기말고사 대비 문제 모음

대부분의 질문에 대한 답변은 강의 내용을 넣어서 작성할 것

1. 운영체제란?

운영체체에 대한 내용 서술

ex) 하나의 소프트웨어이다. 하드웨어를 관리하는 소프트웨어이다. 등등

운영체제란, 하나의 소프트웨어이다. 하드웨어를 관리하느 소프트웨어

2. 전체적인 챕터 검증

1. 배워야 하는 이유
2. 운영체제 주요 특성 중 어디에 해당하는지

- 프로세스 관리
    - 배워야 하는 이유: 프로세스 관리는 컴퓨터로 여러 프로그램을 실행하기 위해 필수적인 기능이다. 프로세스를 스케줄링하고 적절하게 할당하여 원활하게 동작하여야 올바른 운영체제라고 할 수 있다. 이러한 동작 기능을 이해해야 절절한 프로그램을 설계할 수 있다.
    - 주요 특성: 효율적인 측면이 강하다. 사용자가 관여할 수 있는 부분이 아니기 떼문에 O/S에서 적절한 프로세스를 스케줄링하기에
- 메모리 관리
    - 배워야 하는 이유: 메모리는 한정적인 자원이기에 이를 관리하는 것은 매우 중요하다. 메모리 관리는 프로세스 관리와도 밀접한 관련이 있기때문에 같은 이유로 배워야 한다. 실제 코드를 작성할 때도 메모리관리 측면은 매우 중요한 개념이다.
    - 주요 특성: 마찬가지로 효율, O/S적인 개념이 강하다.
- 파일 관리
    - 배워야 하는 이유: 실제 Disk에서 프로그램이 어떻게 load되고 프로세스가 되는지의 일련의 과정이다. 마찬가지로 앞의 메모리, 프로세스에도 밀접한 관련이 있지만 그전에 일어나는 선행 과정이다. 다른 관리와 마찬가지로 다양한 알고리즘 접근 방식이 존재하낟.(로지컬 피지컬)
    - 주요 특성: 사용자에 가깝기 때문에 편리성이 좀 더 강하다. 사용자가 파일을 직접 관리하기에.. 그렇다고 내부 동작이 간단한 것은 아니다. 다양한 알고리즘
- 입출력 관리
    - 배워야 하는 이유: 입출력 또한 앞선 운영체제의 다양한 특성에 밀접한 관련이 있지만 모든 요즘의 모든 프로그램은 I/O 관리가 매우 빈번하다. 따라서 실제 컴퓨터가 동작하는 (키보드를 입력 하고 모니터에 표시되는) 과정을 이해하기 위해선 I/O 관리를 이해해야 한다.
    - 주요 특성: 사용자에게 가깝~

3. scheduling에 대한 정의

(1+1) 일반적으로 scheduling 이란 무엇이며 언제 scheduling 이 발생하는가?

(CPU 스케줄링, I/O 스케줄링을 구분하여 각각 제시해야 함)

스케줄링이란 Task를 관리하는 것인데, O/S관점에서 스케줄링은 프로세스, 쓰레드, I/O등을 스케줄링한다고 할 수 있다. 각 우선순위에 맞게 인터럽트로 스케줄링을 발생시킨다.

- CPU 스케줄링
  - CPU스케줄링은 프로세스를 스케줄링하는 것이다. 프로세스는 CPU를 할당받아 실행되는데, 이때 CPU를 할당받기 위해선 스케줄링이 필요하다. 스케줄링은 프로세스의 상태에 따라 다르게 스케줄링이 발생한다.
- I/O 스케줄링
  - I/O 스케줄링은 다양한 입출력 처리에 대한 우선순위를 스케줄링하는 것이다. CPu와 마찬가지로 다양한 스케줄링 알고리즘이 있다.

4. 멀티 프로그래밍

1. 멀티 프로그래밍이란 무엇이고, 이것이 가능하려면 운영체제는 무엇을 제공해야 하는가

멀티 프로그래밍이란, 한 프로세서에서 여러 프로그램이 실행되는 것 처럼 보이는 것이다. 이를 위해선 CPU 스케줄링이 필요하다. CPU 스케줄링은 프로세스의 상태에 따라 다르게 스케줄링이 발생한다.

