운영체제 5장 과제

1731017 김민정

4. 스레드가 없고 프로세스만 있는 운영체제에서 프로세스를 다룰 때 프로세스의 생성 시간, 컨텍스트 스위칭, 프로세스 사이의 통신 3가지 점도의 관점에서 어떤 문제가 있는지 설명하라.

첫째, 프로세스 생성 시간의 오버헤드가 크다.

모든 프로세스들이 독립적인 메모리 공간을 가지므로, 생성되는 자식 프로세스에게 부모 프로세스의 주소 공간과 별도의 메모리가 할당되고, 이곳에 부모프로세스가 복사된다. 따라서 프로세스 생성에 많은 시간이 걸린다. 운영체제 또한 새로 생성되는 프로세스를 위해 PCB, 페이지 테이블 등 프로세스 관리를 위한 구조체를 생성하는데 많은 시간을 소모한다.

둘째, 프로세스 컨텍스트 스위칭의 비효율성이 크다.

현재 실행 중인 프로세스를 중단시키고 다른 프로세스를 실행시키는 컨텍스트 스위칭에 따른 시간적 공간적 오버헤드가 크다. 구체적으로는 프로세스의 논리적 주소와 물리 메모리 주소의 페이지 테이블을 새로 실행될 프로세스의 페이지 테이블로 교체하는 시간, CPU와 PCB 간 메모리 저장 및 로딩 시간, 캐시 메모리 초기화 및 로딩에 걸리는 오버헤드가 있다.

셋째, 프로세스 사이의 통신 어려움이 있다.

프로세스들은 상호 독립된 메모리를 가지고 있어, 프로세스가 다른 프로세스의 메모리를 전혀 접근할 수 없다. 따라서 프로세스들은 공유메모리, 신호, 파이프, 파일, 소켓, 메시지 큐, 세마포어, 메모리 맵 파일 등을 통해 데이터를 주고 받는다. 그런데 이 기능들은 거의 모두 커널의 직접적인 지원을 받을 수 밖에 없으며, 운영체제에 따른 호환성이 부족하고, 개념적으로 복잡하며, 코딩도 어렵고 실행속도도 느린 어려움이 있다.

2. 프로세스 대신 스레드를 실행 단위로 할 때, 좋은 점은 무엇인가 5가지 정도로 나열하라.

첫째, 작업의 내용들을 하나의 프로세스에 스레드별로 정의가 가능하기 때문에 사용자 입장에서는 작업의 처리가 빨라져 반응성이 향상되고 프로세스 내부에 여러 작업을 정의할 수 있으므로 시스템 전체의 작업 처리량이 증가한다.

둘째, 프로세스 간 문맥교환보다 스레드 간 문맥교환의 속도가 빠르기 때문에 CPU의 부담이 경감된다.

셋째, 멀티 코어 프로세서 시스템에서는 하나의 프로세스에 멀티 스레드가 병렬 처리가 될 수 있으므로, 멀티 코어 프로세서 시스템에 효율적이다.

넷째, 생성과 소멸에 따른 오버헤드가 적다.

다섯째, 프로세스 내부 스레드들은 주소공간을 공유하기 때문에 상호 통신이 프로세스간 통신에 비해 간단하다.

3. 스레드란 무엇인가? 스레드를 정의하라.

스레드는 응용 프로그램 개발자에게는 태스크를 만드는 단위이며, 운영체제에게는 실행 단위이다. 생성, 실행, 중단, 종료등 프로세스 생명주기와 동일한 형태의 생명을 가진 독립된 단위이다. 스레드는 코드와 데이터를 가진 실체로서, 그 실행이 운영체제나 스레드 라이브러리에 신고되어야 한다. 그러면 운영체제나 스레드라이브러리는 함수의 시작 주소를 스케쥴링 목록에 기록하고 독립적으로 함수의 시작 주소부터 실행을 시작시킨다. 그리고 스레드마다 TCB 구조체를 만들어서 TCB 리스트를 유지 관리한다.

4. 스레드의 컨텍스트란 어떤 정보를 말하는가? 그리고 어디에 저장되는가?

