기말고사 정리

Chap 9. 메인 메모리

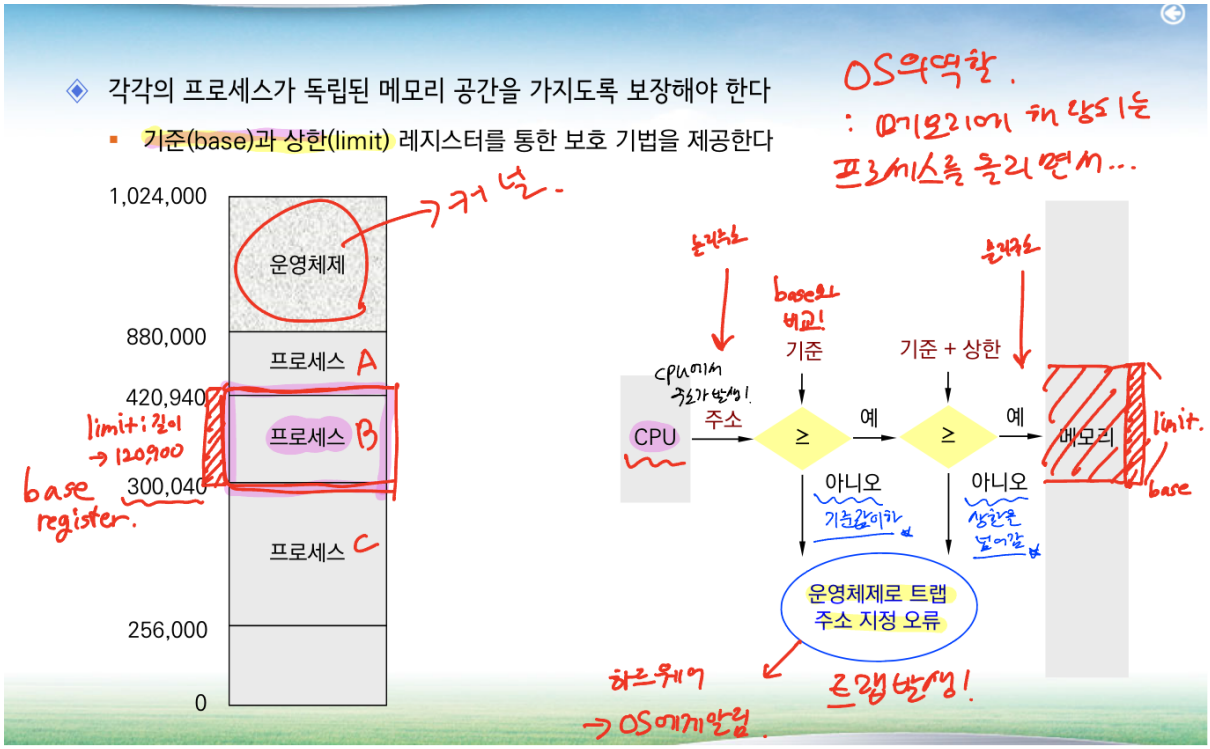
하드웨어가 메모리를 관리하는 방법?

기본 하드웨어는 **각각의 프로세스가 독립된 메모리 공간을 가지도록 보장해야 한다.**

기준 레지스터와 상한 레지스터를 사용해 보호 기법을 제공하며,

잘못된 주소에 접근하게 되면, 트랩(trap)을 발생시켜 운영체제로 알리는 역할도 한다.

OS는 기준 레지스터와 상한 레지스터에 값을 설정하는 일을 한다.



주소의 할당

바인딩이란 ? : 주소 값을 어떻게 세팅하는가..?

