

프로세스 vs 스레드

25.08.14 안효성

프로세스(Process)

프로세스는 **실행 중인 프로그램**을 의미한다. 프로세스는 운영체제에 의해 관리되며, 독립적으로 실행되고 자원을 할당 받을 수 있는 단위이다. 운영체제는 프로세스들에게 적절히 자원들을 분배하여 여러가지 작업을 수행할 수 있게 한다.

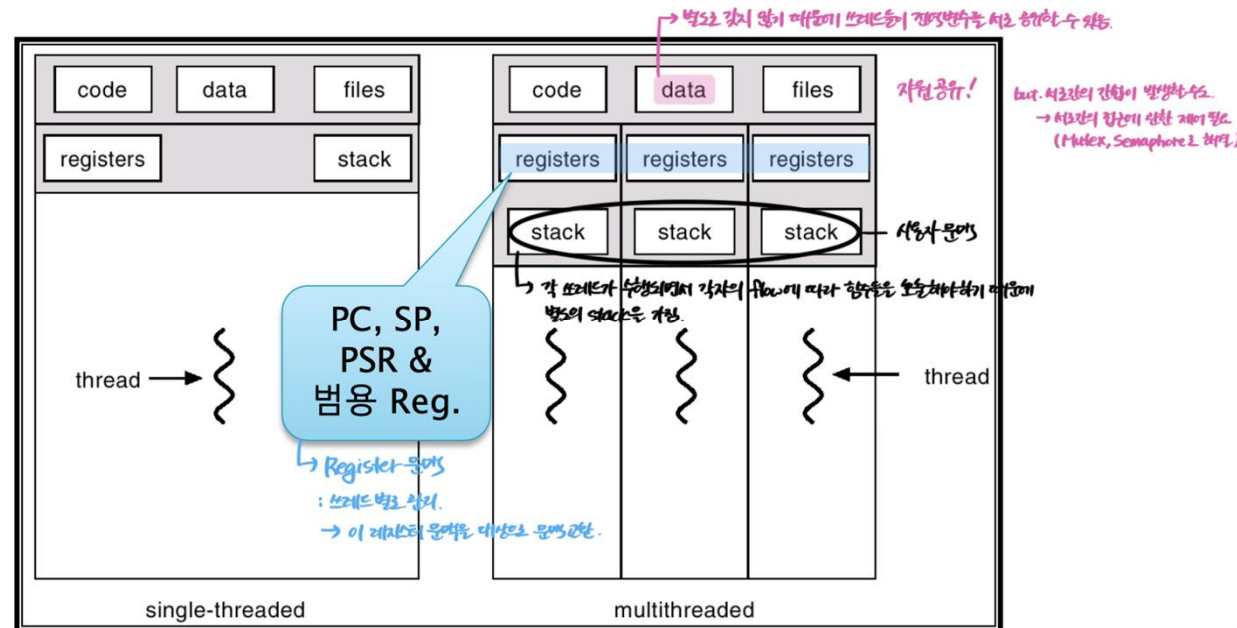
메모리에 올려진 데이터를 프로세스라고 함

[프로세스: Excel]

- 메모리 (코드, 데이터, 힙, 열린 파일) ← 모든 스레드가 공유
- 스레드 1 (UI 처리)
- 스레드 2 (계산 처리)
- 스레드 3 (자동 저장)

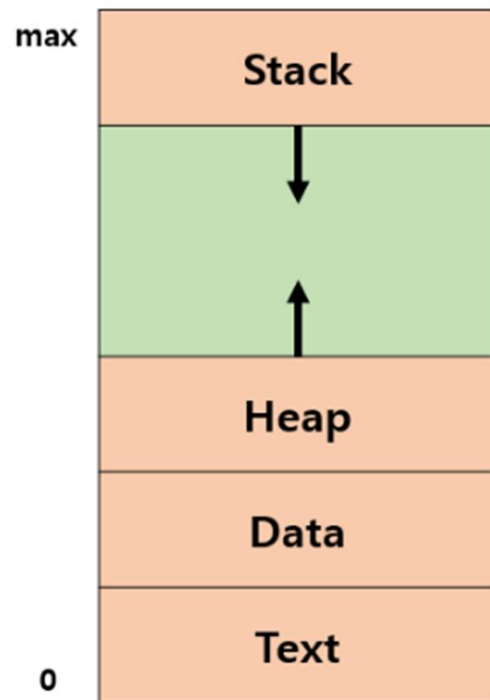
스레드(Thread)

