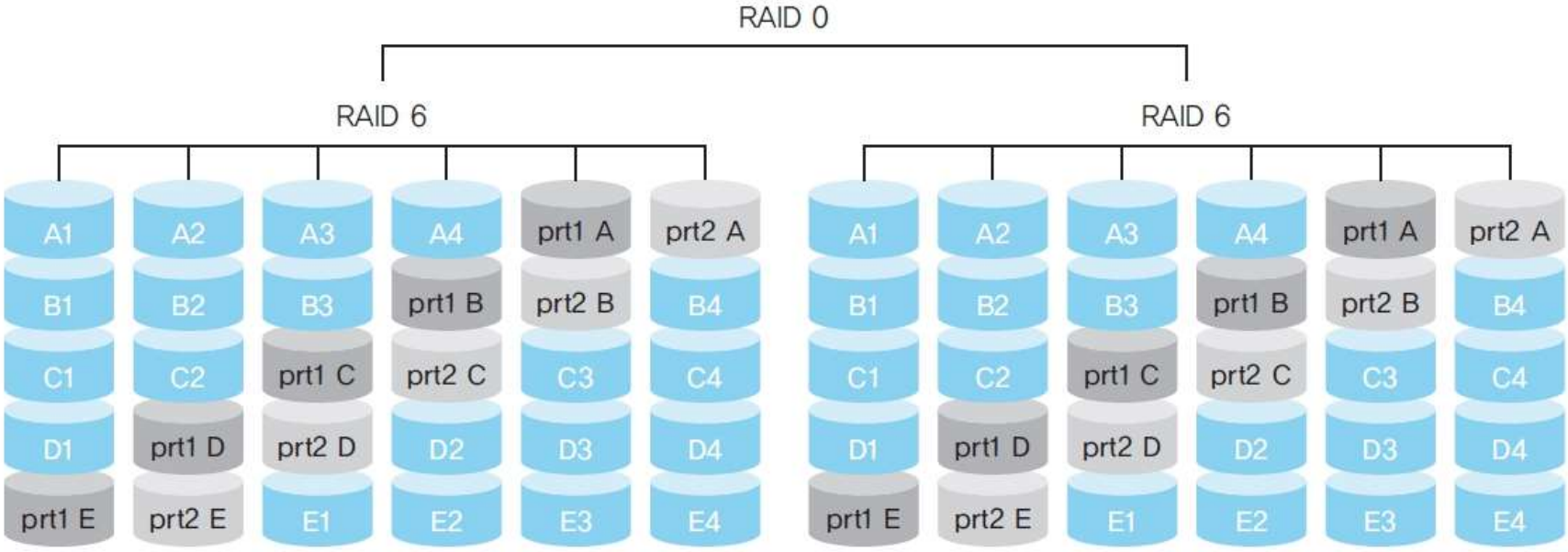


그림 10-35 RAID 60의 구조

5-1 포트의 규격

■ 메인보드의 포트

- CPU 포트: CPU 꽂는 곳
- 램 포트: 램을 수직으로 꽂는 곳
- 그래픽 포트: 외부 그래픽카드를 연결하는 포트
- SATA(Serial ATA): 하드디스크 같은 저장장치를 연결하는 직렬 ATA 포트
- PCI: 그 외의 주변장치는 메인보드에 주변장치를 연결하는 포트

