대규모 병렬 컴퓨팅

Massively Parallel Computing with CUDA

biztripcru@gmail.com

대규모 병렬 컴퓨팅 소개

Introduction to Massively Parallel Computing

내용 contents

- **병렬 컴퓨팅 parallel computing 의 도입 과정**
- **CPU / GPU의 설계 철학 design philosophy**
  - 반응 시간 latency 우선
  - 처리량 throughput 우선
- **대규모 병렬 컴퓨팅 massively parallel computing**
  - **MPC** = massively parallel computing

CPU와 GPU

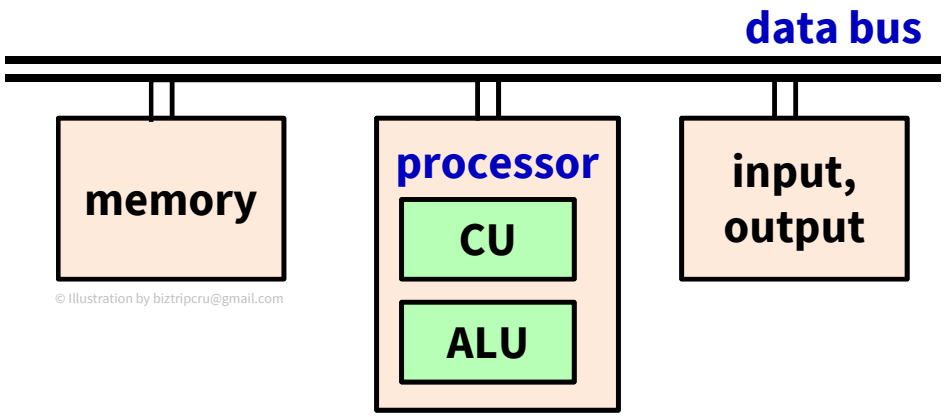
- **CPU : central processing unit**
  - 중앙 처리 장치
  - 보통 1개의 독립된 칩 → **CPU 칩** chip
  - Intel 인텔, AMD 에이엠디, ARM 암, …
- **GPU : graphics processing unit**
  - 그래픽스 처리 장치
  - 보통 카드 형태 → **그래픽 카드, 비디오 카드** video card
  - NVIDIA 엔비디아, AMD (ex-ATI), Intel (CPU-integrated), …
- **새로운 계산 기기** new computing device !

public domain
https://commons.wikimedia.org/wiki/File:Computer_from_inside_018.jpg

pixabay license
https://pixabay.com/ko/photos/%EC%B9%B4%EB%93%9C-%EA%B7%B8%EB%9E%98%ED%94%BD-%EC%B9%B4%EB%93%9C-%ED%99%94%EC%9D%B4%ED%8A%B8-329267/