이 스케줄링의 전환 속도가 매우 빠르기 때문에 우리는 프로세스위의 프로그램들이 동시에 실행되는 것 처럼 보이게 된다.

5. 컴퓨터 시스템 하드웨어적 / 소프트웨어적구성

컴퓨터 시스템은 하드웨어적 / 소프트웨어적으로 어떻게 구성되어 있는가? 각각 구성에 대해서 중요 요소를 거론하고 간단히 설명하라.

- 하드웨어적 구성
  - 컴퓨터 시스템은 하드웨어적으로 크게 Disk, Cpu, Ram, I/O 장치로 구성되어 있다. 이들은 각각의 역할을 수행하며, 이들의 역할을 O/S가 관리한다.
- 소프트웨어적 구성
  - 소프트웨어적 구성은 O/S와 Application으로 구성되어 있다. O/S는 하드웨어를 관리하며, Application은 O/S위에서 동작하는 프로그램이다. O/S에서 중요한 지점은 내부에 존재하는 커널이라는 소프트웨어이다. 실제로 시스템 콜또한 커널의 동작이고 우리는 여기서 요청을 보내는 것이다.

6. 파일을 구성할 경우 필요한 전체 사이즈 계산

2000K 의 텍스트 파일이 있다. 블록 사이즈를 120K 이라고 가정 할 때 인덱스 와 링크 리스트 방법으로 파일을 구성할 경우 필요한 전체 사이즈를 각각 제시하라. (인덱스 크기는 120K, 포인터 크기는 10K)

링크 리스트 방식: 한 블록이 120K이지만 포인터의 크기가 10k이므로 저장할 수 있는 영역은 110K이다. 따라서 2000/110 = 18.18... 이므로 19개의 블록이 필요하다. 19*120 = 2280K

인덱스 방식: 인덱스 블록으로 1개의 블록을 사용하고 2000k/120k = 16.66... 이므로 17개의 블록이 필요하다. 따라서 18개의 블록이 필요

7. disk access time 이란?

disk access time 이란? 이것은 무엇에 의해서 지배를 받으며 시스템 효율에 어떠한 영향을 미치는지 상세히 논의하라.

disk access time이란, 디스크에 접근하는데 걸리는 시간을 말한다. 이는 디스크의 헤드가 움직이는 시간, 디스크의 회전이 멈추는 시간, 데이터를 읽어오는 시간으로 구성된다.

디스크 내부적으로 접근하려는 데이터가 disk track에서 멀리 떨어져 있을수록 disk access time은 길어진다. 이는 시스템의 효율을 저하시키는 요소이다.

따라서 다양한 접근 알고리즘이 존재한다. S-Scan or FIFO, SSTF, SCAN, C-SCAN, LOOK, C-LOOK 등이 있다.

8.메모리 관리 모듈의 요소 하드웨어 및 소프트웨어 관점

(1.5점+1.5점) 운영체제 중 메모리 관리 모듈의 요소를 하드웨어 및 소프트웨어 관점에서 제시하고, 이 운영체제가 가상 메모리를 가능하게 하기 위해서 필요한 요소를 하드웨어 및 소프트웨어 관점에서 구체적으로 제시하라.

메모리 관리 모듈의 요소는 크게 메모리 할당, 메모리 보호, 메모리 공유, 주소 변환으로 나뉜다.

메모리를 할당하고 이를 logical주소를 physical주소로 변환해주는 것이 MMU이다. 이는 하드웨어적으로 구현되어야 한다.

가상 메모리는 요구 페이징으로 구현된다.

필요한 프로그램의 일부만 메모리에 적재시켜 흉율적으로 메모리를 사용할 수 있다.

요구 페이징은 pageFalut(인터럽트)를 발생시켜 OS에게 요청하여 필요한 페이지를 메모리에 적재시킨다.(페이지 테이블)

9. 라운드 로빈 스케줄러

(2점) 라운드 로빈 스케줄러는 일반적으로 정확히 한번 나타나는 각 프로세스들과 함께 모든 수행 중인 프로세스들의 리스트를 유지한다. 만약 프로세스가 리스트에서 두 번 나타난다면 어떻게 되는가? 이것을 허용한 이유는 어떤 장점을 얻고자 함인가? 단점은 무엇인가?