- 프로세스 안에서 실제로 명령을 실행하는 실행 단위
- 프로세스가 '집'이라면, 스레드는 그 집 안에서 '일하는 사람'
- 하나의 프로세스는 최소 1개의 스레드를 가지며(= 메인 스레드), 여러 스레드를 만들어 동시에 여러 작업을 병렬적으로 처리할 수 있음.



프로세스 구조

프로세스마다 고유한 가상 메모리 공간을 제공



지역변수, 매개변수, return 주소들을 저장

```
int n;  
scanf("%d", &n); // 사용자에게 입력 받을  
int* arr = malloc(n * sizeof(int)); // 크기를 런타임에 결정
```

동적으로 생성되는 데이터 구조나 객체들을 저장

전역 변수, 정적 변수, 상수 등을 저장

내가 작성한 코드가 저장되는 공간

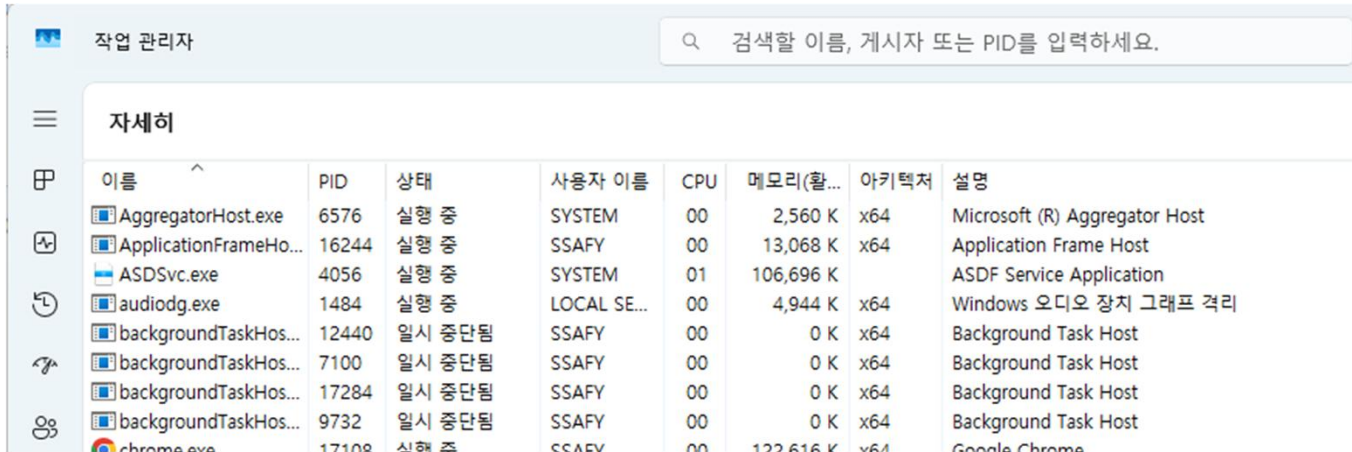
PCB(Process Control Block)

운영체제가 프로세스를 관리하기 위해 사용하는 데이터 구조

하드웨어와 소프트웨어의 접점(인터페이스) 대표적으로 Linux.
운영체제에서 Text editor같은 user application 빼면 커널이라함

프로세스들의 정보를 저장하는 공간 → 운영체제 **커널**의 데이터 영역에 생김

운영체제의 커널(Kernel) 또한 하나의 프로그램이므로 프로세스와 같이 정보를 저장할 수 있는 공간(stack, data, heap ...)이 생긴다. 이때, 커널의 데이터(Data) 영역에서는 각 프로세스의 상태, CPU 사용의 정보, 메모리 사용 정보 등 각종 자원을 관리하기 위해서 PCB라는 공간을 둔다.



