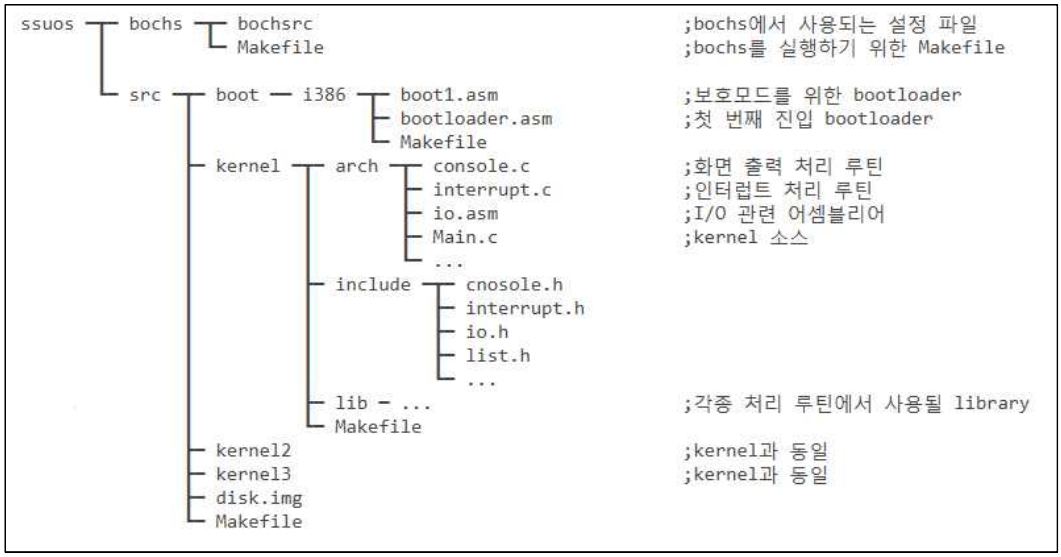
과제 #2 : Bootstrap

○ 과제 목표

- 부트로더 이해 및 분석
- 리얼모드에서 보호모드로 전환 과정 이해
- 부트로더에서 키보드 인터럽트를 이용한 커널 선택

○ 기본 배경 지식

- SSU OS 소스 목록



- 리얼모드
  - ✓ 80286 이후의 x86 호환 CPU의 운영 방식
  - ✓ 리얼 주소 모드 혹은 호환 모드라고도 하며, 80186 계열 CPU와 호환을 위해 만듬
  - ✓ 최대 1 MB의 메모리가 번지에 기록될 수 있음
  - ✓ 80286 계열의 CPU는 전원이 켜질 때 리얼모드로 동작함
  - ✓ 80186 계열의 CPU는 하나의 운영방식만 존재했으며, 리얼모드와 동일
- 보호모드
  - ✓ 보호 가상 주소 모드라고도 하며, x86 호환 CPU의 운영 방식
  - ✓ 시스템 소프트웨어가 다중 작업, 가상 메모리, 페이징, 그리고 응용 소프트웨어를 넘는 운영 체제 제어 능력을 높이기 위해 고안된 운영 체제의 다른 기능들을 이용할 수 있게 도와줌
  - ✓ 일반적으로 CPU는 BIOS 이후 리얼모드에서 보호모드로 전환
  - ✓ GDT 등의 부가적 정보가 필요

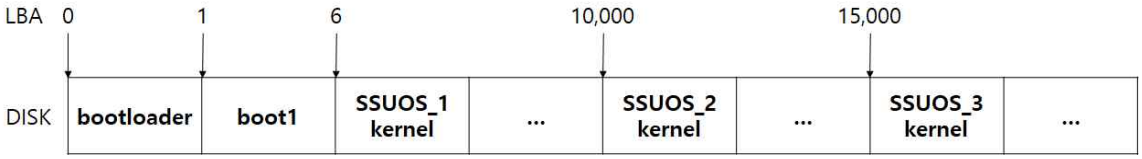
- GDT (Global Descriptor Table)
  - ✓ 인텔 x86 계열 CPU가 사용하는, 특정 레벨 메모리 영역의 사용 범위와 특성을 정의하기 위한 테이블 자료구조
  - ✓ 보호모드에 들어가기 위해서는 NULL, 코드 세그먼트, 데이터 세그먼트를 서술한 GDT가 필요
- IDT (Interrupt Descriptor Table)

✓ 인텔 x86 계열 CPU가 사용하는, 인터럽트 벡터 테이블을 정의하기 위한 자료구조
✓ CPU가 인터럽트와 예외에 대해 정확한 답변하기 위해 사용
✓ 인터럽트가 발생하였을 때 처리해주는 함수의 루틴을 포함