스레드 컨텍스트는 CPU가 스레드를 실행하고 있을 때의 CPU 레지스터 값들이다.
CS 레지스터는 스레드의 코드 영역 주소를, PC 레지스터는 현재 실행 중인 명령어 주소,
DS 레지스터는 스레드의 데이터 영역 주소, SS 레지스터는 스레드의 스택 영역 주소,
SP 레지스터는 스레드 스택의 탑 주소를 저장한다. 이외에도, 다양한 레지스터들이
있으며 TCB (Thread Control Block)에 저장된다.

5. 컨텍스트 스위칭 시간은 어떤 시간의 합으로 나타낼 수 있는가? 그리고 그 시간은
CPU와 운영체제에 따라 다를 수 있지만, 어느 정도 수준인가?

컨텍스트 스위칭 시간은 사용자 모드에서 커널모드로 전환하는 시간, 커널모드에서
사용자 모드로 전환하는 시간, 주소 공간을 교체하는 시간, CPU에서 PCB의 값을
레지스터에 저장 및 로딩하는 시간, 그 외 스케줄링에 들어가는 시간을 합한 값이다.
이는 운영체제에 따라 다르지만 보통 수백 ns 에서 몇 MS 사이의 시간이
소요된다.

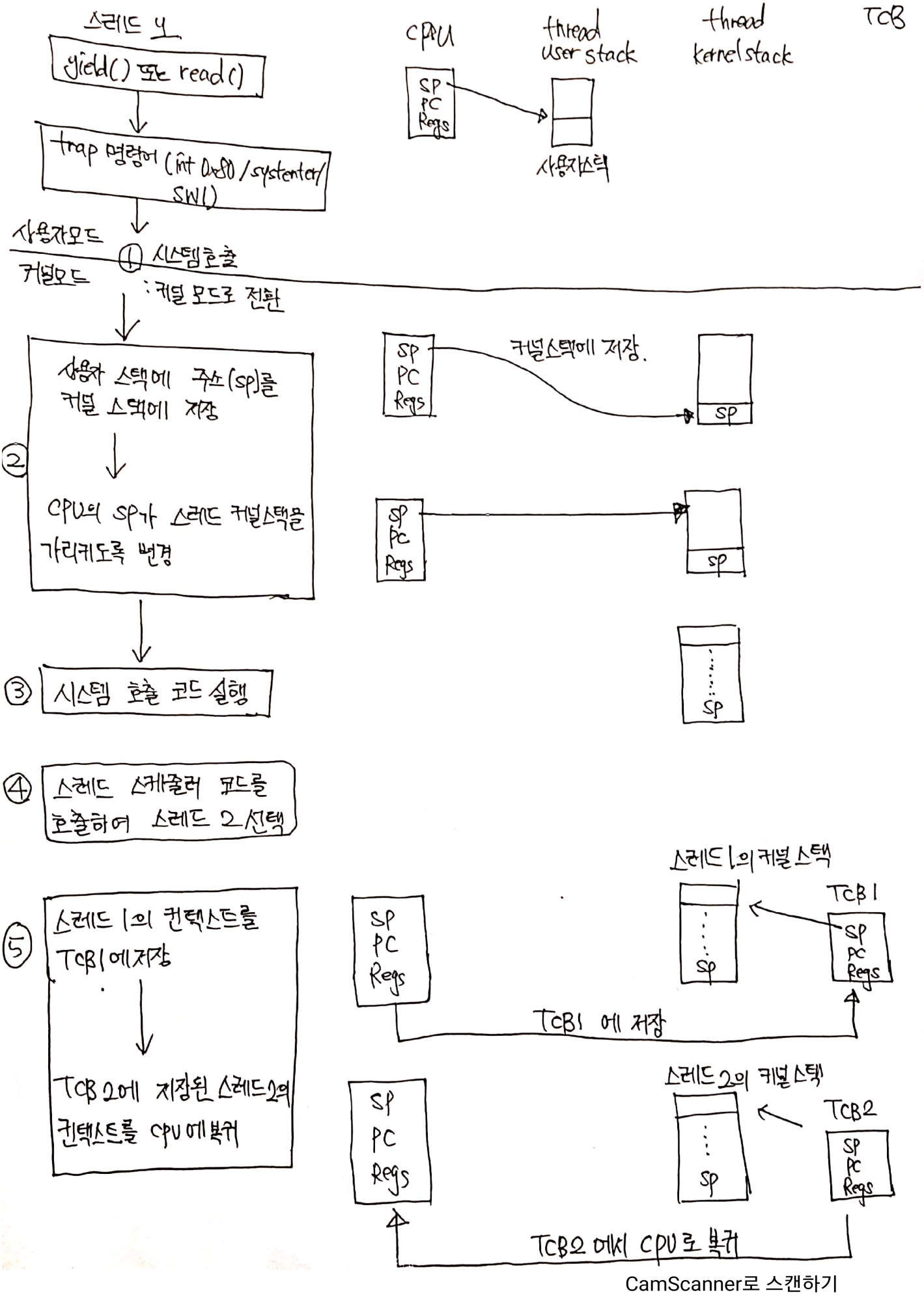
6. 슬라이드에 주어진 4개의 자식 스레드를 생성하여 1에서 4000까지의 합을
구하는 프로그램을 작성하라.