라운드 로빈 스케줄러란, 스케줄링은 대화형 시스템에서 사용되는 선점 스케줄링 방식이다.

각 프로세스들은 동일한 크기의 할당 시간을 가지며, 할당 시간이 지나면 프로세스는 선점되고 준비 큐의 끝으로 이동한다.

(시간 할당량)

때문에 두번 등장했다고 하는 것은 할당 시간이 끝나고 다시 준비 큐의 끝으로 이동했다는 것이다.

이는 한 가지 프로세스에 고착되지 않고 다른 프로세스가 무기한 기다려야 하는 상황을 방지할 수 있다.(무한루프)

때문에 할당 시간을 두고 프로세스를 선점하는 것이다.

단점은 할당시간이 너무 짧으면 너무 많은 오버헤드가 발생하고 너무 길면 응답시간이 느려진다.

10. Page Fault(복습)

페이지 부재 시 발생하는 일련의 과정이다. 이 때 아래 문제에 대해 적절한 답변을 서술하시오

1. Trap to the operating system
2. Save the user registers and process state
3. Determine that the interrupt was a page fault
4. Check that the page reference was legal and determine the location of the page on the disk
5. Issue a read from the disk to a free frame:
    1. Wait in a queue for this device until the read request is serviced
    2. Wait for the device seek and/or latency time
    3. Begin the transfer of the page to a free frame
6. While waiting, allocate the CPU to some other user
7. Receive an interrupt from the disk I/O subsystem (I/O completed)
8. Save the registers and process state for the other user
9. Determine that the interrupt was from the disk
10. Correct the page table and other tables to show page is now in memory
11. Wait for the CPU to be allocated to this process again
12. Restore the user registers, process state, and new page table, and then resume the interrupted instruction

1. 2번의 과정에서 해당 프로세스의 상태가 변화하는 원인과 과정을 서술하시오(프로세스 상태도를 고려)

page fault가 발생하여 os를 통해서 요청해야 한다.

Cpu에서 실행중인 프로세스가 page fault가 발생하면 해당 프로세스는 block 상태가 되고 PCB에 page fault가 발생한 주소를 저장한다.(정보 저장)

2. 위의 과정에서 Context Switch가 발생하는 과정 번호와 발생하는 원인, 위의 과정에서 Context Switch가 발생하는 총 횟수를 서술하시오.

