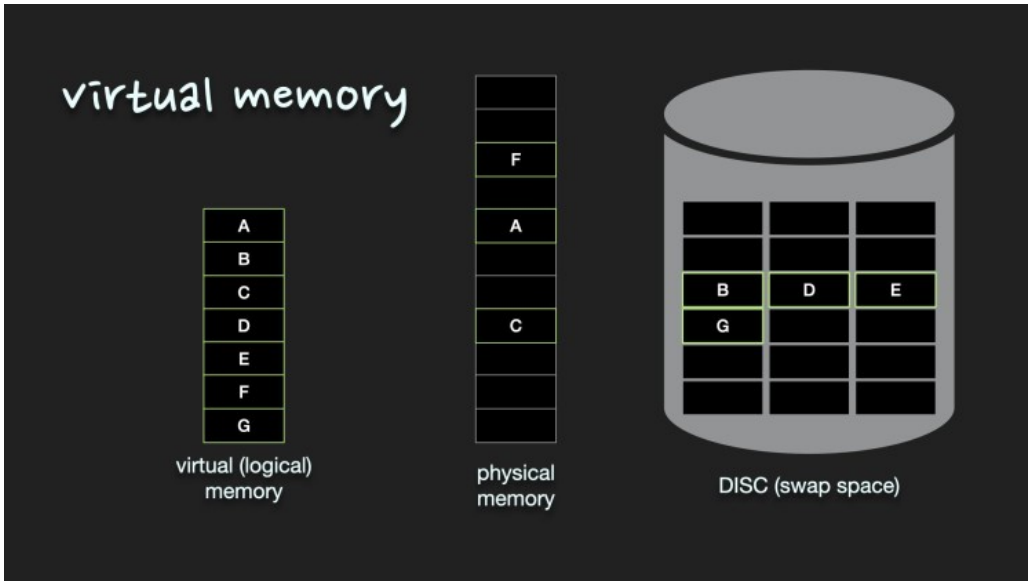
● 가상 메모리(virtual memory)에 대해 설명해라

▶ 가상 메모리란, process 전체가 메모리에 올라오지 않더라도 실행이 가능하도록 하는 기법입니다. 가상 메모리 기법을 통해서 사용자 프로그램이 물리적 메모리보다 커져도 실행이 가능하다는 장점이 있습니다. 가상 메모리는 실제의 물리 메모리 개념과 개발자 입장의 가상 논리 메모리 개념을 분리한 것이기 때문에, 개발자가 메모리 크기에 관련한 문제를 염려할 필요 없이 쉽게 프로그램을 작성할 수 있게 해줍니다. 운영체제는 가상 메모리 기법을 통해서 프로그램의 논리적 주소 영역에서 필요한 부분만 물리적 메모리 RAM에 적재하고, 직접적으로 필요하지 않은 메모리 공간은 디스크(Swap 영역)에 저장하게 됩니다.



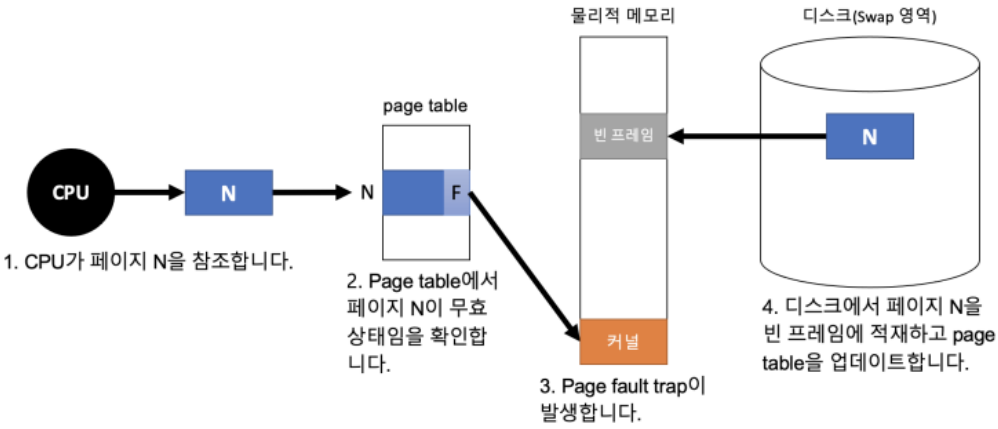
● 요구 페이징(demand paging)

▶ 요구 페이징(demand paging) 기법은 특정 page에 대해 cpu의 요청이 들어왔을 때만 해당 page를 메모리에 적재합니다. 당장 사용될 주소 공간만 page단위로 메모리에 적재하기 때문에 메모리 사용량이 감소하고, 프로세스 전체를 메모리에 적재할 필요가 없기 때문에 입출력 오버헤드도 감소하는 장점이 있습니다. 요구 페이징 기법에서는 유효, 무효 비트(valid/invalid bit)를 두어서 page table에 각 page가 메모리에 존재하는지 표시하게 됩니다. bit가 유효하면 메모리에 있는 것을 의미하고, 무효하면 메모리에 없음을 의미합니다.

● Page fault

▶ CPU가 무효 비트(invalid bit)로 표시된 page에 엑세스하는 상황을 page fault라고 합니다. CPU가 무효 page에 접근하면 주소 변환을 담당하는 하드웨어인 MMU(Memory Management Unit - 가상 주소를 물리 메모리 주소로 변환해주는 하드웨어 장치)가 page fault trap을 발생시키게 되고, 다음과 같은 순서로 page fault를 처리하게 됩니다:

1. CPU가 페이지 N을 참조합니다.
2. Page table에서 페이지 N이 무효 상태임을 확인합니다.
3. MMU에서 Page fault trap이 발생시킵니다.
4. 디스크에서 페이지 N을 물리적 메모리의 빈 프레임에 적재하고 page table을 invalid에서 valid로 업데이트합니다.



● page 교체 알고리즘(replacement algorithm)

▶ page fault가 발생하면, 요청된 page를 디스크에서 메모리로 가져옵니다. 이 때, 물리적 메모리에 공간이 부족할 수 있습니다. 그럴 경우에는 메모리에 올라와 있는 다든 page를 디스크로 옮겨서 메모리 공간을 확보해야 합니다. 이것을 페이지 교체(page replacement)라고 하고, 어떤 page를 교체할 것이냐를 결정하는 알고리즘이 page교체 알고리즘(replacement algorithm)입니다. 교체 알고리즘은 최대한 page fault가 적게 일어나도록 도와줘야 합니다. 따라서 앞으로 참조될 가능성이 적은 page를 선택해서 교체하는 것이 성능을 향상시키는 방법입니다. 페이지 교체 알고리즘에는 FIFO, 최적 페이지 교체, LRU, LFU가 있습니다.

페이지 교체 알고리즘 (page replacement algorithm)	설명
FIFO(First In First Out)	메모리에 올라온지 가장 오래된 page를 교체합니다.
최적 페이지 교체	앞으로 가장 오랫동안 사용되지 않을 page를 찾아 교체합니다. 실제구현은 어렵습니다.
LRU(Least Recently Used)	가장 오랫동안 사용되지 않은(즉, 가장 오래 전에 참조가 된) page를 교체합니다.
LFU(Least Frequently Used)	물리적 메모리 내에 올라와 있는 page 중에서 지금까지의 참조 횟수가 가장 적은 page를 교체합니다. 비용 대비 성능이 좋지 않아 잘 쓰이진 않습니다.

Resources: inflearn 개발남 노씨