Context Switching

박창희

목차

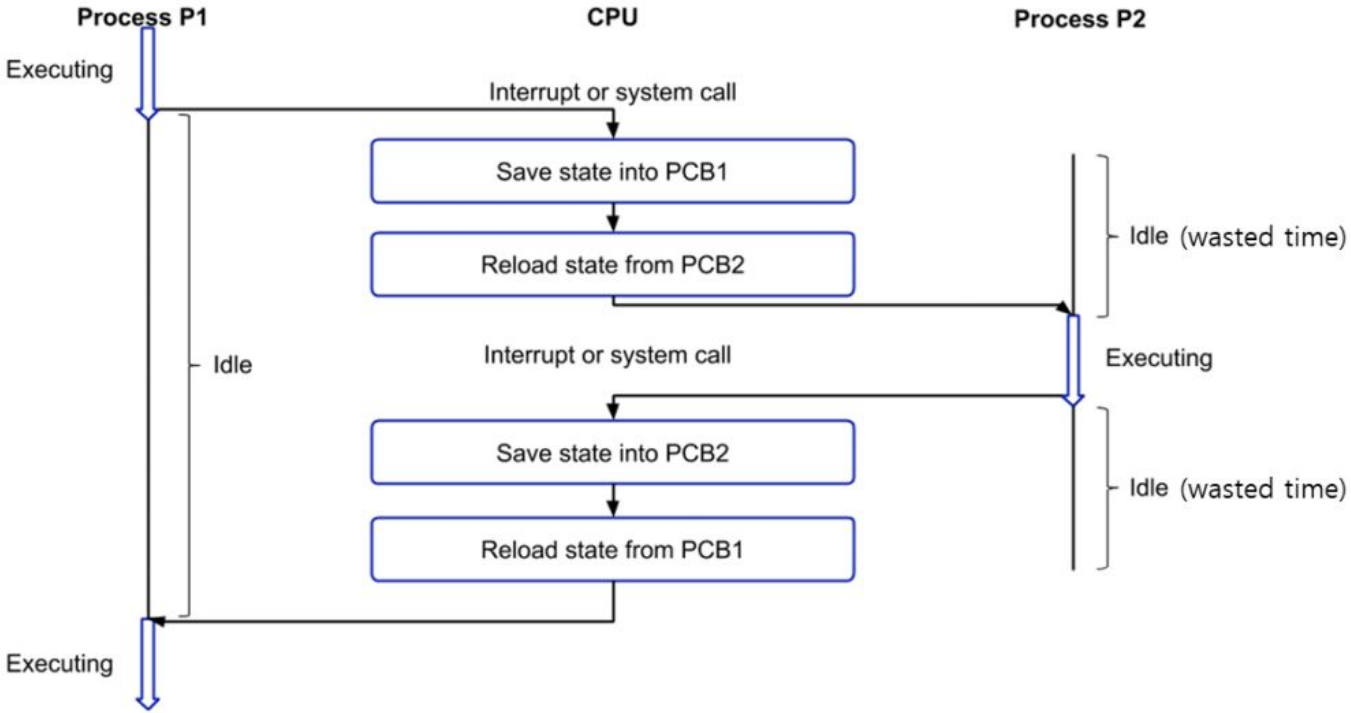
Context Switching 이란?

운영체제가 CPU 자원을 효율적으로 관리하기 위해, 현재 CPU를 점유하고 있는 프로세스(P1)를 중단하고 다른 프로세스(P2)로 전환하는 과정

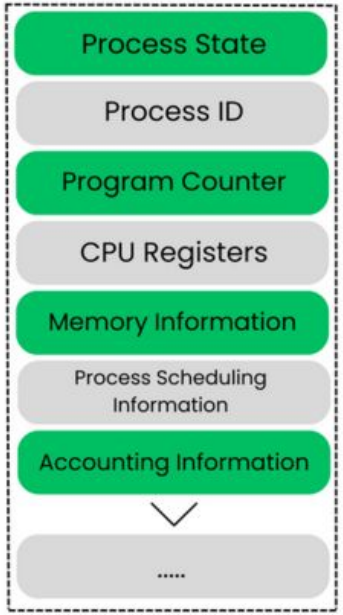
Context Switching이 필요한 이유

- CPU 자원의 효율적 사용
- 멀티태스킹 지원
- 프로세스 간 공정성 보장
- 대기 시간 최소화 (I/O 등)