2번) 그 전에 진행중이었던 프로세스를 PCB에 저장하고 새로운 일을 시작하기 위해서

6번) Waiting을 하는 동안, CPU가 놀면 안되니 other user의 프로세스를 진행하기위해서

8번) Waiting이 종료되었기 때문에 other user의 프로세스를 중단하고 저장하고 다시 기다 기다리던 일을 실행하기 위해서

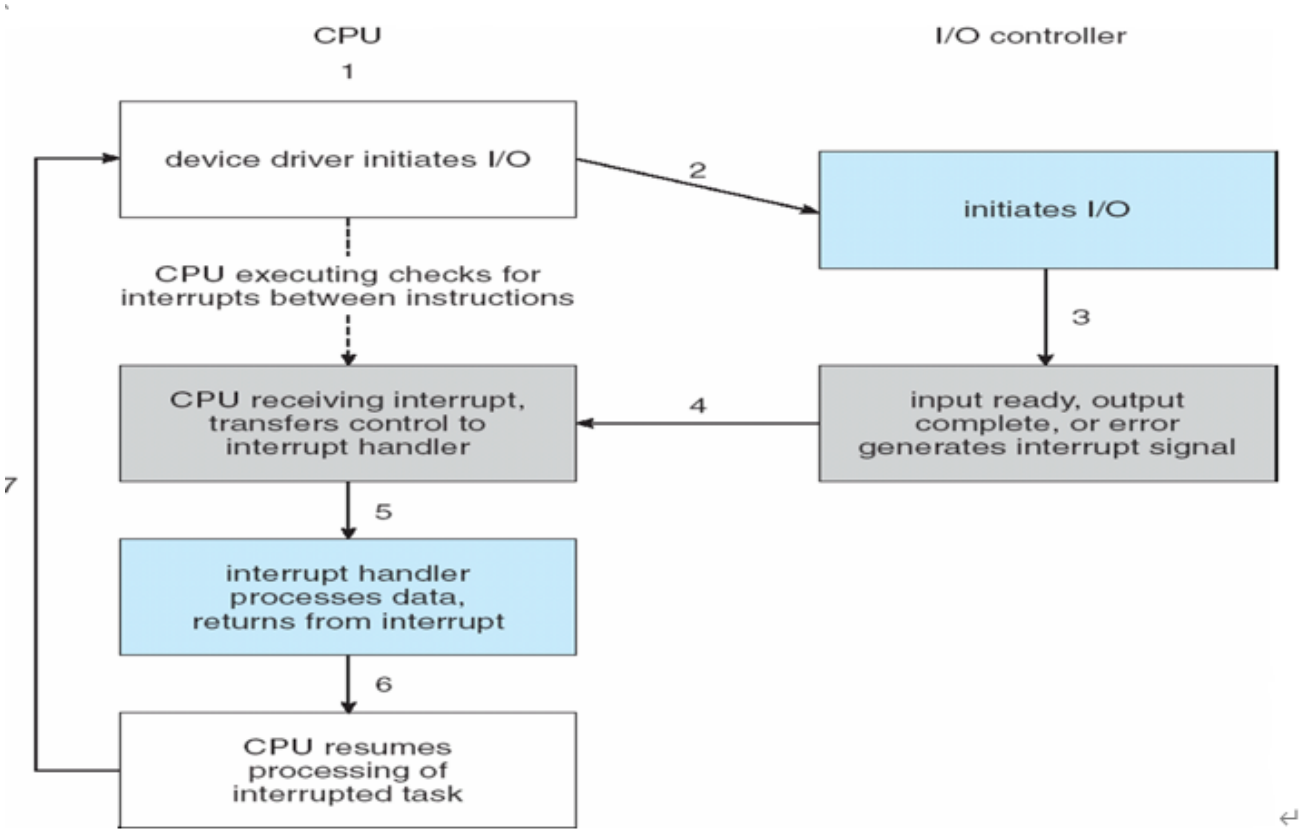
12번) 모든 Page Fault 과정이 종료되었음으로 그전에 수행하던일을 지속하기 위해서 PCB에서 저장되어 있던 정보를 가져와서 다시 진행하기 위해서

3. 필요한 페이지가 메모리에 적재된 후 10번 과정에서 페이지 테이블에 어떤 변화가 생기는지 서술하시오.

페이지 테이블의 valid-invalid bit의 변경

11. file open statement

프로그램에서 file open statement를 수행할 때 일어나는 interrupt-driven I/O의 과정을 나타낸 것이다.