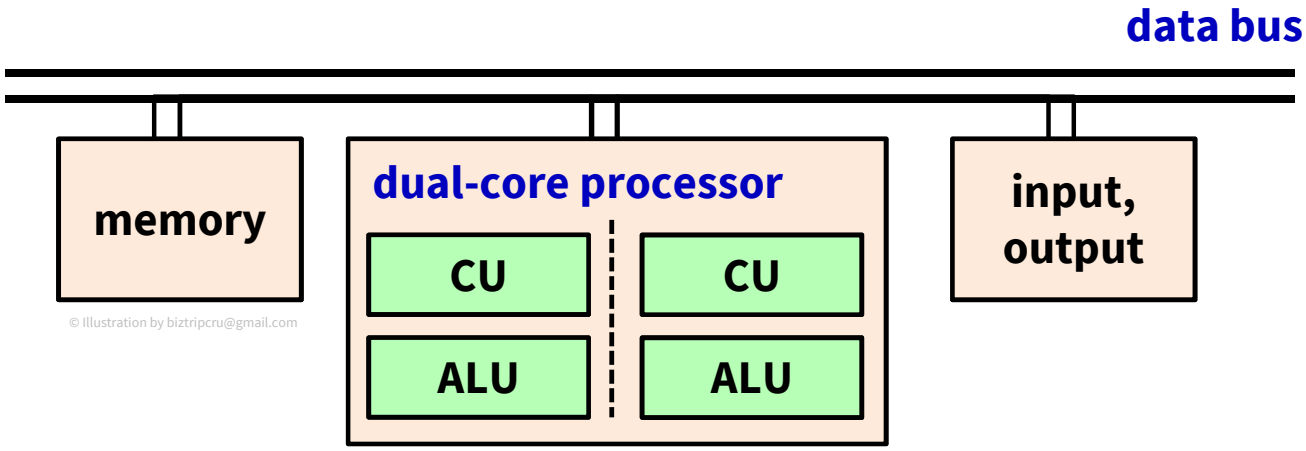
병렬 컴퓨팅의 도입 과정

- **2003년 이전: 싱글 코어 single core CPU**
  - 폰 노이만 von Neumann 구조
  - **코어 core** = CPU (또는 processor) 내의 계산 유닛 computation unit
    - **ALU arithmetic-logic unit** 을 의미하는 경우가 많음
    - **CU control unit** 은 ALU 의 제어를 담당



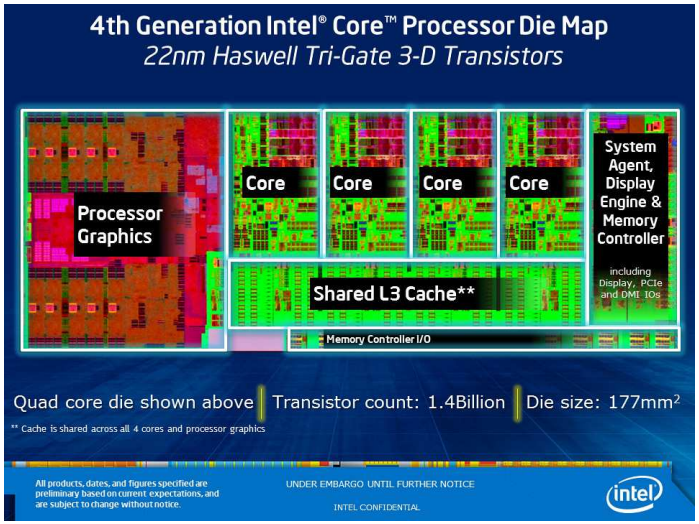
병렬 컴퓨팅의 도입 과정 계속

- 2003년 이후
  - 멀티 코어 multi-core CPU : 2 ~ 32+ cores
  - 매니 코어 many-core GPU : 1024 ~ 8192+ cores



병렬 컴퓨팅의 도입 과정 계속

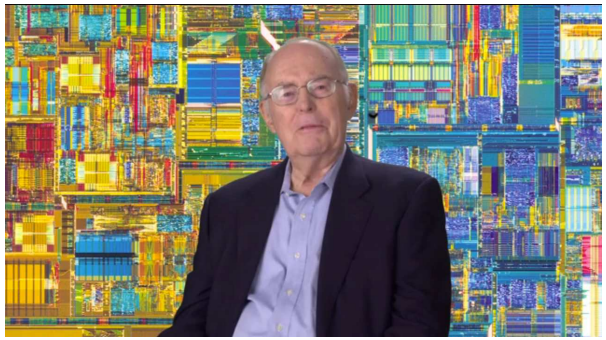
- **병렬 컴퓨팅 parallel computing**
  - 과거: 슈퍼 컴퓨터super-computer 전용
  - 현재: **스마트폰 CPU도 multi-core** → 언제 어디서나! **유비쿼터스 ubiquitous**
- **프로그래머 입장: "must-know"**
- **현재의 딜레마 dilemma**
  - HW 는 이미 병렬 기기 parallel device
  - SW 는 아직 순차 처리 sequential processing

