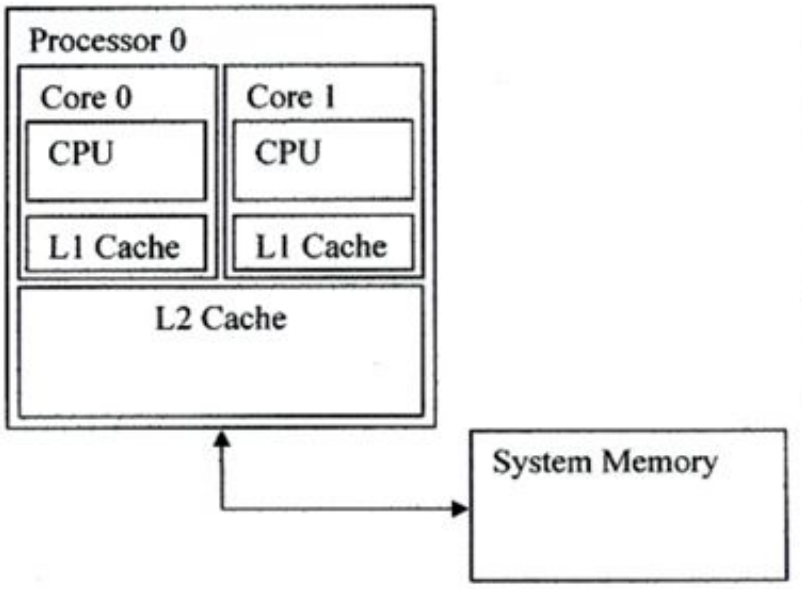
아래의 그림과 같이 Multi-Core Processor에서 각각의 구성요소들이 하는 역할에 대해서 조사하고 Dual-core Processor는 Single-core Processor보다 무조건적으로 두 배 빠른 처리 속도를 가지는가? (물론 단편적인 답은 없다.)

=> 직접 그림을 그려보시는걸 추천!! (잊어버릴 수가 없다.)



Multi-Core Processor with Shared L2 Cache

Processor

- Processor는 컴퓨터에서 작동할 수 있는 기본 명령어를 해석하고 수행한다.
    - 기본 명령에는 수를 이용한 연산, 논리 연산, I/O, ... 등의 OS로부터 전달받은 연산들이 포함됩니다.
- Processor는 명령어를 해석하고 실행하는 곳인 만큼 Processor의 성능이 컴퓨터의 종합적인 성능을 결정한다고 해도 과언이 아니다.
- 이러한 일련의 과정을 공장에 비유한다면, Processor는 공장의 기계들, Processor에서 처리되는 Command 들은 공장에서 가공되어지는 재료들, OS는 이 재료들을 기계에 가공시키는 인부, 기계 등을 의미하겠네요.
- Processor는 4개의 기본 요소로 구성되어져 있으며 이는 **arithmetic logic unit(ALU), the floating point unit(FPU), registers, cache momories** 를 의미한다.
    - arithemetic logic unit과 floating point unit은 기본 또는 복잡한 수식과 논리 연산을 수행하며 결과를 registers에 넘겨준다.
    - register는 또한 명령어를 저장하는 공간으로 사용되어질 수 있으며 Cache는 자주 사용되어지는 값들을 저장하기 하는 작고 빠른 메모리를 의미한다.
- CPU는 대략 3단계의 명령어 사이클을 통해 연산을 수행하며 이는 fetch -> decode -> execute 순이다.
    - Fetch: CPU는 RAM과 같은 메모리를 통해 명령어를 검사한다.

    - Decode: decoder는 명령을 컴퓨터의 다른 구성 요소에 대한 신호로 변환한다.
    - Execute: 디코딩 되어진 명령어는 각 구성요소에 보내지고 기대되어지는 연산이 수행된다.
- 각 processor는 core라고 불리는 개별 processing 단위를 가지고 있다. 코어는 단일 컴퓨팅 작업의 명령을 "GHz"로 측정되는 "clock speed로" 정의된 속도로 처리한다.
    - 2.3GHz 8코어 9세대 Intel Core i9 프로세서란 **2.3GHz의 clock speed를 가진 개별 작업을 수행할 수 있는 코어 8개를 가진 Intel의 9세대 프로세서를 의미하는 것이다.**

스페이스 그레이

2.3GHz 8코어 프로세서
1TB 저장 용량
AMD Radeon Pro 5500M

2.3GHz 8코어 9세대 Intel Core i9 프로세서

최대 4.8GHz Turbo Boost

AMD Radeon Pro 5500M(4GB GDDR6 메모리)

