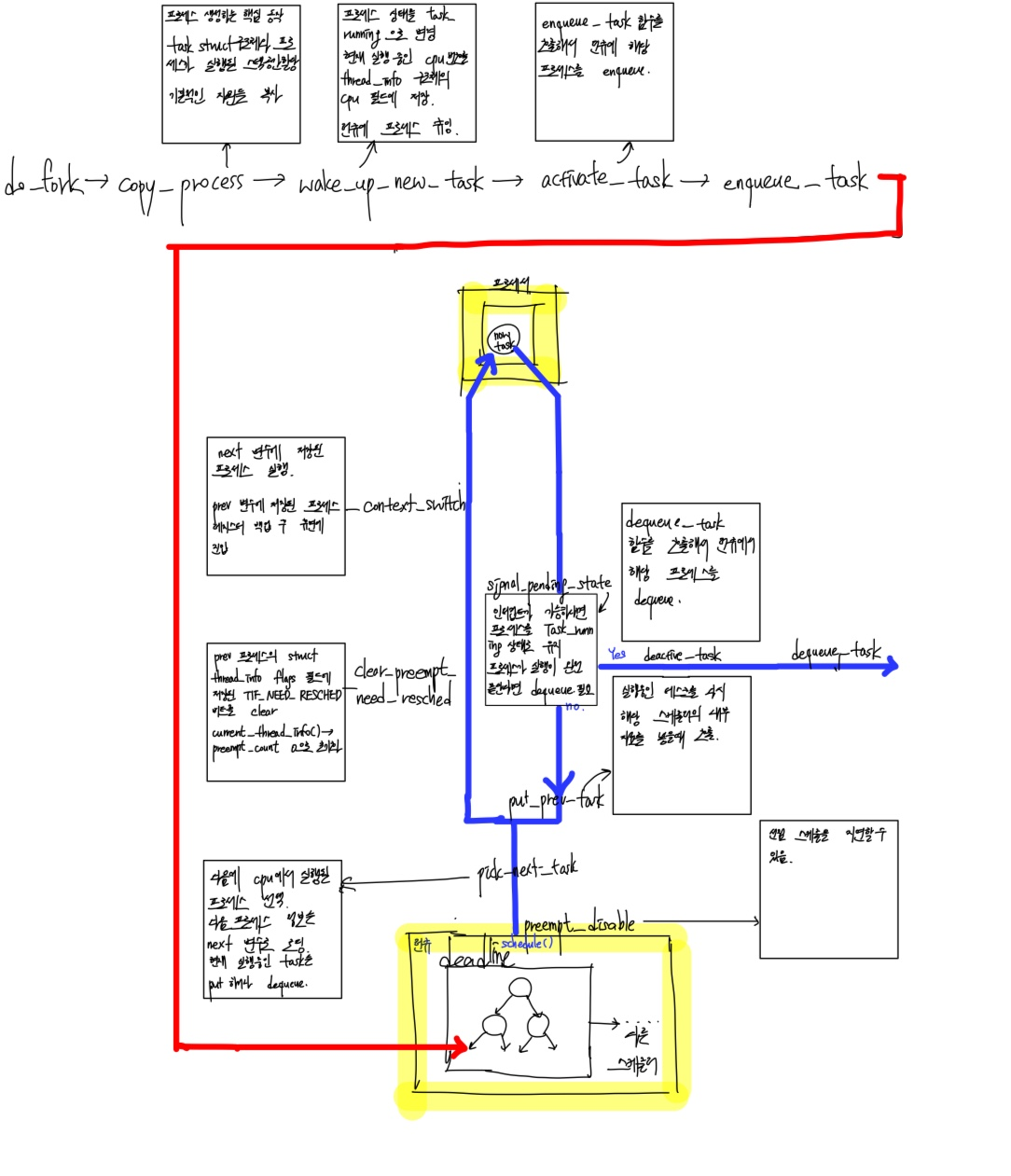
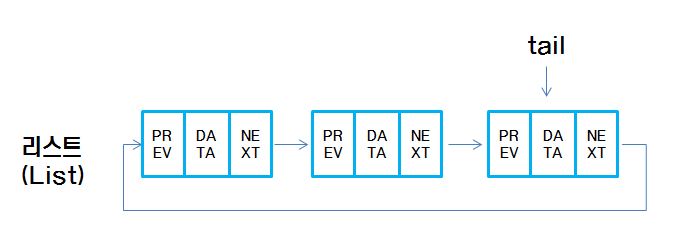
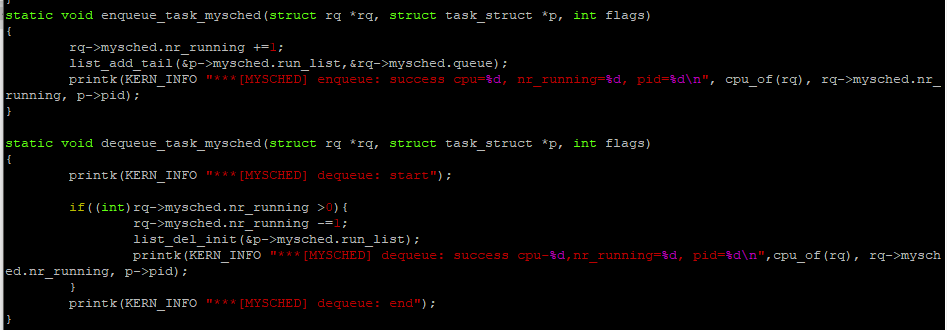
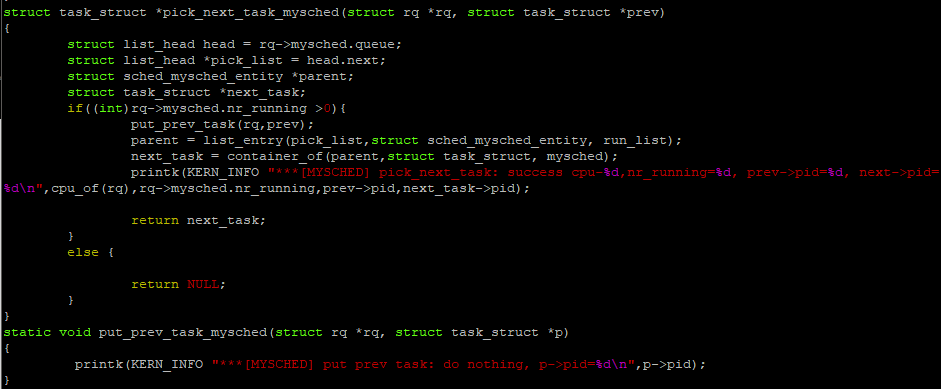
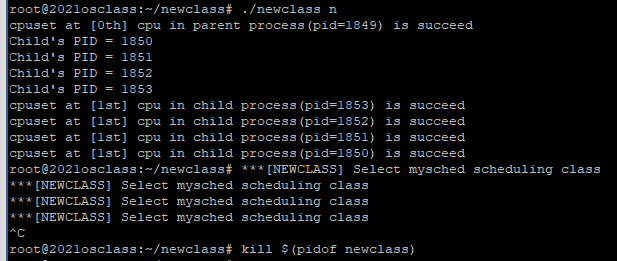
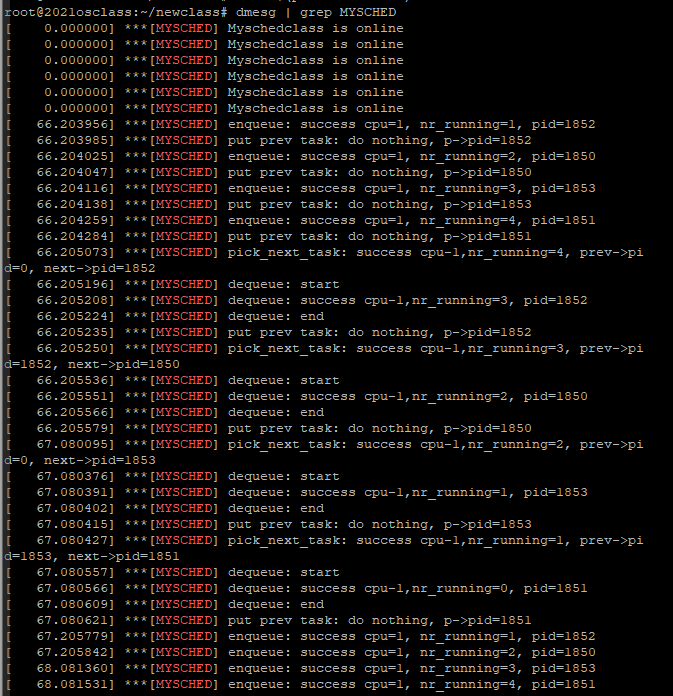
**2019062833 컴퓨터 소프트웨어학부 김유진**

**운영 체제 HW#8**

**제출 일자: 2021/05/20**

1. **과제 1: \_\_schedule() 함수의 흐름도**
   1. 빨간줄 : do\_fork 부터 enqueue까지의 흐름
   2. 파란줄: schedule()



1. **과제 2(가상머신 링크:** <https://drive.google.com/file/d/13-sg98uXfE-Lm2aAKDWyl4CwMhrnZR7F/view?usp=sharing>)
2. **리눅스 스케줄러 전체 동작에 대한 간단한 설명**
   1. 어떤 프로세스의 task state가 running 상태로 전이되면 각 스케줄러가 가진 서브 런큐에 task를 enqueue 한다.