Context Switching 동작 방식



PCB (Process Control Block)에 저장되는 정보



Process State : Ready , Running Waiting 프로세스의 상태

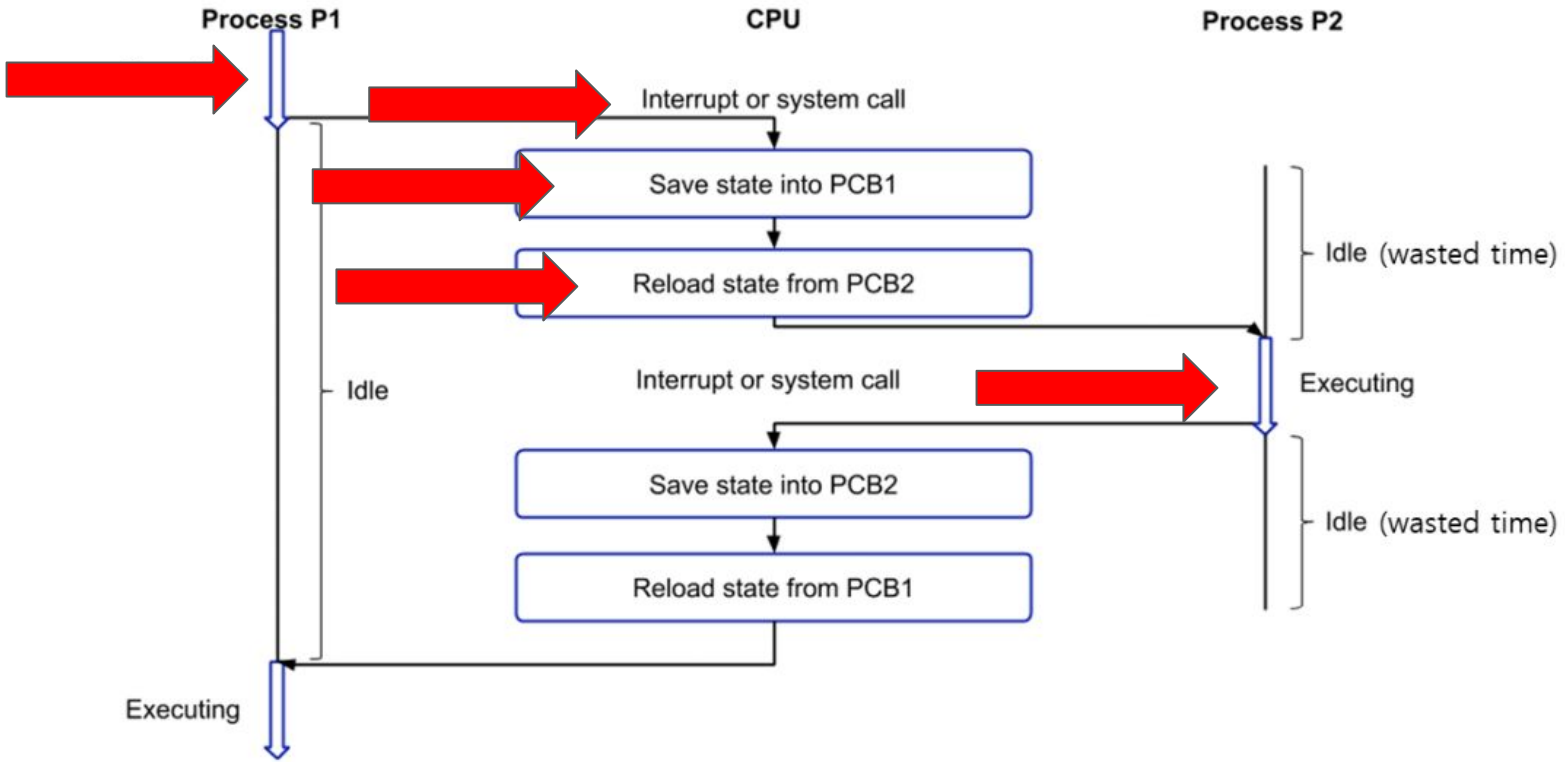
ProcessID : 저장된 프로세스