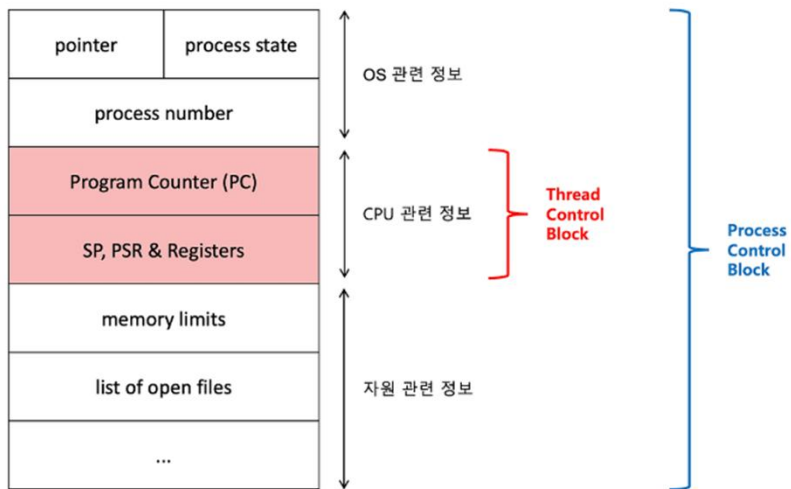
작업 관리자							
자세히							
이름	PID	상태	사용자 이름	CPU	메모리(활...	아키텍처	설명
AggregatorHost.exe	6576	실행 중	SYSTEM	00	2,560 K	x64	Microsoft (R) Aggregator Host
ApplicationFrameHo...	16244	실행 중	SSAFY	00	13,068 K	x64	Application Frame Host
ASDSvc.exe	4056	실행 중	SYSTEM	01	106,696 K		ASDF Service Application
audiodg.exe	1484	실행 중	LOCAL SE...	00	4,944 K	x64	Windows 오디오 장치 그래픽 격리
backgroundTaskHos...	12440	일시 중단됨	SSAFY	00	0 K	x64	Background Task Host
backgroundTaskHos...	7100	일시 중단됨	SSAFY	00	0 K	x64	Background Task Host
backgroundTaskHos...	17284	일시 중단됨	SSAFY	00	0 K	x64	Background Task Host
backgroundTaskHos...	9732	일시 중단됨	SSAFY	00	0 K	x64	Background Task Host
chrome.exe	17100	실행 중	SSAFY	00	122,616 K	x64	Google Chrome

프로세스들의 정보가 다 담겨있음=>스케줄링, 문맥 교환, 자원 관리, 상태 전이 모두 PCB에 의존

Thread는??

