**[Difference between context switch and mode switch in Linux](https://daehee87.tistory.com/473)**

[Programming](https://daehee87.tistory.com/category/Programming) 2014. 12. 19. 17:33

Context 스위칭과 kernel-user 모드스위칭은 다르다. 두가지 경우 모두 현재 레지스터 컨텍스트를 메모리에 저장함과동시에 기존에 저장해놨던 레지스터 컨텍스트를 메모리로부터 복원한다는 점이 동일해서 햇갈리기가 쉬운데, 둘을 잘 구분할 필요가 있다.

프로세스 A 가 User 모드에서 작동하다가 시스템콜이나 인터럽트에 의해 커널모드로 진입시에는 유저모드에서의 레지스터 컨텍스트가 프로세스 A 의 커널스택에 저장되고, 커널스택에 저장되어있던 프로세스 A 의 커널모드 레지스터 컨텍스트가 복원된다.

프로세스 A 가 Context Switching 을 할때는 이미 프로세스 A 가 반드시 커널모드인 상태이다. 이 경우는 프로세스 A 의 커널모드 레지스터 컨텍스트가 메모리에 저장되면서 다음번 스케줄될 프로세스 B 의 레지스터 컨텍스트(커널모드의) 가 복원된다.

즉 유저 프로세스가 돌다가 갑자기 한번에 스케줄이되서 프로세스 B 로 넘어가는것 같지만 정확한 순서는

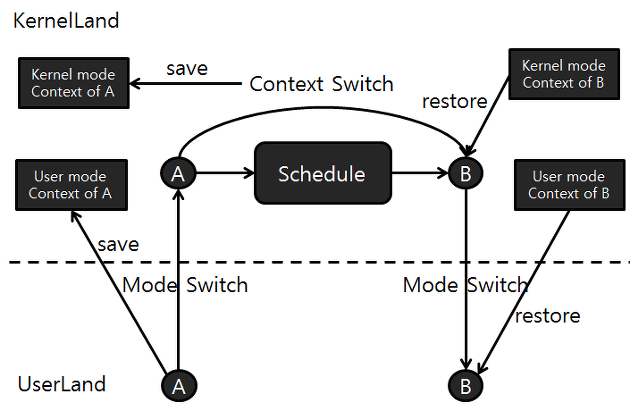
1. 유저프로세스 A 가 커널모드로 진입하면서 유저모드 레지스터 컨텍스트를 A 의 커널스택에 저장

2. 프로세스 A 는 커널컨텍스트를 가진상태로 Schedule 을 호출

3. A 의 커널컨텍스트가 메모리에 저장되고 프로세스 B 의 저장된 커널컨텍스트를 레지스터에 복원

4. B 의 커널컨텍스트가 메모리에 저장되면서 B 의 유저컨텍스트를 레지스터에 복원하면서 유저모드로 리턴

그림으로 표현하면 아래와 같다.



무한루프를 돌던 프로세스 A 가 프로세스 B 로 스케줄이 되는 과정을 묘사하면, 먼저 프로세스 A 는 유저모드에서 커널모드로 Mode Switching 을 하고 (타이머 인터럽트에 의해서), 그러면서 타이머 인터럽트가 발생하던 시점의 유저모드 레지스터 컨텍스트를 커널스택에 저장한뒤에 Schedule 을 호출, Schedule 시에는 다시 A 의 커널모드 레지스터 컨텍스트를 메모리에 저장해놓고 B 의 커널모드 레지스터 컨텍스트를 복원, B 는 이전에 멈췄던 Schedule 시점에서 다시 resume 하여 유저모드로 나가기위한 실행흐름을 타고, 예전에 저장해놓았던 B 의 유저모드 레지스터 컨텍스트를 복원하면서 sysret, iret 등으로 유저모드 복귀후 실행.

모드스위칭 과정은 아키텍쳐마다 상이하고 H/W 적인 지원이 같이 들어가기때문에 조금더 복잡하다. Context Switching 의 경우도 훨씬 복잡하지만 아주 대략적으로 묘사하자면 이정도로 정리할수 있다.

출처: <https://daehee87.tistory.com/473> [SHALL WE PLAY A GAME?]