Program Counter : 프로세스의 다음 실행 명령어 주소

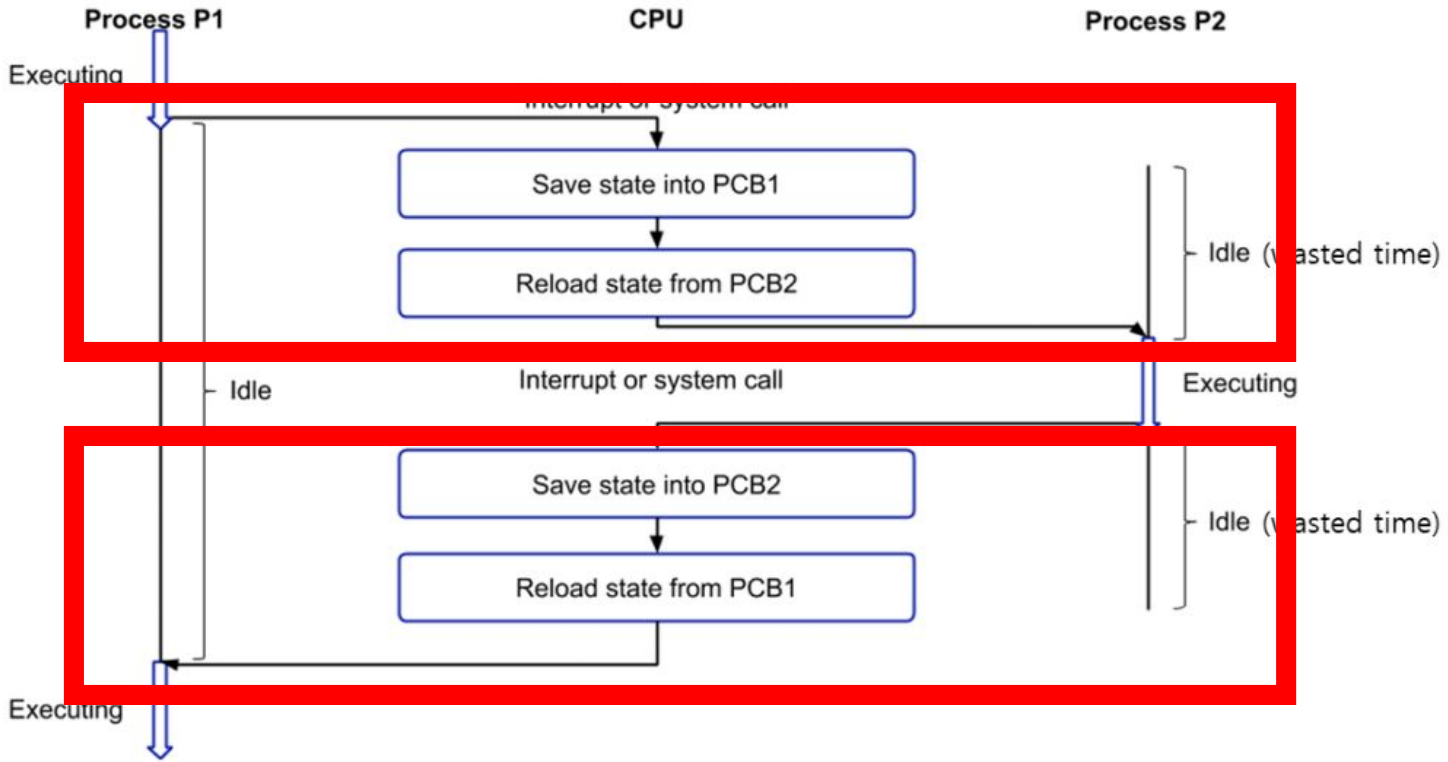
CPU 레지스터 값

프로세스 스케줄링 정보 : 우선순위 / CPU 사용 시간 , 대기 시간, 실행 시간 , 메모리 사용량

Context Switching 동작 방식



Context Switching 오버헤드란?