첨부 사진 올렸습니다.

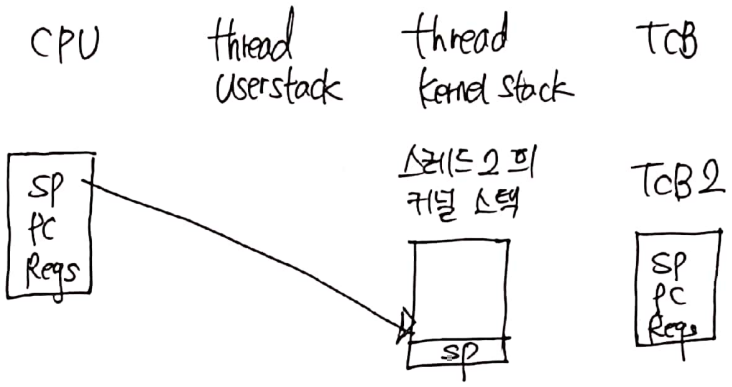
7. 하나의 스레드가 실행되기 위해 주어지는 주소공간은 총 6개의 영역으로 나누어진다.
이 6가지가 무엇인지 설명하라. 이 중에서 다른 스레드와 공유하는 공간은 무엇인가?

스레드와 6가지 주소 공간은 다음과 같다.
첫째, 스레드 코드 공간. 둘째, 스레드 전용 전역변수 공간 (Thread Local Storage).
셋째, 스레드 전용 스택 공간 넷째, 데이터 공간. 다섯째, 힙 공간. 여섯째,
스레드가 커널에 진입하였을 때 할당되는 스레드별 커널 스택.
이 중 다른 스레드와 공유하는 공간은 데이터 공간과 힙 공간이다.

8. 스레드 1 이 시스템 호출을 수행 중에 I/O 작업으로 인해 blocked 되고, 스레드 2가 선택되어 스위칭 되는 과정을 그림을 그리고 자세히 설명하라.

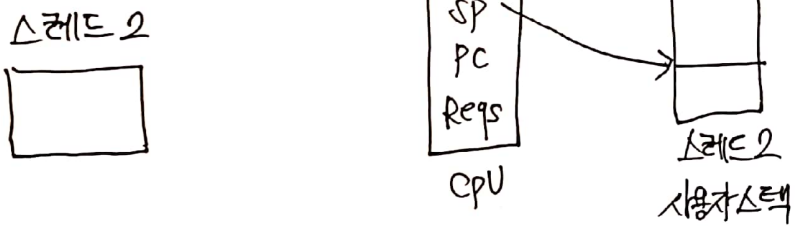


⑥ 스레드 2의 남은 작업 완료



⑦ 스레드 2의 커널 스택에 저장된 사용자 SP를 CPU에 복귀

⑧ 시스템 호출 코드를 끝내고 사용자 모드로 전환하는 기계어 명령을 통해 사용자 모드로 전환



9. 프로세스 1에 속한 스레드에서 프로세스 2가 속한 스레드로 컨텍스트 스위칭이 일어날 때, 같은 프로세스에 속한 스레드 스위칭의 경우보다 추가적으로 더 필요한 작업은 무엇인가?

