● Multi process에 대해서 설명해라.

▶ Multi process란 두 개 이상의 프로세스가가 동시에 실행되는 것을 말합니다. 동시에라는 말은 동시성(concurrency)과 병렬성(parallelism) 두 가지를 의미합니다. 병렬성은 CPU core가 여러개일 때, 각각의 core가 각각의 프로세스를 연산함으로써 여러 프로세스가 실제로 동시에 실행되는 것입니다. 반면 동시성은 CPU core가 1개일 때, 하나의 CPU core가 여러 프로세스를 짧은 시간동안 번갈아 가면서 연산을 하게 되는 시분할 시스템(time sharing system)으로 실행되는 것으로, 여러 프로세스가 동시에 실행되는 것처럼 보이는 것입니다.

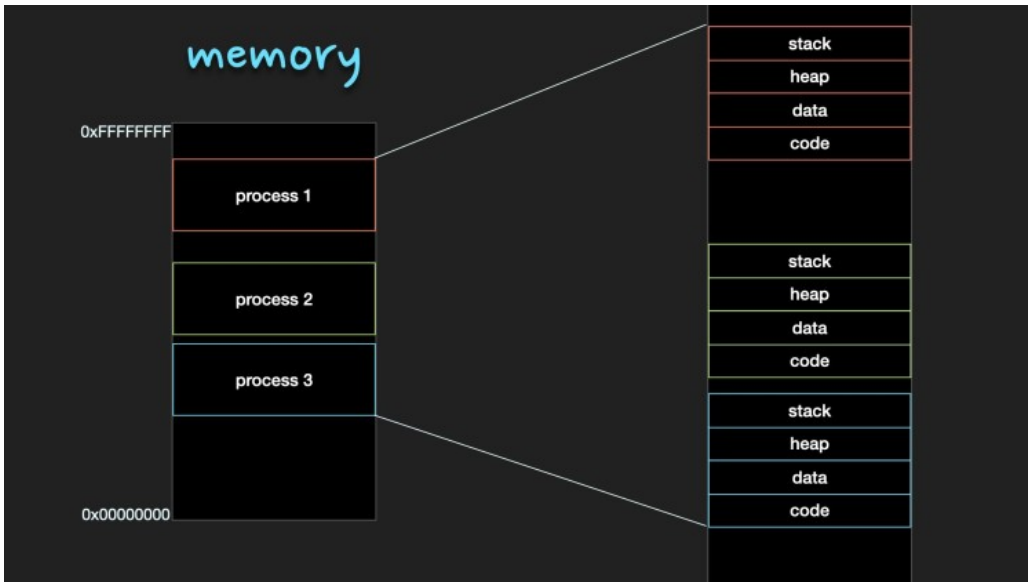
[Single core의 동시성에 초점]

● Single Core의 동시성을 통한 Multi process

▶Multi process, 즉 2개 이상의 프로세스가 동시에 실행될 때, 프로세스들은 CPU와 메모리를 공유하게 됩니다. 메모리의 경우 여러 프로세스들이 각자의 메모리 영역을 차지하여 동시에 적재됩니다. 반면 하나의 CPU는 매 순간 하나의 프로세스만 연산할 수 있지만, CPU의 처리 속도가 워낙 빨라서 수 ms 이내의 짧은 시간동안 여러 프로세스들이 CPU에서 번갈아 실행되기 때문에, 사용자 입장에서는 여러 프로그램이 동시에 실행되는 것처럼 보이는 것입니다. 이처럼 CPU의 작업시간을 여러 process들이 조금씩 나누어 쓰는 시스템을 시분할 시스템(time sharing system)이라고 부릅니다.

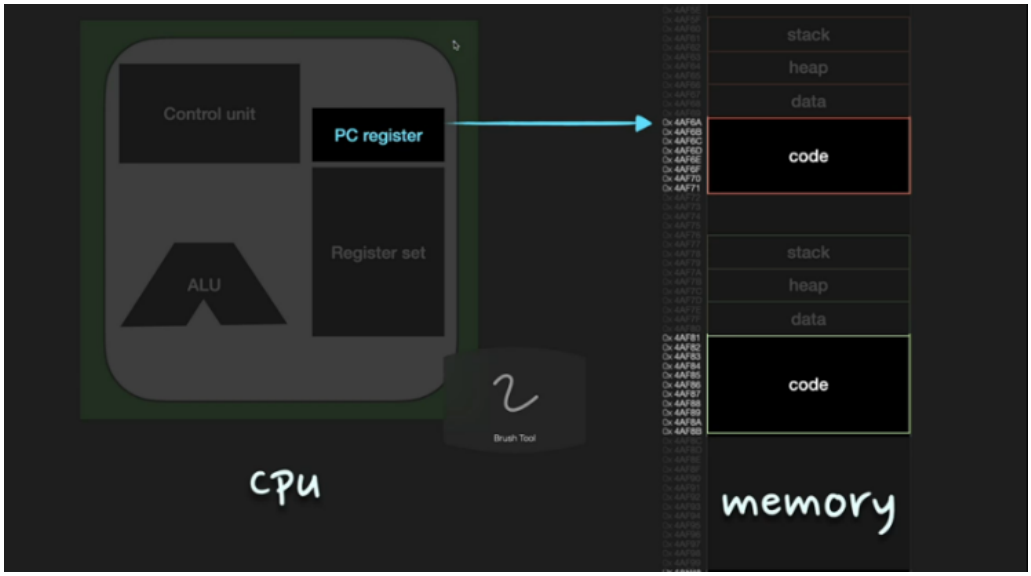
● 메모리 관리는 어떻게 하나?

