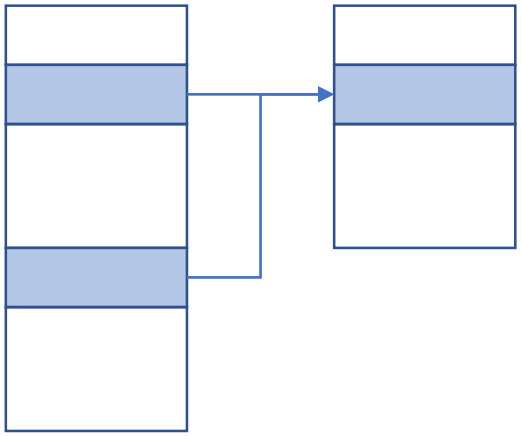
Homework #3

- 운영체제는 페이지 테이블을 이용해 물리 메모리를 가상화 하게 된다.
- 운영체제가 페이지 테이블을 통해 할 수 있는 두 가지 기능을 간접적으로 체험해 보자.

Homework #3

- Double Mapping
  - 페이지 테이블을 이용하면 서로 다른 가상 주소가 하나의 물리 메모리를 가리키도록 할 수 있다.



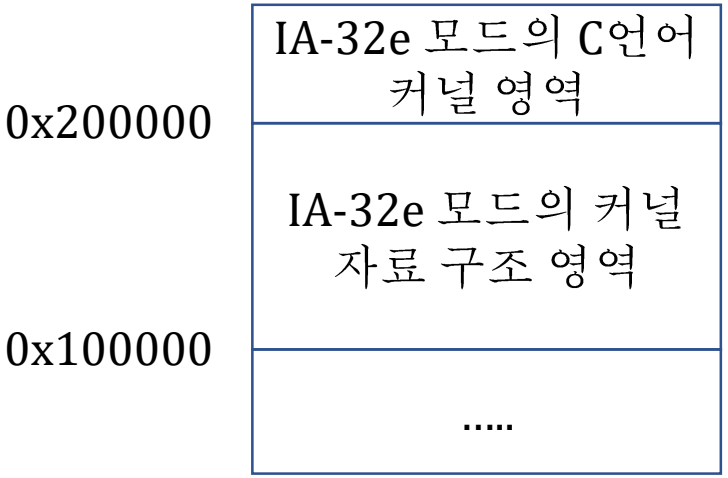
  - 현재 비디오 메모리는 0xB8000에 위치한다.
    - Kernel64의 main함수 참조
  - 우리는 페이지 디렉토리 엔트리 (2MB 단위)를 수정하여
    - 0xAB8000에 비디오 메모리를 맵핑하여, 이 주소를 통해서 문자열을 출력해 보자.

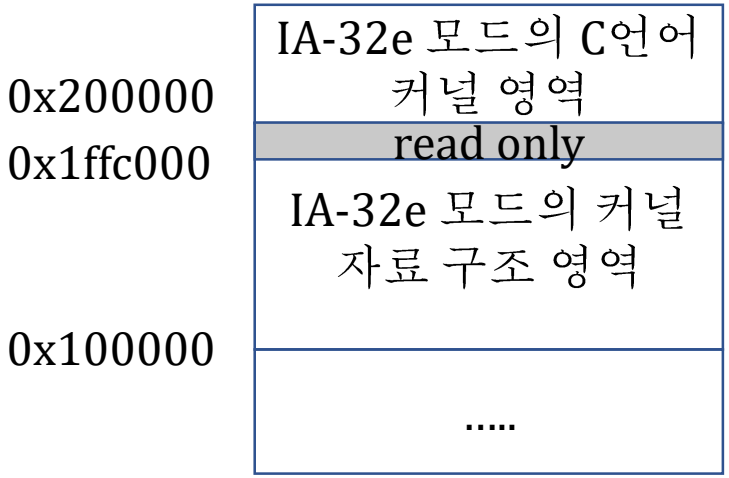
기대하는 출력 값

MINT64 OS Boot Loader Start~!!
Current Time: 23:12:44
OS Image Loading… Complete~!!
OS Image Checking… Okay~!!
Switch To Protected Mode Success~!!
…..
IA-32e C Language Kernel Start …….. [Pass]
This message is printed through the video memory relocated to 0xAB8000

Homework #3

- Finer-grained Page Table
  - 우리는 페이지 테이블을 이용해 커널 자료구조 영역의 일부 영역을 읽기 전용으로 만들어 보고자 한다.
  - 기본 구현에서는 2MB 단위의 페이지 디렉토리 까지만 사용한다.
  - 이것은 너무 coarse-grained하여 우리의 목표를 달성하기에 적합하지 않다. 우리는 4KB의 페이지 테이블 엔트리를 이용해 VA-to-PA 메모리 맵핑을 해야만 할 것이다.





Homework #3

- Read-only page
  - Read only page를 만들기 위해선 페이지 테이블 엔트리의 다양한 flag 중 R/W를 수정하면 된다.
  - 다만! 커널 모드에서는 페이지 테이블의 R/W에 상관없이 무조건 읽기와 스기가 가능하다.
  - 따라서 커널도 R/W flag에 영향을 받기 위해선 CR0레지스터의 WP를 1로 설정해야만 한다.

기대하는 테스트 결과

- 페이지 테이블 설정 전
  - Kernel64의 main함수에서 주소 0x1ffc000에서 값에 값을 읽고 쓸 수 있는 것 확인
- 페이지 테이블 설정 후 (재컴파일)
  - Kernel64의 main함수에서 주소 0x1ffc000에서 값에 값을 읽을 수는 있지만 쓸 수는 없는 것 확인