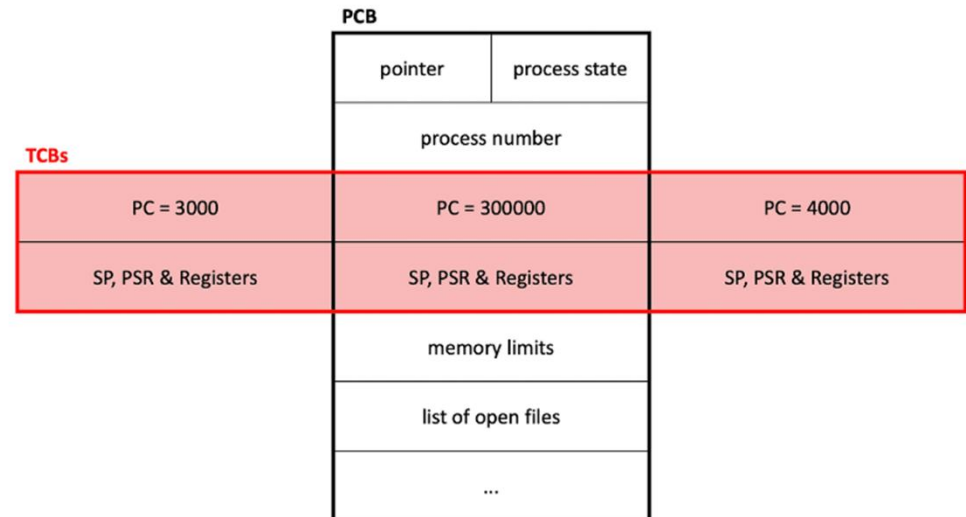
별도의 PCB를 가지진 않음 대신 TCB를 가짐

PCB 내 TCB를 수용하는 것으로 커널은 스레드를 실현시킨다.



스레드가 하나 생성될 때마다 PCB 내에서 TCB가 확장된다.

아래의 예시에서는 한 프로세스 내에서 2개의 스레드가 추가 생성되어 총 3개의 스레드가 독립적으로 실행되는 중이다.



스레드 종류

1. 사용자 스레드
2. 커널 스레드

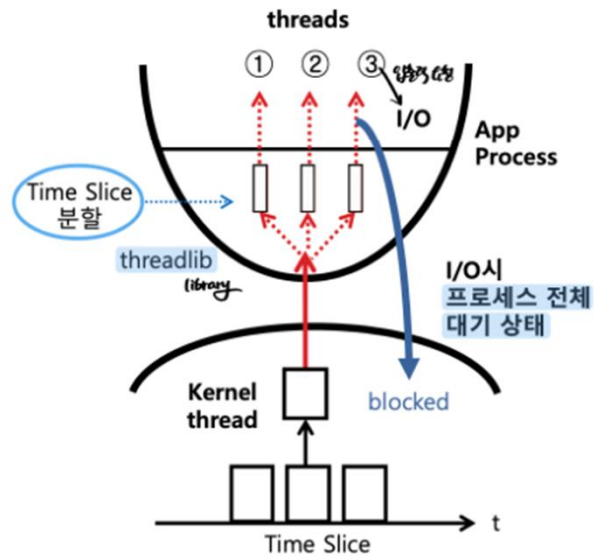
1. 사용자 스레드 (user thread)

응용 프로그램 내의 라이브러리에 의해서 구현 및 관리되는 스레드

커널 위에서 지원되기 때문에 커널은 사용자 스레드를 인식하지 못한다.

장점 : 생성과 관리가 빠르다.

단점 : 한 스레드가 봉쇄 시스템 호출을 통해 대기 상태가 되어(ex. 입출력 요청) blocking 된다면 다른 스레드도 함께 block 된다.



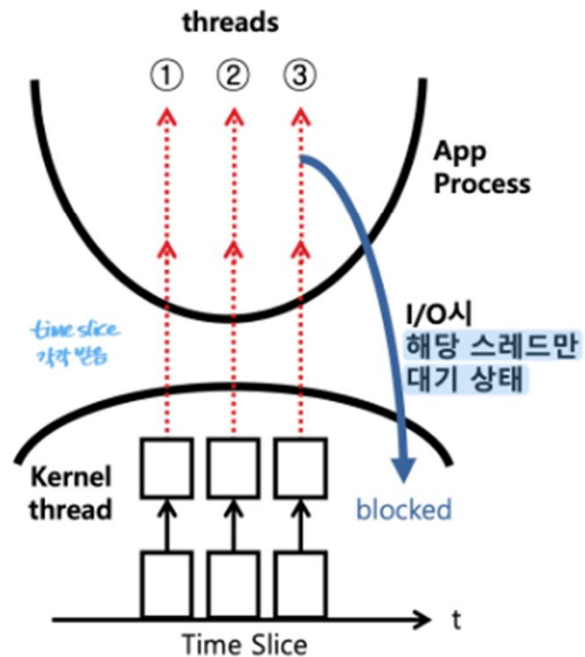
2. 커널 스레드 (kernel thread)

운영체제에 의해 직접 지원되고 관리되는 스레드

스레드의 생성, 스케줄링 등의 관리가 커널에서 이루어진다. (커널이 TCB를 만들어서 관리)

장점 : 한 스레드가 대기 상태로 들어가 blocking 되어도 다른 스레드는 실행을 계속할 수 있다.