- MBR (Master Boot Record)
✓ 파티션된 기억 장치의 첫 섹터로 부트 섹터임
✓ 부팅 시 메모리로 올라간 뒤 저장된 코드가 실행됨
✓ 부트 코드(446 Byte), 파티션 테이블(64 Byte), 시그니처(2 Byte)로 나누어져 있음
- MBR에서 동작하는 코드의 길이는 446 Byte를 넘을 수 없음

○ 과제 수행방법

- 리얼모드에서 키보드 인터럽트를 통한 커널 선택
✓ 부팅 시 SSUOS_1, SSUOS_2, SSUOS_3 에 대하여 키보드 방향키(상, 하, 좌, 우)를 통해 커널 선택
✓ 커널 선택 후 엔터 입력 시 선택된 커널로 부팅
- 디스크 이미지 파일에 3개의 커널이 있음
- 각 커널의 LBA(Logical block addressing)를 CHS(Cylinder-Head-Sector)로 변환 후 디스크 인터럽트 코드에 삽입



- (1) boot1.asm 소스 분석
✓ 운영체제 공지사항 게시판에서 소스 파일과 함께 코드분석 한글 파일 다운로드
✓ 소스 코드 줄 단위 빈칸 채우기

- (2) src/boot/i386/bootloader.asm 추가, 수정

```
org     0x7c00

[BITS 16]

START:
                jmp             BOOT1_LOAD ;BOOT1_LOAD로 점프

BOOT1_LOAD:
                mov     ax, 0x0900
        mov     es, ax
        mov     bx, 0x0

                ;0x13 인터럽트는 DISK I/O, 64비트책 p136 참조
        mov     ah, 2           ;0x13 인터럽트 호출시 ah에 저장된 값에 따라 수행되는 결과가 다름. 2
는 섹터 읽기
        mov     al, 0x4         ;al 읽을 섹터 수를 지정 1~128 사이의 값을 지정 가능
```

```
        mov     ch, 0               ;실린더 번호 cl의 상위 2비트까지 사용가능 하여 표현
        mov     cl, 2               ;읽기 시작할 섹터의 번호 1~18 사이의 값, 1에는 부트로더가 있으니 2
이상부터
        mov     dh, 0               ;읽기 시작할 헤드 번호 1~15 값
        mov     dl, 0x80    ;드라이브 번호. 0x00 - 플로피; 0x80 - 첫 번째 하드, 0x81 - 두 번째 하드

        int     0x13                ;0x13 인터럽트 호출
        jc      BOOT1_LOAD          ;Carry 플래그 발생시(=Error) 다시 시도
```

✓ 커널 선택을 위한 코드 수정 및 삽입

- [ ]SSUOS_1, [ ]SSUOS_2, [ ]SSUOS_3 출력
- 키보드 인터럽트를 이용하여 커널 선택 구현
- PTE 값을 보고 KERNEL_LOAD 부분을 수정

```
KERNEL_LOAD:
        mov     ax, 0x1000
        mov     es, ax
        mov     bx, 0x0

        mov     ah, 2
        mov     al, 0x3f
        mov     ch, 0
        mov     cl, 0x6
        mov     dh, 0
        mov     dl, 0x80

        int     0x13
        jc      KERNEL_LOAD
```