16GB 2666MHz DDR4 메모리

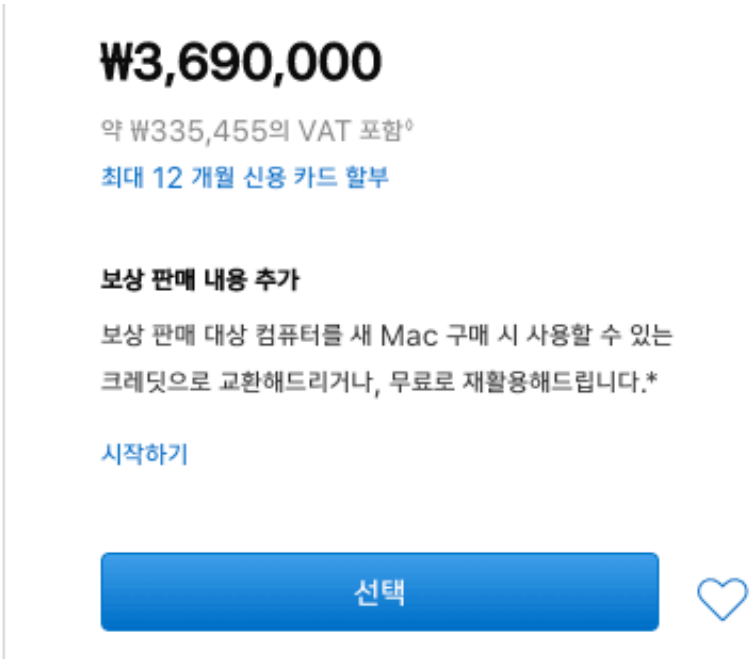
1TB SSD 저장 장치[1]

True Tone이 적용된 16형 Retina 디스플레이

Magic Keyboard

Touch Bar 및 Touch ID

Thunderbolt 3 포트 4개



- **그렇다면, 2.6GHz 6코어 9세대 Intel Core i7 프로세서를 가진 컴퓨터가 2.3GHz 8코어 9세대 Intel Core i9 프로세서를 가진 컴퓨터보다 가격이 저렴한데 무조건 2.6GHz 6코어 9세대 Intel Core i7프로세서를 가진 프로세서보다 빠를까?**

2.6GHz 6코어 프로세서
512GB 저장 용량
AMD Radeon Pro 5300M

2.6GHz 6코어 9세대 Intel Core i7 프로세서

최대 4.5GHz Turbo Boost

AMD Radeon Pro 5300M(4GB GDDR6 메모리)

16GB 2666MHz DDR4 메모리

512GB SSD 저장 장치[1]

True Tone이 적용된 16형 Retina 디스플레이

Magic Keyboard

Touch Bar 및 Touch ID

Thunderbolt 3 포트 4개

₩3,190,000

약 ₩290,000의 VAT 포함◊

최대 12 개월 신용 카드 할부

보상 판매 내용 추가

보상 판매 대상 컴퓨터를 새 Mac 구매 시 사용할 수 있는 크레딧으로 교환해드리거나, 무료로 재활용해드립니다.*

시작하기

선택

도착:

목 2021/07/22 — 무료 배송

추가 배송 옵션 확인

픽업:

Processor vs CPU

- 오늘날 Processor는 CPU, Microprossor와 동일한 용어로 사용되지만 이는 기술적으로 올바른 예시가 아닙니다. CPU는 단지 PC 내에 존재하는 Processor들 중 하나이기 때문입니다.
- Graphics Processing Unit인 GPU도 또 다른 프로세서이고, 심지어는 몇몇의 Hard Drives도 processing을 수행할 수 있기 때문에 CPU, Microporcessor는 동일한 개념이 한 단계 낮은 추상화 계층의 개념이다.

What is the core?