단점 : 생성과 관리가 느리다.

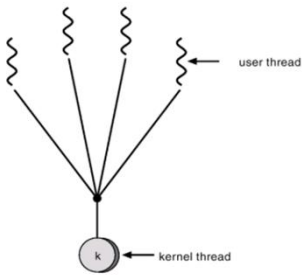


예시)
유튜브 영상은 버퍼링 걸려도 채팅은 칠 수 있는거

현대 대부분의 시스템들은 혼합형 모델로 두 가지 형태의 스레드를 모두 지원한다.

혼합방식

다대일 모델



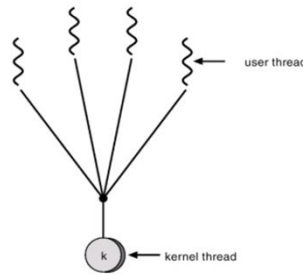
많은 사용자 스레드 - 하나의 커널 스레드

개발자가 원하는 만큼의 사용자 스레드를 생성할 수 있지만, 스레드의 관리가 사용자 공간에서 일어나기 때문에 하나의 스레드가 봉쇄된다면 전체 프로세스가 봉쇄된다.

또한, 한번에 하나의 스레드만이 커널에 접근할 수 있어, 다중 스레드가 다중 코어 시스템에서 **병렬로 실행될 수가 없다**.

→ 다대일 모델을 사용하는 시스템은 거의 없다.

일대일 모델



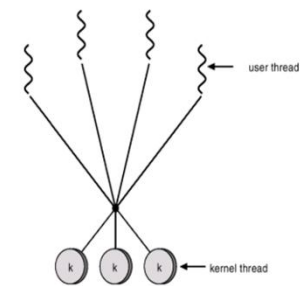
하나의 사용자 스레드 - 하나의 커널 스레드

하나의 스레드가 봉쇄되어도 다른 스레드가 실행될 수 있다.

다중 처리기에서 다중 스레드가 병렬로 수행되는 것을 허용하지만, 사용자 수준 스레드를 생성할 때 그에 따른 커널 스레드를 생성해야 하기 때문에 그에 따른 오버헤드가 발생한다.

→ 시스템에 의해 지원되는 스레드의 수를 제한

다대다 모델



여러 사용자 스레드 - 그보다 작거나 같은 수의 커널 스레드

다대다 모델은 다대일, 일대일 모델의 단점을 해결해 준다.

다대다 모델은 필요한 만큼 사용자 스레드를 생성할 수 있으며, 그의 개수에 상응하는 커널 스레드가 다중 처리기에서 병렬로 수행될 수 있다.