Context Switching 오버헤드

Context Switching 의 오버헤드 발생은 단순히 PCB 저장/복원에만 그친게 아닌 **하드웨어 레벨**에서 추가적인 비용이 발생!

-> 때문에 Context Switching이 왜 비용이 크다

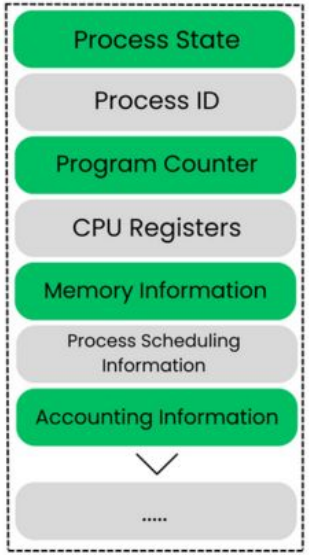
1) **캐시(Cache)** 영향

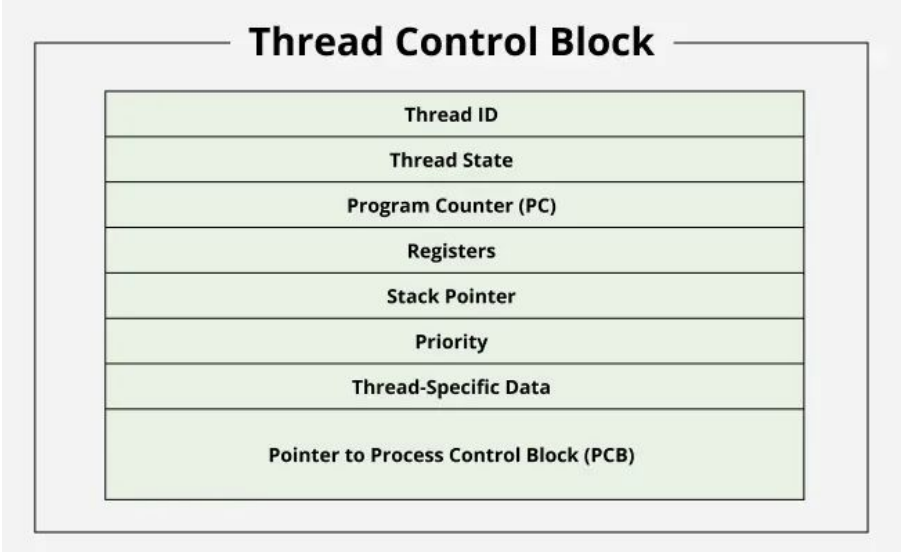
2) **파이프라인(Pipeline)** 손실

3)**TLB(Translation Lookaside Buffer)** 무효화

(Process vs Thread) Context Switching 차이

Process Control Block





Thread Context Switching 더 깊게 보기

Green Thread vs Native Thread Switching

Green Thread(유저 레벨 스레드)

사용자 공간 스케줄러 가 스택/레지스터 교체 → **커널 진입 없음**, **Context Switching 비용이 매우 저렴**.

Java Virtual Thread: Green Thread 모델을 발전시킨 형태 → OS 스레드 위에서 동작하지만, 유저 레벨에서 수백만 개 스레드를 경량으로 관리 가능.

Native Thread(OS 스레드)

운영체제가 직접 관리하는 스레드.

커널이 스케줄링 → **진짜 선점형 멀티태스킹** , 블로킹 syscall 발생 시 해당 스레드만 멈춤.