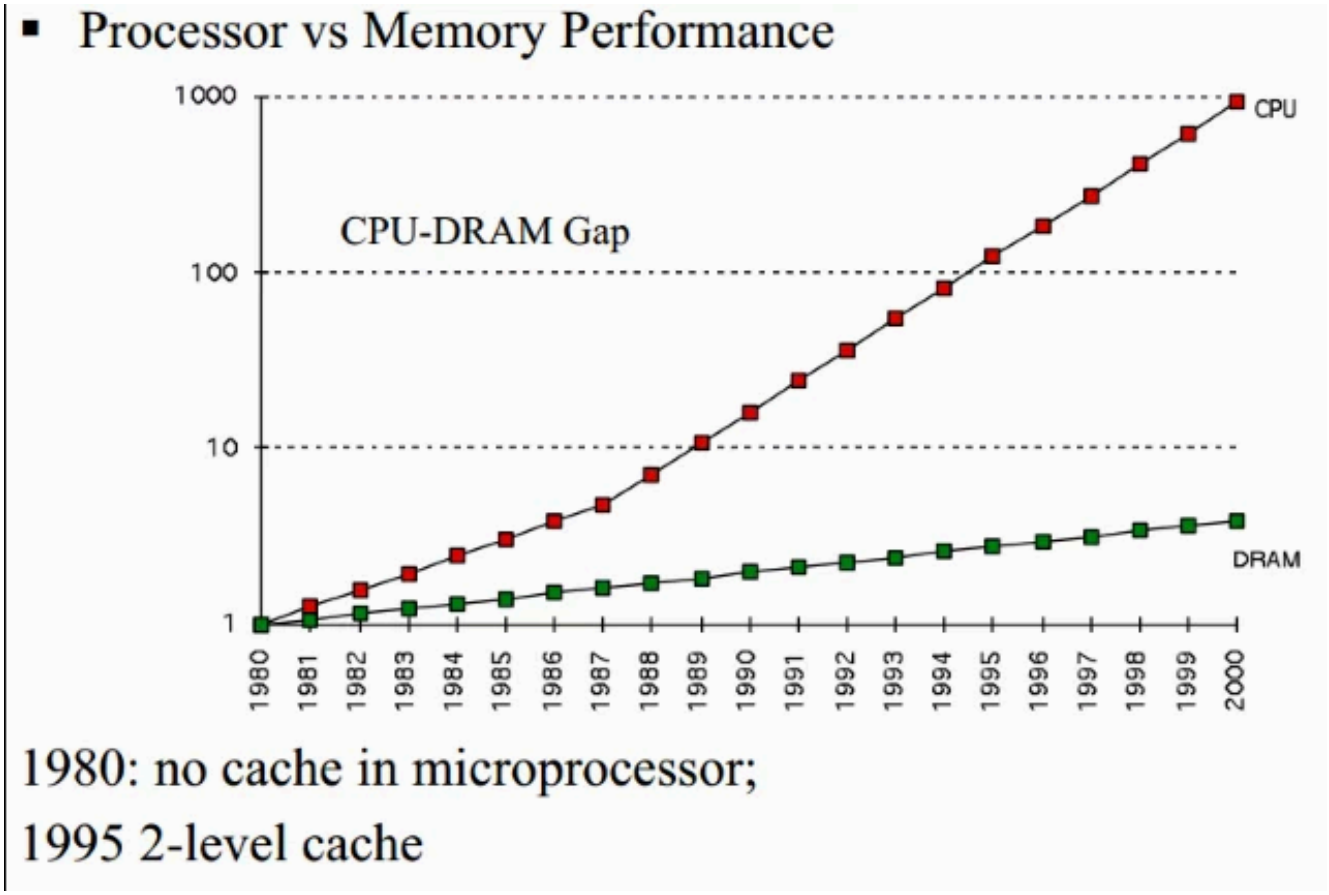
- CPU 내에 명령어를 수행할 수 있는 회로의 집합을 가르킨다.
- OS의 입장에서는 코어 하나를 개별 CPU로 여기며, CPU가 2개의 코어를 가진다면 OS는 2개의 CPU를 가진다고 생각해 한번에 2개의 명령어를 동시에 수행할 수 있는 동시성 연산의 수행이 가능해진다.
- 동시성 연산을 통해 우리는 비동기적인 작업을 할 수 있게 되고 이를 통해 수행시간이 오래걸리는 Blocking I/O 작업을 매번 기다리지 않게끔 만드는 non-blocking I/O를 만들 수 있다.

What is the Multicore?

- Multicore란 하나의 processor가 두개 이상의 core를 가지는 아키텍처 즉, 설계안을 의미한다. 좀 더 하드웨어적으로 설명하자면 Multicore Architecture는 여러 Core를 위치시켜 단일의 물리적 프로세서로 패키징합니다.
- Multicore Architecture의 목적은 동시에 두개 이상의 task를 진행할 수 있는 동시성을 가진 시스템을 만드는게 목적이며 이는 종합적으로 시스템 성능에 매우 긍정적인 영향을 미칩니다.
- 물론 단일 프로세서라도 클럭 속도(GHz)가 높다면 빠르겟지만 보통 이 아키텍처 인코딩, 3d 게임, 비디오 편집 등을 위한 어플리케이션이나 작업을 처리하는데 매우 유용합니다.