서로 다른 프로세스에 속한 스레드 간의 컨텍스트 스위칭이 되는 경우는 같은 프로세스의 스레드들 간 컨텍스트 스위칭의 경우보다 다음 2개의 오버헤드가 더 추가된다.

첫째, 메모리 관련 오버헤드가 추가된다.
프로세스의 논리주소와 물리주소를 매핑하는 MMU(Memory Management Unit)에 현재 프로세스의 맵 테이블을 제거하고, 새로운 맵 테이블을 로딩해야 한다.
또한 이 과정에서 새로운 프로세스의 코드나 데이터가 메모리에 없고 하드 디스크에 있는 경우 이를 메모리에 옮겨와야하는 과정(페이지 폴트)를 처리하는 오버헤드도 있다.
또, CPU 내에 현재 프로세스의 TLB 버퍼를 모두 비우고(flush) CPU에 새 프로세스의 TLB를 채우는 작업이 필요하며, 이 시간은 결코 적지 않다.
둘째, 추가적인 캐시 오버헤드가 추가된다. CPU 캐시를 새 프로세스의 데이터로 채워야하는 경우도 발생한다.

10. 스레드 라이브러리에 의해 스케줄되는 스레드는 사용자 레벨 스레드 인가 커널 레벨 스레드인가? 커널 레벨 스레드로 다루기 위해 커널이나 사용자 공간에 어떤 정보가 유지되어야 하는가?

스레드 라이브러리에 의해 스케줄되는 스레드는 사용자 레벨 스레드이다.
커널 레벨 스레드의 코드와 데이터의 위치는 사용자 영역이나 커널 영역 어느 하나에 고정되어 있지 않은데, 이때 커널 레벨 스레드로 다루기 위해서는 TCB가 존재해야 한다.

11. 커널 레벨 스레드의 정의는 무엇인가?

커널 레벨 스레드란 스레드에 관한 정보 TCB를 커널이 가지고 있고, 커널에 의해 스케줄되는 스레드이다. 사용자가 시스템 호출을 통해 스레드를 만들면 TCB가 커널 공간에 만들어지고, 커널이 TCB를 스케줄링 하기 때문에 커널 레벨 스레드가 된다. 그리고 스레드 동기화 등 스레드의 운용에 관한 모든 기능은 커널에 의해 제공되므로 커널 스레드들은 시스템 호출을 통해서만 이들 기능을 이용할 수 있다.
특별히 응용 프로그램의 시스템 호출을 통해 스레드를 만든 것이 아니고, 커널 코드가 직접 스레드를 만든 경우도 있는데 이는 순수 커널 레벨 스레드라고 한다. 이 스레드는 처음부터 커널 모드에서 실행되므로 하드웨어를 제어하는 특권 명령어를 포함하여 모든 CPU 명령들을 실행할 수 있다.

12. 커널 레벨 스레드는 사용자 공간에서 코드를 실행하는가? 아니면 커널공간에서만 실행되는가?

커널 레벨 스레드는 커널에 의해 스케줄링 된다는 의미일 뿐, 커널 레벨 스레드의 코드와 데이터가 커널에 있어야 한다는 것은 아니다. 따라서 커널 레벨 스레드의 주소공간은 사용자 공간에 있을 수도 있고, 커널 공간에 있을 수도 있다.

13. 사용자 레벨 스레드가 커널 레벨 스레드보다 좋은 점은 무엇인가?

커널 레벨 스레드는 컨텍스트 스위칭에 많은 시간이 소요되어 시스템의 성능을 떨어뜨리는 원인이 되었다. 이에 따라 커널의 도움을 받지 않고도 스레드를 만들고 스케줄링할 수 있는 스레드 라이브러리가 개발되었고 사용자 레벨 스레드가 도입되었다. 사용자 레벨 스레드는 컨텍스트 스위칭에 작은 시간이 소요되며, 스레드 라이브러리의 개발로 사용자가 쉽게 멀티 프로그램을 작성할 수 있게 되었다. 그리고 스레드를 지원하지 않는 운영체제 에서도 멀티 스레드 응용 프로그램을 작성할 수 있어 이식성이 높다.