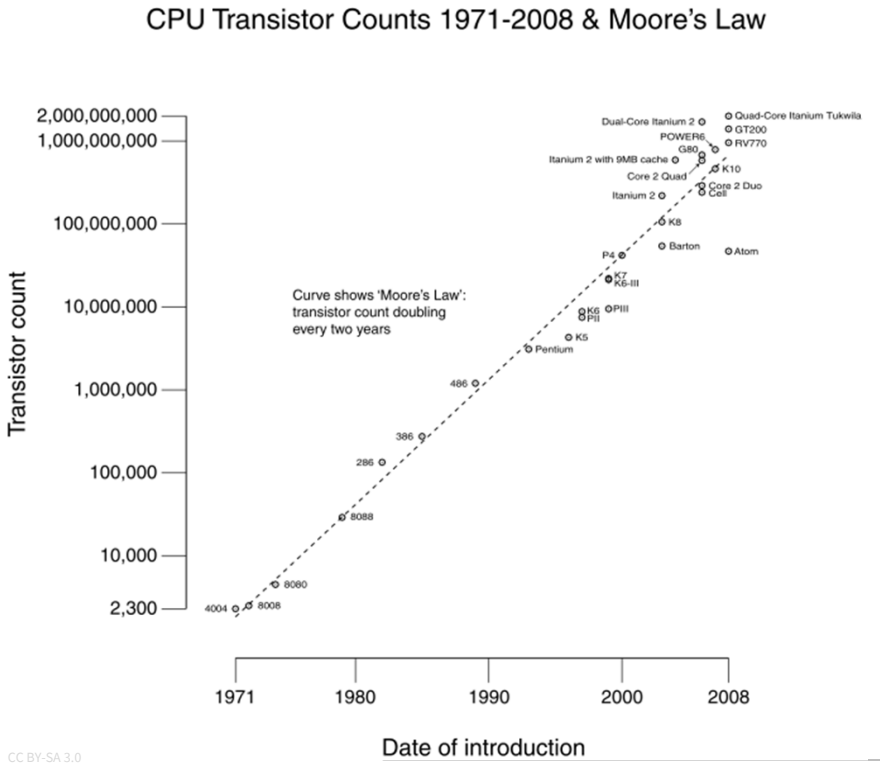


Intel CPU 내부 구성도 (내장 그래픽 포함)

무어의 법칙 Moore's law

- **IC칩 1개에 들어가는 트랜지스터 숫자는 매 2년마다 2배가 된다.**
  - The number of transistors on an integrated circuit **doubles every two years**.
  - Gordon E. Moore (1929, ex-Intel CEO)





물리적 한계 physical limitation

- **전자의 속도 = 광속 light speed = 3 x 10^8 m/sec**
- **3 GHz CPU 기준**
  - 1 clock 소요 시간: (1 / 3 x 10^9) sec
  - **1 clock 당 전자의 이동 거리:**
    - (3 x 10^8) / (3 x 10^9) = (1 / 10) m = **10 cm**
- **10 GHz CPU 기준**
  - 1 clock 당 전자의 이동 거리:
    - (3 x 10^8) / (10 x 10^9) = (3 / 100) m = **3 cm**
- **CPU 클럭 clock을 올리는 것은 이미 한계에 도달 !**
  - 남아도는 트랜지스터를 어디에 쓸 것인가?

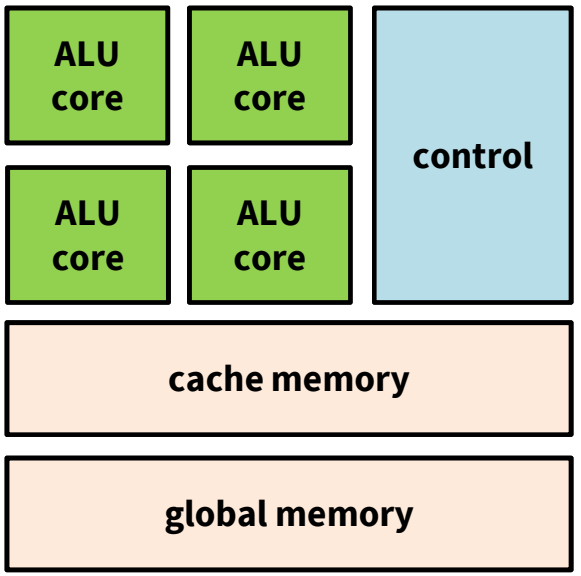
factor	symbol	name
10^{24}	Y	yotta 요타
10^{21}	Z	zetta 제타
10^{18}	E	exa 엑사
10^{15}	P	peta 페타
10^{12}	T	tera 테라
10^{9}	G	giga 기가
10^{6}	M	mega 메가
10^{3}	k	kilo 킬로