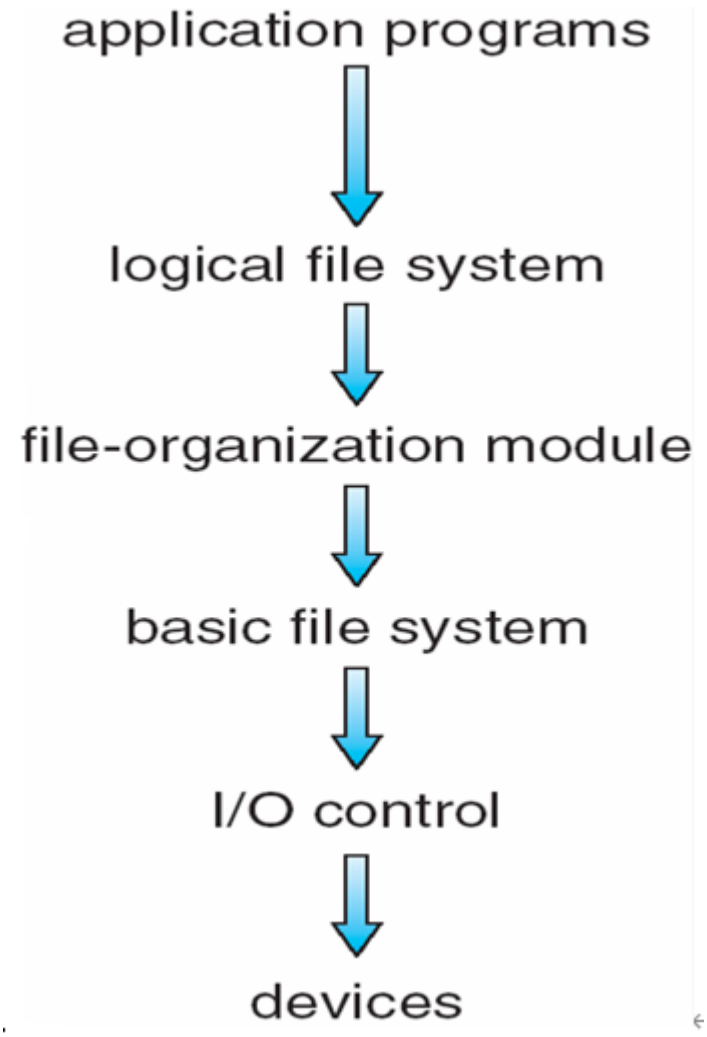


여러분의 프로그램이 현재 파일을 읽고 있는 과정이고 그다음 step은 연산을 하는 statement 가 있다고 가정하고, 현재의 step 과 그 다음 연산을 하는 step 사이에 일어나는 일을 상세히 설명하라. 여러분의 프로세서 번호가 P1번이고, 다른 사용자의 번호가 P2-P9번, 그리고 OS가 P0 이다. (각 스텝 번호에서 어느 프로세서가 어떤 수행을 하는지 상세 제시해야 함, 그렇지 않으면 0점 처리).

1. 커널에게 fOpen()통해 해당 파일 디스크럽터(포인터)를 요청
2. device driver에게 해당 주소를 전달 후, Cpu는 나머지 프로세스 excute(P2~P9)
3. I/O작업 종료
4. CPU에게 인터럽트를 보내 PO의 작업을 요청
5. OS가 해당 작업을 이어 받음
6. driver의 작업이 종료되어 fileOpen 준비가 끝나 컨택스트 스위치가 발생
7. P1에 대해서 실행하게 된다.

12. 파일이 실제 디스크에 저장되는 과정

(0.8+0.8+0.8+0.8+0.8 = 4점) 여러분의 파일이 실제 디스크에 저장되는 과정을 아래의 그림을 참조해서 상세히 설명하라. applicatio programs 부터 devices 까지 각 모듈 간의 맵핑 방법을 제시해야 하고 그렇지 않으면 0점 처리.



1. application programs: 해당 파일을 저장할 때 name을 넘긴다.
2. logical file system: 해당 파일의 logical 주소를 file-organization에 넘겨준다.
3. file-organization module: logical 주소를 physical하게 바꿔주는 역할을 합니다. (base 레지스터, limit 레지스터)
4. basic file system: 해당 주소를 I/O control에게 블락의 취를 전달한다.
5. I/O Controller: devices가 disk에서 몇번 실린더, 트랙, 섹터에 상세한 주소로 커맨드를 바꿔주는 역할을 한다. 이후 저장 작업을 수행

13. disk read

10, 22, 20, 2, 40, 6, 38 실린더와 같은 디스크 요청들이 Disk driver로부터 들어왔다. 실린더 1을 움직이는데 걸리는 탐색시간은 6msec이다. 다음 알고리즘들을 사용하면 얼마나 많은 탐색시간이 필요한가? 모든 경우에 arm은 cylinder20에서 시작한다고 가정하자

1. First Come, First Service

먼저 들어온 것 부터 처리

20에서 10으로 (+ 10) 10에서 22로(+ 12) 22에서 20(+ 2) 20에서 2(+ 18) 2에서 40(+ 38) 40에서 6(+ 34) 6에서 38(+ 32)