CPU Cache vs System Memory

- CPU Cache Memory는 보다 빠른 SRAM을 사용하지만, System Memory는 DRAM을 사용한다.
- DRAM을 사용하는 System Memory는 오랜 기간 데이터를 유지하기 위해 지속적으로 새로 고침되어야해서 전력도 많이 소비하고 속도도 느립니다.
- 하지만 SRAM을 사용하는 CPU Cache는 Refresh를 하지 않기 때문에 더 효율적입니다.
- 아래 그래프에서 볼 수 있듯이 CPU와 DRAM 즉, System Memory의 속도 차이는 1980년대 이후 날이 갈 수록 커졌습니다. 그래서 모든 연산을 처리하기 위해 CPU와 DRAM만을 사용하기에는 병목이 발생해 좋은 성능의 CPU를 쓸 이유가 없겠죠?



그렇다면 CPU를 만드는 Intel과 같은 회사에서는 가만히 있었을까요?? 당연히 자기 CPU를 팔기위해 해결 방법을 세웠겠죠? 그 해결방법이 바로 CPU Cache를 의미하는 L1, L2, L3 Cache 입니다.

이 해결 방법은 CPU vs Memory 사이의 병목 현상을 줄이기 위해서만 사용하는게 아니라 시스템 운영간에 발생하는 대부분의 병목 현상을 없애기 위해 Cache를 사용한다는 겁니다.

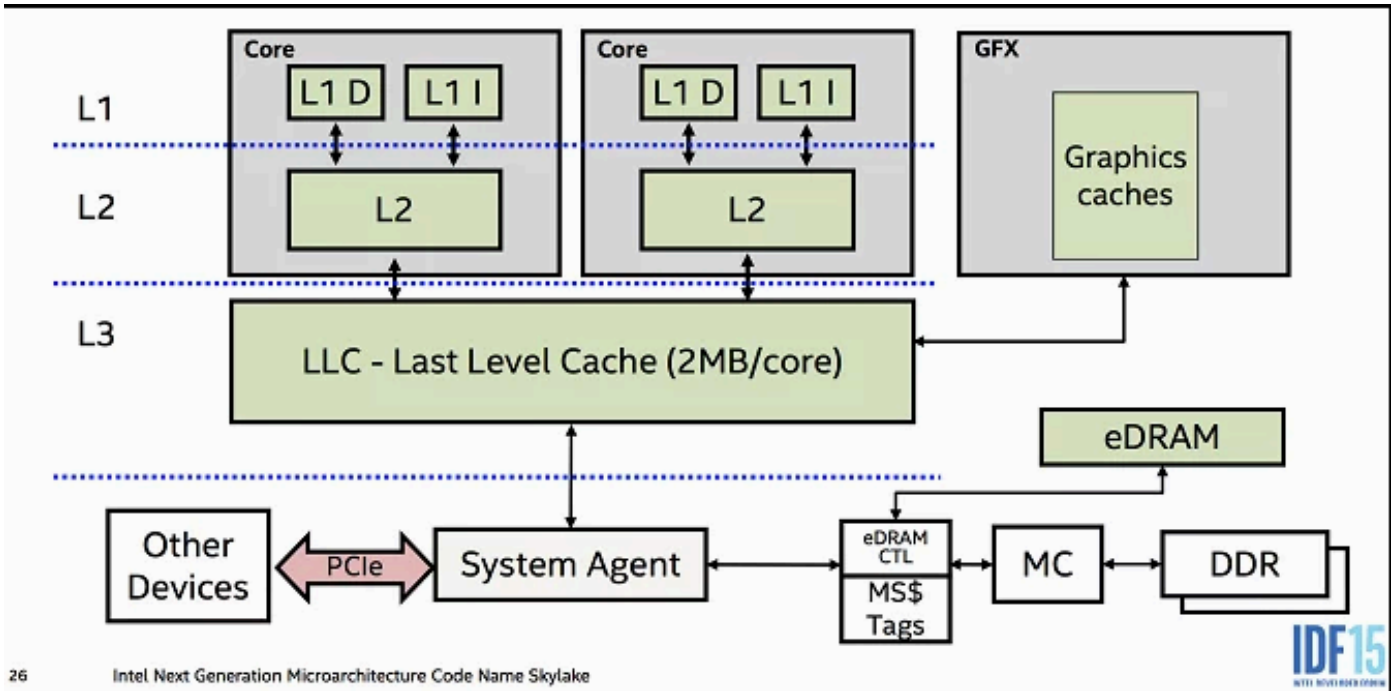
대표적으로 DNS Cache, Application Cache(Redis), Spring Cache, ...가 있겠죠?