✓ 커널 선택을 위한 코드 수정 및 삽입

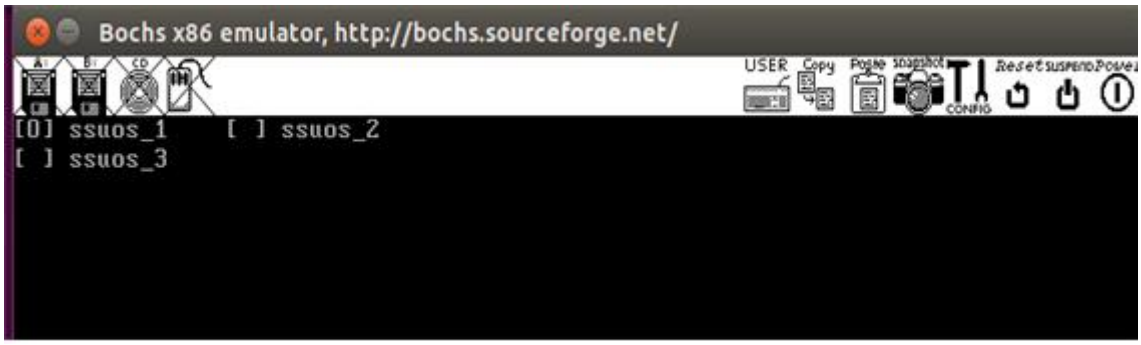
- [ ]SSUOS_1, [ ]SSUOS_2, [ ]SSUOS_3 출력
- 키보드 인터럽트를 이용하여 커널 선택 구현
- PTE 값을 보고 KERNEL_LOAD 부분을 수정

```
KERNEL_LOAD:
        mov     ax, 0x1000
        mov     es, ax
        mov     bx, 0x0

        mov     ah, 2
        mov     al, 0x3f
        mov     ch, 0
        mov     cl, 0x6
        mov     dh, 0
        mov     dl, 0x80
```

```
        int     0x13
        jc      KERNEL_LOAD

jmp                 0x0900:0x0000

select db "[O]",0
ssuos_1 db "[ ] SSUOS_1",0
ssuos_2 db "[ ] SSUOS_2",0
ssuos_3 db "[ ] SSUOS_3",0
partition_num : resw 1

times   446-($-$$) db 0x00

PTE:
partition1 db 0x80, 0x00, 0x00, 0x00, 0x83, 0x00, 0x00, 0x00, 0x06, 0x00, 0x00, 0x00, 0x3f, 0x0, 0x00, 0x00
partition2 db 0x80, 0x00, 0x00, 0x00, 0x83, 0x00, 0x00, 0x00, 0x10, 0x27, 0x00, 0x00, 0x3f, 0x0, 0x00, 0x00
partition3 db 0x80, 0x00, 0x00, 0x00, 0x83, 0x00, 0x00, 0x00, 0x98, 0x3a, 0x00, 0x00, 0x3f, 0x0, 0x00, 0x00
partition4 db 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
times     510-($-$$) db 0x00
dw        0xaa55
```

○ 과제 수행 전 실행 결과

<bochs GUI>

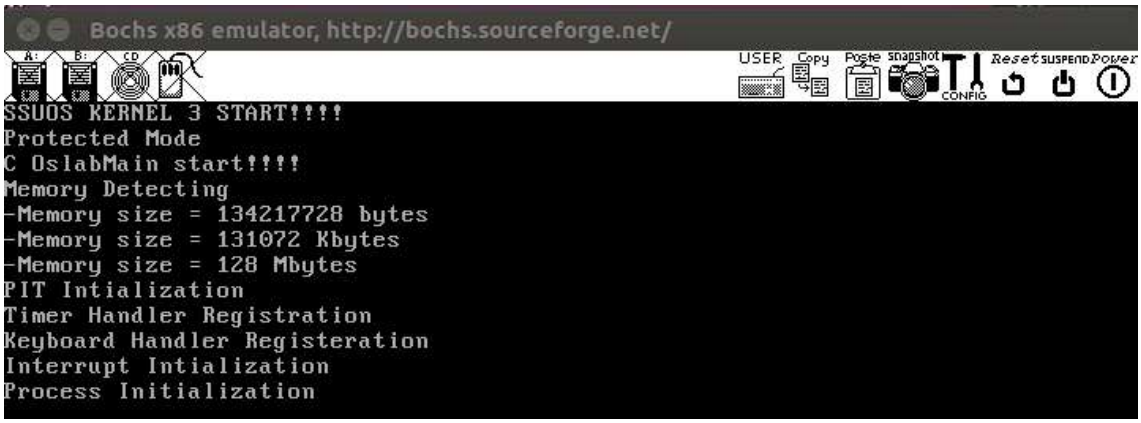


<과제 실행 후 초기 화면>

<키보드의 상,하,좌,우 방향키로 부팅할 커널 선택 후 엔터 입력 시 부팅>

<SSUOS_1 부팅 시 실행 결과>

<SSUOS_2 부팅 시 실행 결과>



<SSUOS_3 부팅 시 실행 결과>

- 과제제출
  ✓ 2019년 09월 10일 (화) 23시 59분까지 제출

- 배점 기준
  ✓ 보고서 10%
  - 개요 2%
  - 상세 설계 명세 5%
  - 실행 결과 3%

  ✓ 소스분석 25%

  ✓ 소스코드 65%
  - 컴파일 여부 5%(설계 요구에 따르지 않고 설계된 경우 0점 부여)
  - 실행 여부 60%(커널 이름 출력 5% + 키보드 인터럽트로 커널 선택 25% + 선택된 커널 부팅 30%)

- 최소 구현사항
  ✓ 커널 이름 출력 + 키보드 커널 선택 + 선택된 커널 부팅