▶ 여러 프로세스가 동시에 메모리에 적재된 경우, 서로 다른 프로세스의 영역을 침범하지 않도록, 각 프로세스가 자신의 메모리 영역에만 접근하도록 운영체제가 관리해줍니다. (base register, limit register를 통해 각 프로세스를 처리할 때에는 정해진 메모리 구간 외에는 접근할 수 없도록 막는 것입니다.)



● CPU의 연산과 PC register

▶ CPU는 PC(Program counter) register가 가리키고 있는 명령어를 읽어들여 연산을 진행합니다. PC register에는 다음에 실행될 명령어의 주소값이 저장되어 있습니다. multi process시스템에서는 process1이 진행되고 있을 때는 process1의 code 영역을 PC register가 가리키다가, process2가 진행되면 process2의 code 영역을 가리키게 됩니다. CPU는 PC register가 가리키는 곳에 따라 프로세스를 변경해 가면서 명령어를 읽어들이고 연산을 하게 됩니다.



● Context

▶context란 process가 현재 어떤 상태로 수행되고 있는지에 대한 정보입니다. 해당 정보는 PCB(Process Control Block)에 저장을 합니다.

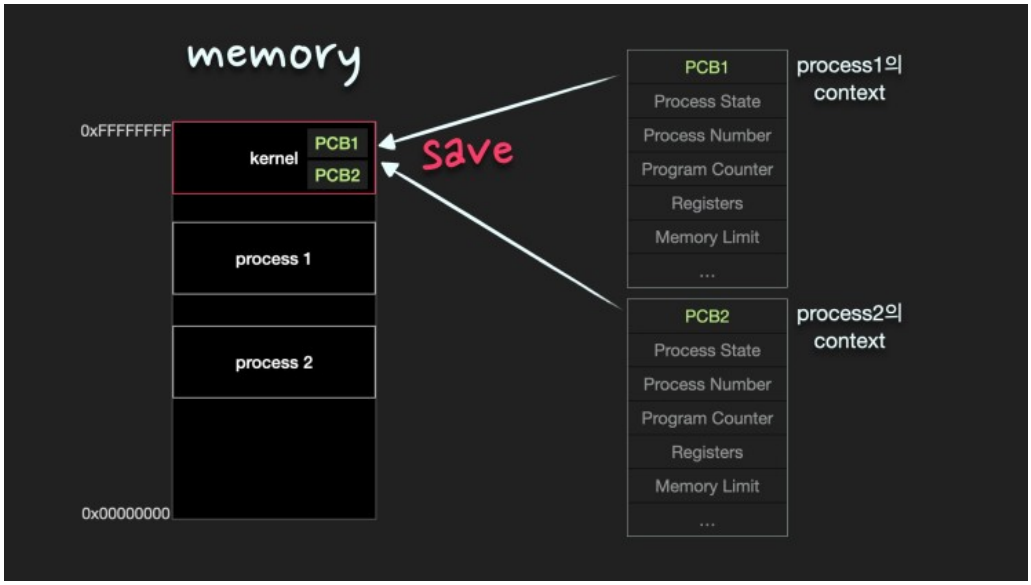
▶ 시분할 시스템에서는 한 프로세스가 매우 짧은 시간동안 CPU를 점유하여 일정부분의 명령을 수행하고, 다든 프로세스에게 넘깁니다. 그 후 차례가 되면 다시 CPU를 점유하여 명령을 수행합니다. 따라서 이전에 어디까지 명령을 수행했고, register에는 어떤 값이 저장되어 있었는지에 대한 정보가 필요하게 됩니다. process가 현재 어떤 상태로 수행되고 있는지에 대한 총체적인 정보가 바로 context입니다. context 정보들은 PCB(Process Control Block)에 저장됩니다.

● PCB (Process Control Block)

▶ PCB는 운영체제가 process에 대해 필요한 정보를 모아놓은 자료구조입니다. PCB에는 프로세스의 중요한 정보가 포함되어 있기 때문에, 일반 사용자가 접근하지 못하도록 보호된 메모리 영역 안에 저장이 됩니다. 일부 운영 체제에서 PCB는 커널 스택에 위치합니다. 이 메모리 영역은 보호를 받으면서도 비교적 접근하기가 편리하기 때문입니다. PCB에는 일반적으로 다음과 같은 정보가 포함됩니다.