만약 가운데 Cache와 같은 역할을 하는 서버를 두면 우리는 빠르게 접근할 수 있는 장점만 있냐? 아뇨, 이젠 Buffer라는 것도 가능해집니다.

- A 서버의 처리 용량이 100M/s 이고 B 서버에 처리 용량이 200M/s 인데 B서버는 주기적으로 A 서버에게 1G 씩의 데이터를 한번에 팍 쏴준다고 가정합시다.
- B -> A 로 1G의 데이터를 200M/s 속도로 쏴준다면 A 서버가 버틸 수 있을까요? 당연히 터지겠죠, 그래서 가운데 서버C를 하나 더 둬서 B는 C에게 200M/s 속도로 1G의 데이터를 쏘고, C는 A에게 100M/s 속도로 1G의 속도로 쏴주는 버퍼링이 가능하겠죠?? 서로 정상적으로 서비스가 잘 될테구요^^
- 이러한 개념또한 현업에서는 다 적용되어 있습니다. 단지 OS 레벨에서 Application 레벨로 추상화 계층이 올라간 것 뿐이죠... 그래서 OS가 중요하단 겁니다.

L1 Cache, L2 Cache, L3 Cache: What's the Difference?

L1, L2, L3 Cache Level을 포함하고 있는 컴퓨터 구조는 다음과 같습니다.



- 현대 프로세서에서, CPU Cache Memory는 사이즈는 증가하고 속도는 감소하는 순서로 L1, L2, L3 캐시인 3부분으로 나뉩니다.
- 과거 프로세서는 L3 Cache를 포함하지 않고, 시스템은 직접적으로 L2 Cache와 상호작용 합니다.

L1 Cache (generally 64KB per core)

- L1 Cache는 L1 Data Cache와 L1 Instruction Cache로 나뉘구요, L1 Instruction Cache는 CPU Core에 의해서 소비되어지는 명령어를 포함하고 전자는 main memory에 다시 기록될 데이터를 보유하는데 사용됩니다.
- L1 Cache는 instruction cache로써의 역할을 하는 것 뿐만 아니라, 사전 디코딩 데이터 및 분기 정보를 공유합니다. 게다가 L1 data Cache는 종종 output-cache로써의 역할을 하는 반면, L1 Instrctuion Cache는 input-Cache 로써의 역할을 합니다.

L2 Cache (generally 512KB per core)

- L2 Cache는 L1 Cache에 비해 더 크지만 느리며, 각 코어에 위치한 마지막 계층의 캐시입니다.

L3 Cache

- 이는 가장 낮은 레벨의 Cache를 의미하며, 10MB부터 256MB까지의 캐시가 위치해있습니다.

Using Cache

- CPU가 data가 필요할 때, 관련 코어의 L1 Cache를 먼저 찾아봅니다. 만약 데이터를 발견하지 못하면, L2, L3 Cache를 다음으로 찾아봅니다.
- 만약 필요한 데이터를 Cache에서 발견했다면 이는 Cache hit라고 하구요, 발견하지 못했다면 이를 Cache miss 라고 합니다. 이러한 지표를 Cache retio 라고 하는거구요.
- **당연히 Cache 크기가 커짐에 따라서 Cache ratio는 올라가겠죠? 하지만, 당연히 검색할 곳이 늘어나다 보니 Cache 자체의 Performance는 떨어지겠죠.... trade off 관계입니다.** ㅎㅎ

Dual-core Processor는 Single-core Processor보다 무조건적으로 두 배 빠른 처리 속도를 가지는가?

- 뭐 아시다 싶이, 무조건적으로 두 배 빠른 처리 속도를 가지진 않겠지요??
- 이는 Processor에게 맡겨지는 작업에 종속되어집니다. 만약 하나의 작업이 둘로 나누어질 수 있고 서로의 수행 여부에 의존하지 않고 반복적인 작업을 수행한다면 어쩌면 두배보다 빠른 속도를 가질 수도 있겠지요?? CPU의 성능은 두배지만, 우리에게는 Cache라는 도구가 존재하기 때문.... 하지만 둘로 나누어진 대부분의 작업의 경우 서로 종속되기 때문에 아무리 Core가 두개여도, Core마다 L1, L2 Cache가 있어도 두 배 정도의 빠른 처리 속도를 가지는건 힘들것 같습니다.
- Processor의 처리속도는 Task 뿐만 아니라, Processor 자체에도 종속되어진다.