명령어와 데이터의 바인딩은 이루어지는 시점에 따라 다음과 같이 구분 됨

- 컴파일 시간 바인딩 : 메모리의 주소가 컴파일 시간에 결정되는 것

  → OS의 커널에 일부 코드 존재
- 적재 시간 바인딩 : 메모리에 올릴 때 주소를 결정
- 실행 시간 바인딩 : 실행 중간에 메모리의 주소가 변경될 수 있음

컴파일시/ 적재 시 주소 바인딩 기법의 경우 → 논리, 물리 주소가 같음

실행시간 주소 바인딩 기법의 경우 → 논리, 물리 주소가 다름 / **논리 주소 = 가상 주소**

메모리 관리기 `MMU` 를 통해 가상 주소에서 물리 주소로의 변환을 수행하며, **재배치 레지스터** 를 사용

동적 적재

프로세스가 실행되기 위해 그 프로세스 전체가 미리 메모리에 올라와 있어야함.

이러한 상황에서, 메모리 공간의 더 효율적인 이용을 위해 동적 적재 `dynamic loading` 를 해야함

각 루틴은 실제 호출되기 전까지는 메모리에 올라오지 않고 재배치 가능한 상태로 디스크에서 대기하다

필요한 경우에만 적재되며 아주 간혹 발생하면서도 많은 양의 코드를 필요로 하는 경우 유용하게 사용됨

동적 연결 및 공유 라이브러리

동적 연결 라이브러리는 프로그램이 실행될 때 연결되는 라이브러리로 `.dll` 파일 (dynamic linking library)

동적 적재를 수반하며, 많은 실행 파일들에서 공통으로 사용할 수 있고, 루틴을 바꿀 때 유용하다는 장점이 있음

동적 메모리 할당 문제 해결책

1. 최초 적합

   → 첫 번째 사용 가능한 가용 공간에 할당

2. 최적 적합

   → 사용 가능한 공간 중에서 가장 작은 가용 공간을 할당

3. 최악 적합

   → 가장 큰 가용 공간을 할당

단편화

외부 단편화

프로세스들이 메모리에 적재되고 제거될 때 할당되지 않고 작은 조각들로 메모리가 나뉘어져 있는 상태

외부 단편화 해결책

1. **메모리의 내용을 한쪽으로 밀어서 큰 공간을 만드는 방법 : 밀집**

   → 시간 낭비가 심함

2. **요구되는 메모리 크기보다 더 크게 할당**

   → 이때, 할당 했지만 안쓰는 공간을 내부 단편이라고 함

3. 한 프로세스의 논리 주소 공간을 여러 개의 비연속적인 공간에 나누어 할당하는 방법

   → **세그멘테이션과 페이징을 사용**

페이징

메모리를 일정한 크기의 페이지로 나누어 관리하는 메모리 관리 기법

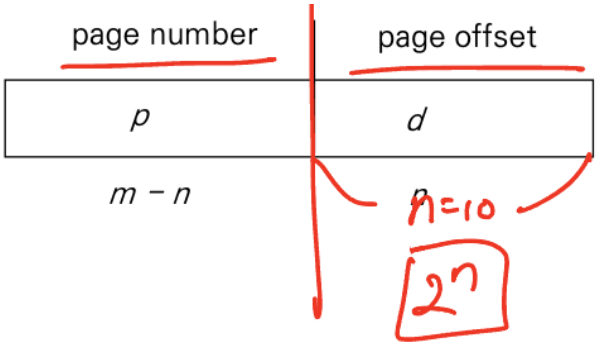
논리 주소 공간이 한 연속적인 공간에 모여 있어야 한다는 제약을 없앰

- 논리 메모리는 **페이지**라 불리는 같은 크기의 블록으로 나뉨
- 물리 메모리는 **프레임**이라 불리는 같은 크기의 블록으로 나뉨

페이지 사이즈는 2^n의 크기로 사용

페이지 테이블은 다음과 같이 구성

- 페이지 번호
  - 페이지 테이블의 index로 사용
- 페이지 변위
  - 해당 페이지에서의 데이터의 시작 주소


page number
page offset
p
d
m − n
n=10
27