(커널은 운영체제. 운영체제도 하나의 프로세스. 메모리에 올라간다. 그래서 커널 영역은 컴퓨터를 키는 순간부터 메모리 영역에 적재된다)

Process State	프로세스의 상태를 표시하는 것으로, new, running, waiting, halted 등의 state가 있습니다
Process Number	해당 process의 number입니다
Program counter(PC)	해당 process가 다음에 실행할 명령어의 주소를 가리킵니다
Registers	컴퓨터 구조에 따라 다양한 수와 유형을 가진 register 값들입니다
CPU 스케쥴링 정보, 우선 순위	
메모리 정보 (해당 process의 주소 공간, 범위 등) & Memory limits	base register, limit register, page table 또는 segment table 등입니다

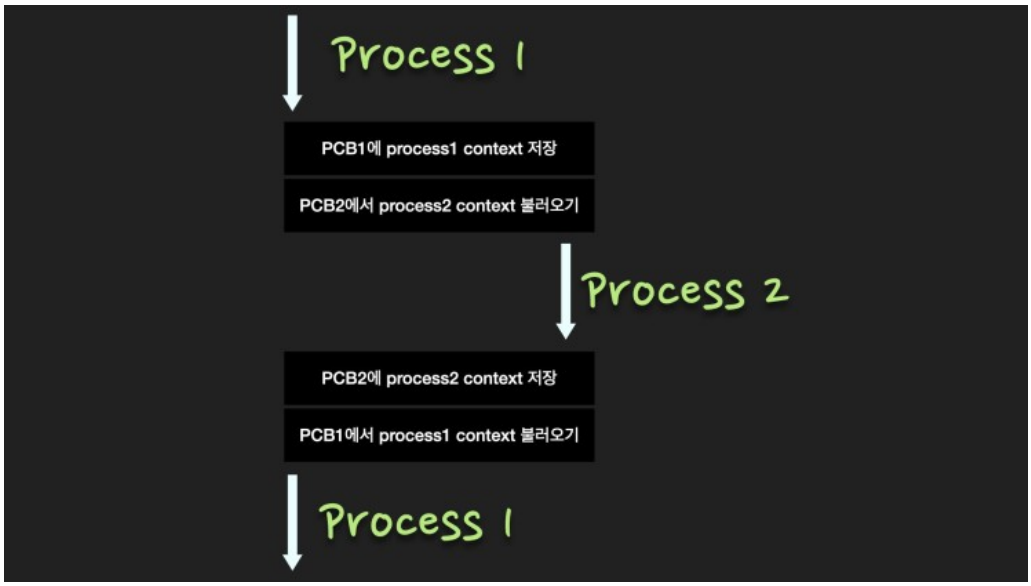


● Process의 state에는 어떤 것들이 있나?

▶ 프로세스는 실행(running), 준비(ready), 봉쇄(wait, sleep, blocked) 세 가지 상태로 구분됩니다. 실행은 프로세스가 CPU를 점유하고 명령을 수행 중인 상태이고, 준비는 CPU만 할당받으면 즉시 명령을 수행할 수 있도록 준비된 상태이고, 봉쇄는 CPU를 할당받아도 명령을 수행할 수 없는 상태(예를 들면 I/O blocking: 입력을 기다리는 경우 등)입니다.

● Context switch

▶ Context switch란 한 프로세스에서 다든 프로세스로 CPU 제어권을 넘겨주는 것을 말합니다. 이 때 이전의 프로세스의 상태(context)를 PCB에 저장하여 보관하고 새로운 프로세스의 PCB를 읽어서 보관된 상태를 복구하는 작업이 이뤄어집니다.



→ PC register과 PCB: 일단 Program Counter register는 CPU에 있습니다. 멀티프로세스 환경에서 context switch가 되면서 현재까지 진행하던 process의 문맥(context)을 어딘가에 저장해야 되기 때문에, PCB에 저장한다고 했죠. 이때 PCB에 PC값도 저장을 하게 됩니다. 그리고 CPU에 있는 PC register에는 새롭게 진행할 process의 PCB에 적혀있는 PC값을 갖게 됩니다 (single thread, multi process)

→ multi thread, multi process 환경에서는 PC 값을 PCB(process control block)가 아닌 TCB(thread control block)에 저장을 합니다. 프로세스 A 에 a,b,c 쓰레드가 있다고 생각해볼게요. a, b, c쓰레드에 해당하는 정보를 저장해두는 a,b,c TCB는 A PCB에 연결이 되어 있습니다. PCB에는 PC값을 저장하지 않고, TCB에서 각각 저장을 해두고 있습니다. 이 때도 TCB는 PC 값을 저장하고 있지만 PC register는 CPU에 속해 있습니다. 또한 Context switch가 될 때마다 TCB에 저장되어있는 PC 값을 PC register에 저장을 하는 것입

니다. 즉, 'a' thread 에서 'b' thread로 context switch가 일어나면, PC register값을 'a' Thread TCB에 저장하고, 'b' Thread TCB의 PC값을 PC register에 저장을 하게 됩니다. CPU입장에서는 PC register에 적힌것만 참조하기 때문에 PC register가 새돕게 가리키는 'b' thread에 해당하는 code영역을 참조하여 실행을 하게 되겠죠. (https://www.inflearn.com/questions/489172)

Resources: inflearn 개발남 노씨