OS_Project 2

2014-1995 송원재

1. _os_context_t
-edi, esi, ebx, edx, ecx, eax, _eflags, resume_eip 를 low→high 방향으로 저장

2. _os_save_context()
-인라인 어셈블리어를 사용하여 resume_eip, _elfags, eax, ecx, edx, ebx, esi, edi 의 순서대로 push
-지금의 stack_pointer 값을 return 해야 하므로 eax 에 esp 의 값을 씌움.
-restore 할 때 NULL 값을 return 해야하므로 eax 의 값을 저장한 스택에 0(NULL)을 씌움.
-현재 stack 을 보존하기 위해 기존의 old ebp, old eip 를 push 한 후에 leave, return 하여 _os_schedule()로 복귀

3. _os_restore_context(addr_t sp)
- esp 를 sp 의 위치로 보낸 후 context 를 pop 한다.(edi, esi, ebx, edx, ecx, eax, _eflags)
- ebp 를 old ebp 위치의 stack 에 보낸다.
- ret 을 실행하면 resume_eip 로 이동

4. _os_create_context()
- stack_base 에 stack_size 를 더하여 주어진 stack 의 가장 high address 로 이동.
- 그 위로 arg, ret_address(NULL), entry, _eflags(1), 레지스터 6 개(모두 NULL)값 저장 하고 stack 의 가장 low address return

5. eos_tcb_t
- task 의 state, priority, period, stack_size, stack_base, stack_pointer, node 멤버 변수를 선언

6. eos_create_task()
- os_create_task() 함수 호출하여 task 생성한 후에 task 의 tcb 에 각 특성들을 저장.
- 그 후에 os_add_node_tail()함수를 이용해 os_ready_queue 에 넣어줌.

7. eos_schedule()
- os_current_task 가 NULL 아 아닐 때, os_save_context()함수 실행하여 context 를 저장한다. 이 때 수행하던 task 의 context 를 온전히 저장하기 위해 os_current_task 의 값만 NULL 과 비교한 후 바로 os_save_context()를 수행한다.
- os_save_context()의 return 값이 NULL 이 아니면 save 한 context 의 스택주소이므로 이를 task tcb 에 저장하여 후에 restore 할 수 있게 한다.
- os_save_context()의 return 값이 NULL 일 경우 restore_context()에서 resume_eip 로 넘어와 return 한 경우이니 추후 과정없이 바로 return 하여 task 수행을 재개한다.
- switch 할 task 를 찾기 위해서 os_ready_queue 에서 priority 가 가장 high 한 task 를 찾는다.
- 자기 자신의 값과 다른지 확인하고 해당 task 로 os_restore_context()수행.
- os_restore_context()를 수행하면 resume_eip 로 건너가 os_save_context()에서 NULL 로 return 하게 되어 해당 task 작업을 수행하게 됨.