   2. schedule 함수가 호출되면 pick next task 함수 호출로 다음으로 CPU에서 실행될 프로세스를 선택하고 현재 실행중인 프로세스는 put prev task 함수 호출을 통해 다시 서브 run queue에 넣거나 종료된 경우 dequeue 한다.
   3. pick next task 함수로에서 선택된 프로세스가 이 후 CPU에서 실행되게 된다.
3. **mysched 스케줄러 동작에 대한 설명(FIFO)**
   1. FIFO 기반의 스케줄링을 위해 이중 원형 링크드 리스트를 활용한다.
   2. enqueue되는 프로세스는 head prev, 즉 가장 뒤쪽으로 보낸다.
   3. 이 후 next task를 선택할 시 head next를 고르게 해서 가장 먼저 들어온 task가 가장 먼저 실행될 수 있도록 한다.
4. **mysched.c 파일에 대한 설명**
   1. 자료 구조
      1. 이중 원형 링크드 리스트
         1. 
         2. 양방향으로 연결이 되어있어서 노드 탐색이 양쪽으로 가능한 이중 링크드 리스트의 특징을 가진다.
         3. 원형 리스트로 헤드 노드와 테일 노드가 연결되어있다.
   2. 함수
      1. enqueue task mysched와 dequeue task mysched
         1. 
         2. Enqueue
            1. mysched 서브런큐의 nr\_running 값을 1 증가시켜 준 후 이중 원형 링크드 리스트에 넣어준다.
            2. FIFO 작동을 위해 list\_add\_tail 함수를 사용하여 task를 head-prev에 넣어준다.
            3. dmesg 시 확인을 위해 printk 함수로 메시지를 띄운다.
         3. Dequeue
            1. 만약 해당 서브런큐에 running 값이 0보다 크다면 현재 큐에 요소가 하나 이상임을 의미하므로 dequeue 진행이 가능하다.
            2. nr\_running 값을 1 감소시키고 list del init 함수로 인자로 받아온 p task를 리스트에서 삭제함과 동시에 prev, next 값을초기화를 시킨다.
            3. 마찬가지로 dmesg 시 확인을 위해 printk 함수로 메시지를 띄운다.
      2. pick next task mysched와 put prev task mysched
         1. 
         2. Put Prev task mysched
            1. p의 pid 값만 출력하는 기능을 한다.
         3. Pick next task mysched
            1. 만약 해당 스케줄러의 서브런큐의 요소가 하나 이상이라면 put prev task 함수를 호출하여서 현재 실행중인 task를 run queue로 가져온다
            2. 이 후 list entry 함수를 통해 head->next가 가리키는 run list를 가지 부모 struct를 찾아오고 이 구조체를 container of 함수에 넣어주어서 task struct를 찾아온다.
            3. 받아온 최종 task struct를 next struct 값으로 반환한다.
5. 최종 결과(dmesg 결과)
   1. 
   2. 
   3. **결과**
      1. 처음 초기화 작업을 마치고 1852 프로세스가 가장 먼저 enqueue 되고 put prev task mysched 함수가 호출됨
      2. 마찬가지로 1850, 1853, 1851 순서로 차례로 enqueue 된다
      3. CPU에서 실행될 프로세스를 고르는 pick next task가 불리게 되고 그 결과 가장 먼저 enqueue된 1852 프로세스가 선택된다.
      4. 일을 마친 1852 프로세스는 dequeue되어 나가게 된다.
      5. 이 후 들어온 순서대로 next task로 선택되어 CPU에서 실행되게 되고 일을 끝마치면 dequeue 된다.