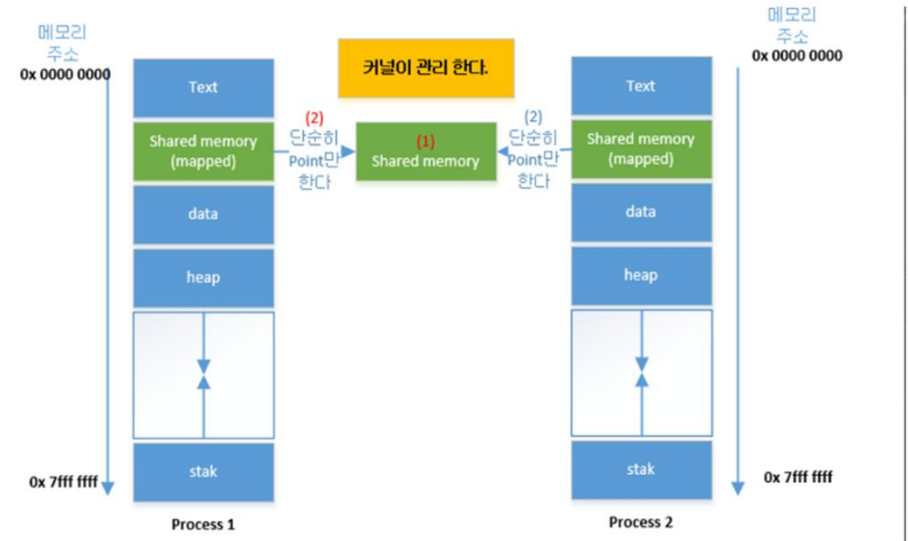
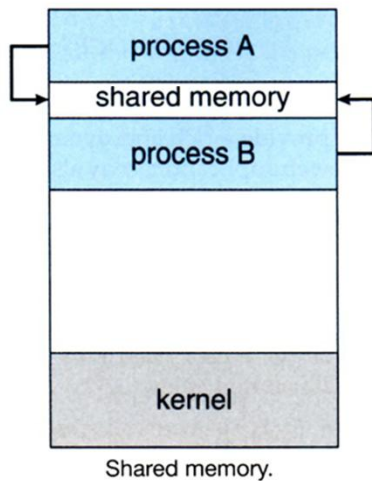
따라서 **한 스레드가 봉쇄형 시스템 호출을 발생시켜 대기 상태로 들어갔을 경우 커널이 다른 스레드의 수행을 스케줄 할 수 있다**.

1. 멀티 프로세스 Ex) 크롬이 부모 프로세스라면 여러 탭 키는건 자식 프로세스를 생성하는 것!

2. 프로세스들은 별도의 메모리공간에서 실행되는데 프로세스간 정보를 주고 받을 순 없을까?->IPC

따라서 프로세스는 통신할 수 있는 공간이 없어 통신을 위한 별도의 공간을 만들어주어야 한다.

1. 공유 메모리 (Shared Memory)



- 프로세스간 메모리 영역을 공유해서 사용할 수 있도록 지원하는 설비
- 프로세스가 공유 메모리 할당을 커널에 요청 시 커널은 해당 프로세스에 메모리 공간을 할당

> 공유 메모리가 각 프로세스에게 첨부하는 방식으로 작동
> 각 프로세스가 메모리 영역에 첨부됨

- 프로세스 간 Read, Write를 모두 필요로 할 때 사용
- 대량의 정보를 다수의 프로세스에게 배포 가능
- 중계자 없이 곧바로 메모리에 접근
- 따라서 IPC 중에서 가장 빠르다

기술면접 질문

1. 프로세스간 통신 방식(IPC)이 뭔지 설명해 주세요.

<https://dar0m.tistory.com/233>

2. 스레드 동기화 방식은 무엇일까요?

<https://jhtop93.tistory.com/40>

IPC 종류	PIPE	Named PIPE	Message Queue	Shared Memory	Memory Map	Socket
사용 시기	부모 자식 간 단 방향 통신	다른 프로세스와 단 방향 통신	다른 프로세스와 단 방향 통신	다른 프로세스와 양 방향 통신	다른 프로세스와 양 방향 통신	다른 시스템 간 양 방향 통신

Mutex / Semaphore / Monitor

프로세스 vs 스레드 모든것

<https://inpa.tistory.com/entry/%F0%9F%91%A9%E2%80%8D%F0%9F%92%BB-%ED%94%84%EB%A1%9C%EC%84%B8%EC%8A%A4-%E2%9A%94%EF%B8%8F-%EC%93%B0%EB%A0%88%EB%93%9C-%EC%B0%A8%EC%9D%B4>

IPC 종류

<https://velog.io/@threeone/%ED%94%84%EB%A1%9C%EC%84%B8%EC%8A%A4-%EA%B0%84-%ED%86%B5%EC%8B%A0-%EB%B0%A9%EB%B2%95Inter-Process-Communication-IPC>