14. 사용자 레벨 스레드와 커널 레벨 스레드를 매핑하는 N:1, 1:1, N:M에 대해 간단히 설명하라.

먼저 N:1은 모든 사용자 스레드를 하나의 커널 스레드로 매핑하는 방법이다. 하나의 코어에서 하나의 커널 레벨 스레드를 다루며, 이 스레드에서 멀티 사용자 레벨 스레드가 돌아가는 형태이다. 장점으로는 스레드의 생성, 스케줄링, 동기화 등 모든 것이 커널 진입 없이 스레드 라이브러리에 의해 사용자 공간에서 이루어지기 때문에 프로그램 실행 속도가 전반적으로 매우 빠르다는 점이 있다. 단점은 다음과 같다.
첫째, 응용 프로그램의 여러 스레드들 중 하나에게만 CPU 코어가 할당되어 있기 때문에 멀티스레드의 병렬성(parallelism)을 얻을 수 없다.
둘째, 하나의 TCB에 모든 스레드가 매핑되어 있어서 한 사용자 레벨 스레드가 시스템 호출이나 입출력 요청으로 인하여 Blocked 상태가 되면 스레드 전체를 사용할 수 없게 되어 응용프로그램 실행이 중단된다.

다음 1:1은 사용자 레벨 스레드 하나당 커널에 의해 스케줄 가능한 엔터티(TCB) 하나를 연계시키는 방법이다. 커널이 스케줄링을 통해 TCB를 하나 선택하면, CPU에게 이 TCB에 연결된 사용자 레벨 스레드의 코드를 실행하도록 하는 기법이다.
사용자 스레드 생성 시 스레드 라이브러리는 새로운 사용자 스레드의 생성을 인지할 뿐 아니라 시스템 호출을 통해 커널에 커널 레벨 스레드를 생성해 줄 것을 요청한다.
장점은 첫째, 개념이 단순하여 구현하기가 쉽다. 둘째, 사용자 레벨 스레드들이 여러개의 CPU 코어에서 동시에 실행 되으로 높은 병렬성을 확보할 수 있다.
단점은 다음과 같다. 사용자 레벨 스레드 개수만큼 커널 레벨 스레드가 생기므로 커널에 많은 구조체가 생성되며, 사용자 레벨 스레드의 생성과 소멸 등 운용되는 각 경우마다 커널 모드에 진입하기 때문에 커널 모드 진입 횟수가 많아진다. 모든 스레드를 커널이 스케줄하기 때문에 스케줄에 따른 시간 부담이 많다. 또한 사용자 레벨 스레드 사이의 컨텍스트 스위칭이 일어날 경우에도 커널 레벨 스레드 컨텍스트 스위칭이 되기 때문에 운영체제의 부담이 매우 크다.

N:M 방식은 N개의 스레드를 M개의 커널 엔티티에 연계시키는 방법이다.
이는 1:1 과 N:1 매핑의 단점을 보완하고자 고안된 방식이다.
스레드 라이브러리가 사용자 레벨 스레드를 생성한 후 시스템 호출을 사용하여 커널 엔티티를 만든다. 여기서 커널 엔티티의 수는 커널에 의해 제한되며, 어떤 사용자 레벨 스레드를 어떤 커널 엔티티에 연계시킬 지 결정하는 것은 스레드 라이브러리이다.
장점은 1:1 매핑에 비해 커널 엔티티의 개수가 많지 않아 커널의 부담이 적다는 것이다.
단점은 복잡하여 구현하기 어려운 점이다. 따라서 현대 운영체제에서는 거의 사용하지 않는다.

15. 현재 이 3가지 모델 중에서 가장 많이 사용되는 것은? 그리고 그 이유는 무엇인가?

현재 가장 많이 사용되는것은 1:1 모델이다. 1:1 매핑은 개념이 단순하여 구현하기 쉽고, 멀티 코어 CPU를 사용할 때 높은 병렬성을 얻을 수 있다. 따라서 현재 윈도우를 포함하여 대부분의 운영체제에서 지원하고 있다.