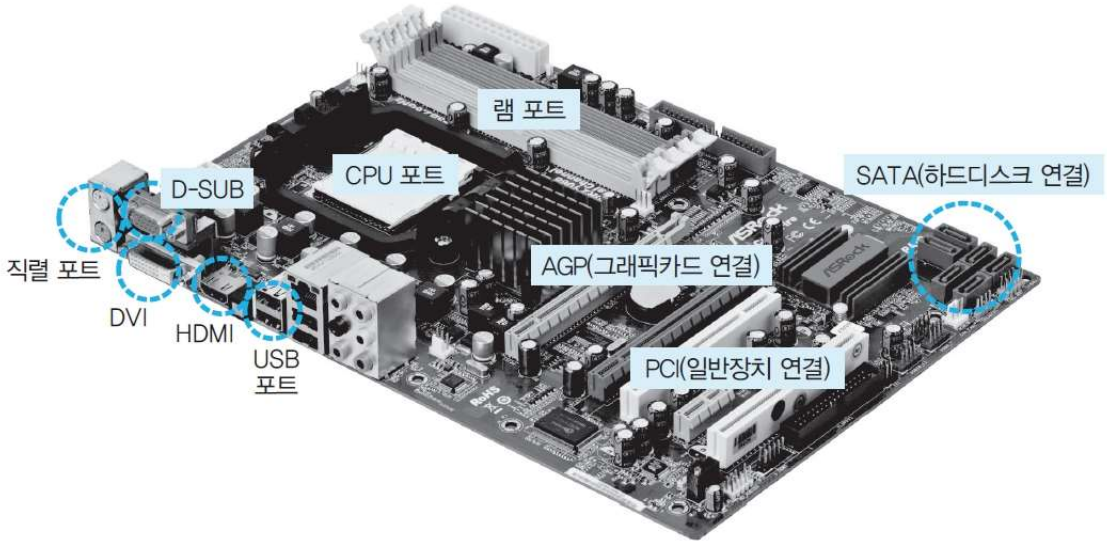


그림 10-36 메인보드의 포트

5-1 포트의 규격

■ 직렬 포트와 병렬 포트

- 버스의 통신 방식은 크게 직렬 방식과 병렬 방식으로 구분
- 직렬 방식에서는 데이터가 한 줄로 이동, 병렬 방식에서는 데이터가 여러 줄로 동시에 이동

■ USB 포트

- 키보드, 마우스, 프린터, 카메라, 저장장치 등 다양한 주변장치를 연결하기 위해 만든 표준 연결 포트로 전원 충전 용도로도 사용

5-1 포트의 규격

■ 포트 연결 단자

- USB: USB 메모리나 카메라 등 다양한 주변장치를 연결할 수 있는 범용 포트
- SATA: 컴퓨터 내부에 있는 각종 저장장치를 연결할 때 사용
- D-SUB(D-SUBminiature): 가장 오래된 모니터 연결 단자로 대개 파란색
- DVI(Digital Visual Interface): 컴퓨터 디스플레이와 디지털 프로젝터 같은 디지털 디스플레이 장치의 화질에 최적화된 표준 영상 인터페이스
- HDMI(High Definition Multimedia Interface): 비압축 방식의 디지털 비디오/오디오 인터페이스 규격

그림 10-37 다양한 포트 연결 단자

5-2 CD의 규격

■ 오디오용 CD

- 약 74분 분량의 음원 파일이 저장되고 디지털 데이터는 약 650MB 저장

■ 비디오 CD

- MPEG 레벨 1로 압축된 최대 83분 분량의 영화 저장

■ DVD

- 기본적으로 4.7GB 저장, 듀얼레이어 DVD는 총 8.5GB 저장

■ 블루레이디스크

- 싱글레이어 블루레이디스크는 25GB, 더블레이어 블루레이디스크는 50GB 저장
- 높은 해상도의 동영상이나 압축하지 않은 고음질의 음악, 게임 저장에 사용

5-3 그래픽카드의 발전

■ 그래픽카드의 발전

- CPU는 복잡한 그래픽 계산에 적합하게 설계되지 않음
- 현대의 컴퓨터 시스템에는 그래픽 계산만 전담하는 GPU가 그래픽카드에 추가
- 일반 작업은 CPU, 그래픽 작업은 GPU가 담당하는 형태로 바뀜
- 고성능 3D 게임을 하기 위해서는 좋은 CPU보다 좋은 GPU가 달린 그래픽카드를 선택해야 함
- GPU의 계산 능력이 커짐에 따라 알파고나 가상화폐 채굴뿐 아니라 다양한 곳에 GPU가 활용되고 있음

그림 10-38 가상화폐 채굴기