10 + 12 + 2 + 18 + 38 + 34 + 32 = 146 * 6sec

2. Elevator(SCAN) algorithm (initially moved upward)

왼쪽을 먼저 찍고 이동

78나옴

14. time sharing

현재 컴퓨터 내에서 A 프로그램과 B 프로그램이 동시(time sharing)에 수행되고 있다고 가정하자. A 프로그램이 Data read instruction (I/O instruction) 을 수행하고 있고 B 프로그램이 수학적 계산 프로그램 (CPU instruction) 을 수행하고 있는 경우 OPTIMAL 한 Operating System 이 하는 일을 I/O 관련 사항 (device, controller, driver)을 포함하여 매우 상세히 설명하라.

A프로그램이 먼저 실행된다면 read가 발생되고 이를 O/S에 Open을 요청하게 된다(시스템콜, 인터럽트)

그렇다면 Cpu는 해당 요청시간동안 기다리는 것이 아닌 time sharing을 통해 OS가(CPU) B프로그램을 수행하도록 하는 것이다.

A프로그램의 read준비가 완료되었다면 인터럽트를 발생시켜 B프로그램의 정보를 PCB에 저장하고 A프로그램에 대한 수행한다. (인터럽트)

여러 프로세스를 동시에 처리하는 것 처럼 보이게 된다.

15

아래의 instruction 에서 address binding 은 compile time, load time, 그리고 execution time 등 3가지 경우에 발생할 수 있다. 세 가지가 발생할 때의 기계어 코드의 내용 변환 (if any) 을 쓰고, 이 세 가지중 가장 효율적이라고 생각되는 방법과 가장 비효율적이라고 생각되는 방법을 한 가지씩 선택하고 그 이유를 간략히 설명하라.

Load R7, abc : 메모리 abc 의 내용을 레지스터 7번에 저장

1010 0111 XXXXXXXX

(1010 : load, 0111 : register 7, XXXXXXXX : the rest 8bits are Memory address)

0X10100000

가장 비효율적인 것은 compile time으로 컴파일 타임에 메모리를 할당하기 때문에(변수 할당) Logical주소가 아닌 Physical주소로써 사용을 해서 같아져버리는 것입니다.

반면에, 효율적인 방식은 실행이 될 때 주소를 바꾸어주는 방식으로써 예를 들어 abc의 내용이 8천 주소의 번지에 있다면 8천을 그대로 쓰는 것이 아닌 load가 되었을때의 시작점이 10000번지에서 시작을 한다면 실제로 abc 내용은 시작을 더해준 18000으로 찾아가는 방식입니다. 이러한 것은 MMU 하드웨어적으로 처리되기에 효율적이고 logical 주소를 사용하기에 더 편리합니다.

++ 실행시 마다 주소를 다시 할당한다.

16

다음의 서로 다른 성격의 프로그램이 있다(below there are two different programs)

(1) 실시간으로 주어진 수학식을 계산하는 프로그램(A program that calculates the math equation in real time)

(2) 월별로 부서별 직원 봉급을 계산하는 프로그램(A program that calculates the employee salary for each departments monthly)

아래의 2단계는 운영체제의 발전과정을 나타낸다.(below two steps define the process of O/S development)

(i). No multiprogramming O/S

(ii). Multiprogramming support O/S

각각의 단계에서 운영체제가 제공해야 할 기능들(프로세서, 메모리, I/O)을 구체적으로 기술하고 (i)->(ii)로 upgrade 할 때 필수적인 것을 거론하라. (두 개의 서로 다른 프로그램이 수행되어야 한다는 점을 상기하기 바람) (describe concretely the function about processor, memory, I/O which is provided by O/S and essential factor to be upgraded in initial O/S. Remind that above two different programs should be executed.)

(i) No Multiprogramming O/S

A. Processor management

B. Memory management

C. File management

D. I/O management

(ii) Multiprogramming support O/S (many program execute at the same time)

A. 전 운영체제에서 필수적으로 upgrade 되어야 할 것(essential upgrading factor in initial O/S)

B. Processor management

C. Memory management

D. File Management

E. I/O management