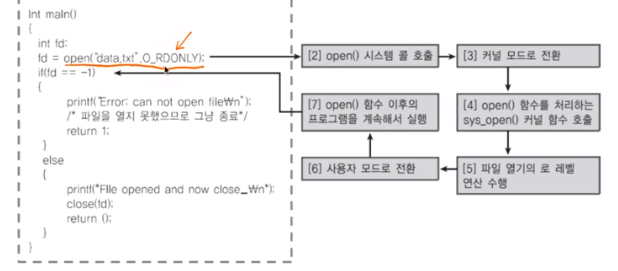
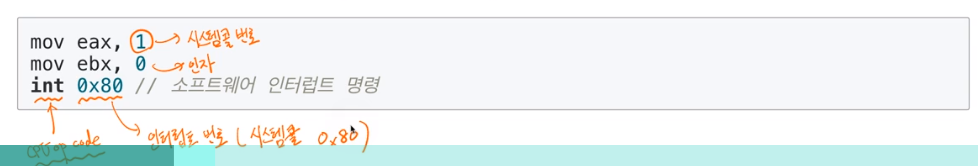
Chap22 인터럽트 내부 동작

**@시스템 콜 인터럽트**



* 시스템 콜 실행을 위해서는 강제로 코드에 인터럽트 명령을 넣어, Cpu에 실행시켜야 한다.
* 시스템 콜 실제 코드
* Eax 레지스터에 시스템 콜 번호를 넣고
* Ebx 레지스터에는 시스템 콜에 해당하는 인자값을 넣고, 그림 화살표의 인자들 read only



* 인터럽트는 코드 안에서 강제 실행할 수 있다 시스템콜 0x80처럼 이상한 언어인데 인터럽트는 외부의 자극이라는 것을 생각하면

\*커널모드영역은 커널모드에서 실행되는 코드들이 있는 특정 메모리 공간이라고 보시면 됩니다.

사용자모드영역 메모리와는 다릅니다.

인터럽트와 IDT

인터럽트 발생시 IDT에서 확인

시스템콜 인터럽트 명령은 0x80이 미리 정의되었기에 운영체제 코드로 system\_Call이라는 함수다

따라서 IDT에는 위의 정보가 주소처럼 나열되어 있다.

* 리눅스의 인터럽트 번호 예
* 0~31 : 소프트웨어/내부 인터럽트(일부는 정의 안됨)
* 32~47 : 하드웨어 인터럽트 (주변장치 종류 개수에 따라 변경)
* 128: 시스템 콜 0x80번이 128번 입니다.

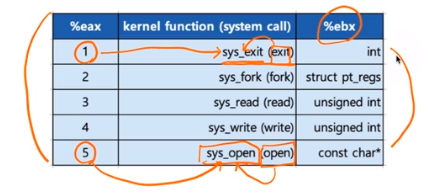
**@인터럽트와 시스템 콜 (고급)**

* 시스템콜 인터럽트 명령을 호출하면서 0x80값을 넘겨준다.

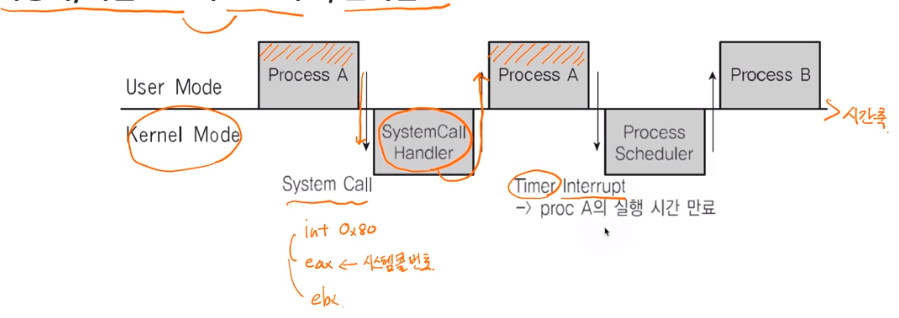
1. Cpu는 사용자 모드를 커널 모드로 변경
2. idt에서 0x80에 해당하는 주소(함수) 찾아서 실행

idt는 인터럽트 테이블인데 /인터럽트 번호 :주소(시스템 함수 코딩이 연결되어 있다.)

마치 단축키처럼



**@사용자.커널모드와 프로세스 인터럽트**



사용자 모드로 A하다 시스템 콜 발생해서 잠시 커널모드되었다가 다시 사용자모드 가서 A하다가 이번엔 인터럽트 발생해서 Timer 시간초과 인터럽트로 A가 완료되고 B가 다시 사용자모드로 실행되고

프로세스는 일정히 작동되는 것인게 아닌 이렇게 여러 모드를 와리가리하면서 복합적으로 작동된다.

**@인터럽트와 IDT**

* 인터럽트는 미리 정의되어 각각 이벤트 번호와 실행코드를 가리키는 주소가 기록되어져 있다.
* 어디에? IDT 테이블에 기록을 활용한다.
* 언제?