Context Switching 비용이 Green Thread보다 큼 → **커널 경계 통과 + PCB 교환 + 캐시/TLB 영향**.

https://github.com/SSAFY-Seoul-Class-7/STUDY_Operating_System/blob/main/Context%20Switching%2C%20Scheduling/ThreadSwitchBenchmark.java

```
Mode=both, Tasks=20000, Block=100ms
[virtual] Progress: 2,000 / 20,000
[virtual] Progress: 4,000 / 20,000
[virtual] Progress: 6,000 / 20,000
[virtual] Progress: 8,000 / 20,000
[virtual] Progress: 10,000 / 20,000
[virtual] Progress: 12,000 / 20,000
[virtual] Progress: 14,000 / 20,000
[virtual] Progress: 16,000 / 20,000
[virtual] Progress: 18,000 / 20,000
[virtual] Progress: 20,000 / 20,000
========== VIRTUAL ==========
Tasks               : 20,000
Per-task block      : 100 ms (simulated blocking)

Wall-clock          : 0.142 s
Throughput          : 140845.1 tasks/s
Used heap (approx.) : 0.0 MB
================================
```

- Wall-clock: **0.142 s**
- Throughput: **140,845 tasks/s**
- → Virtual Thread는 20,000개의 블로킹 작업을 동시에 처리하면서도 실제 걸린 시간은 **0.1초 + α 수준**으로 끝남

```
[platform] Progress: 2,000 / 20,000
[platform] Progress: 4,000 / 20,000
[platform] Progress: 6,000 / 20,000
[platform] Progress: 8,000 / 20,000
[platform] Progress: 10,000 / 20,000
[platform] Progress: 12,000 / 20,000
[platform] Progress: 14,000 / 20,000
[platform] Progress: 16,000 / 20,000
[platform] Progress: 18,000 / 20,000
[platform] Progress: 20,000 / 20,000
========== PLATFORM ==========
Tasks               : 20,000
Per-task block      : 100 ms (simulated blocking)
Platform pool size : 11

Wall-clock          : 189.092 s
Throughput          : 105.8 tasks/s
Used heap (approx.) : 0.0 MB
```

- Platform pool size: 11 (코어 수 기반 풀)
- Wall-clock: **189.092 s**
- Throughput: **~105 tasks/s**
- → Platform Thread는 풀 크기(11개)에 제한되어, 20,000개의 블로킹 작업을 순차적으로 분배하느라 **200초 가까이** 걸림

Interrupt 는 context switching 간의 관계

인터럽트는 컨텍스트 스위칭의 한 종류인가? X

Interrupt는 Context Switching을 유발하는 주요 원인중 하나 → **동작의 계기 (Trigger)**

Interrupt 와 Context Switching의 차이

Interrupt의 주체 : CPU 내부의 **Interrupt Controller**가 직접담당

Context Switching 의 주체 : **OS kernel**이 담당

시스템 콜은 Context Switching인가요?

시스템 콜에서 **Kernel Mode ↔ User Mode 전환은 컨텍스트 스위칭이 아니다**

컨텍스트 스위칭은 **프로세스(또는 스레드)** 자체를 바꾸는 동작이다,

User Mode ↔ Kernel Mode 전환은 같은 프로세스가 실행을 이어가는 상황.

- 시스템콜은 실행 모드(권한 레벨)만 바뀌는 것.
- PCB 교체 없음.

BUT, **전환 과정에서 컨텍스트 스위칭이** ***발생할 수도 있다***

복습 겸 면접 질문으로 생각 해봐야 될것들

- **Context Switching**이 왜 비용이 큰가요?
- **Context Switching**은 어떻게 동작하나요?
- **TLB**와 캐시 관점에서 **Context Switching**이 어떤 영향을 주나요?
- 프로세스 **Context Switching**과 스레드 **Context Switching** 차이는 무엇인가요?
- 인터럽트와 **Context Switching**의 관계는 무엇인가요?
- 시스템 콜은 **Context Switching**인가요?
- **Context Switching** 오버헤드를 줄이는 방법은?

<번외>

- **Green Thread**와 **Native Thread Context Switching** 차이는?
- **Java Virtual Thread**는 어떤 장점이 있나요?
- 멀티코어 환경에서 **Context Switching** 비용이 더 커지는 경우는 언제인가요?

Q&A

질문 받아요!