© Illustration by biztripcru@gmail.com

CPU: 고성능 멀티 코어 multi-core

- 목표: **반응 시간 latency 단축**
  - 순차 처리 sequential processing 에 적합
  - 기존 **고성능 코어**를 추가

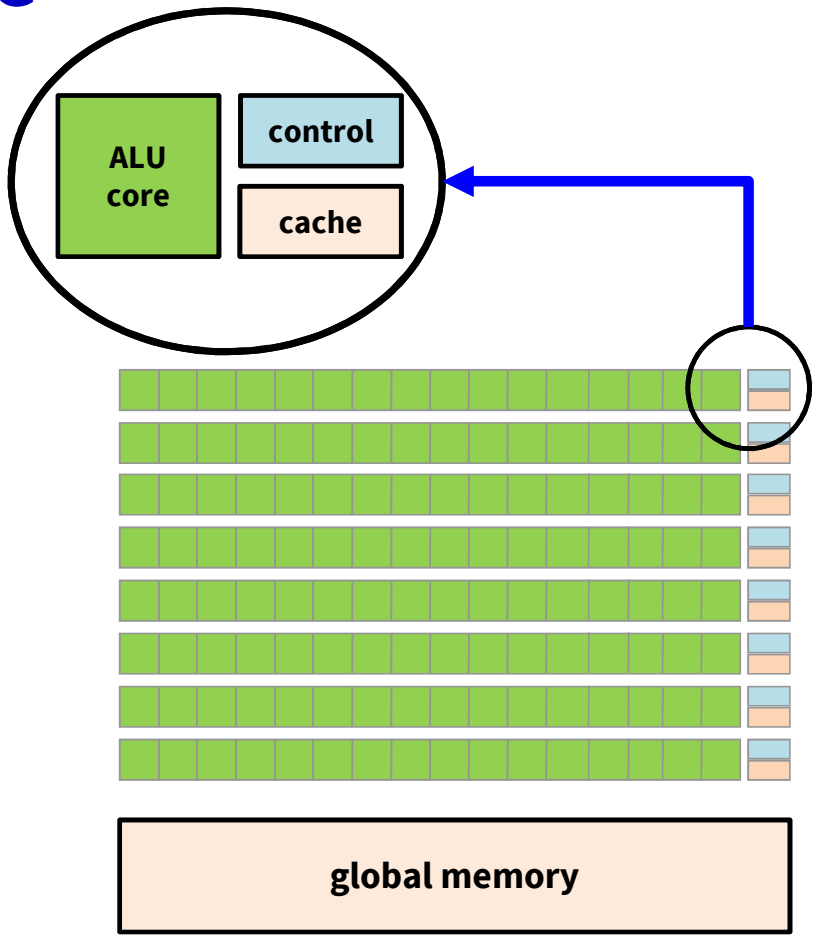
- **대용량 캐쉬**
- **대규모 CU control unit**
- **고성능 ALU**
  - **복잡한 명령어를 빨리 처리**



GPU: 대규모 매니 코어 many-core

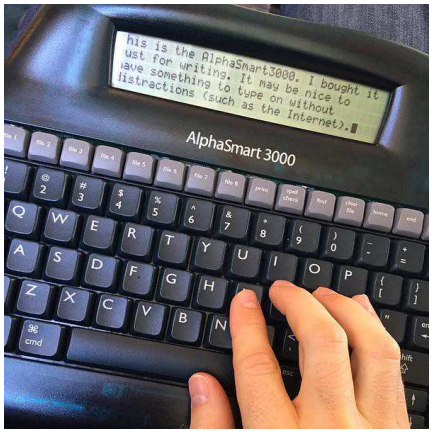
- **목표: 처리량 throughput 확대**
  - 병렬 처리 parallel processing 에 집중
  - (성능에 무관하게) **코어 숫자** 증가에 집중

- **소규모 캐쉬 메모리 cache memory, 간단한 CU**
- **대규모 ALU → 1000개 이상의 동시 실행**
- **쓰레드 풀 thread pool 의 효과적 집중**
  - 최종적으로, **단위시간당 처리량 대폭 확대 !**



적용 분야의 차이

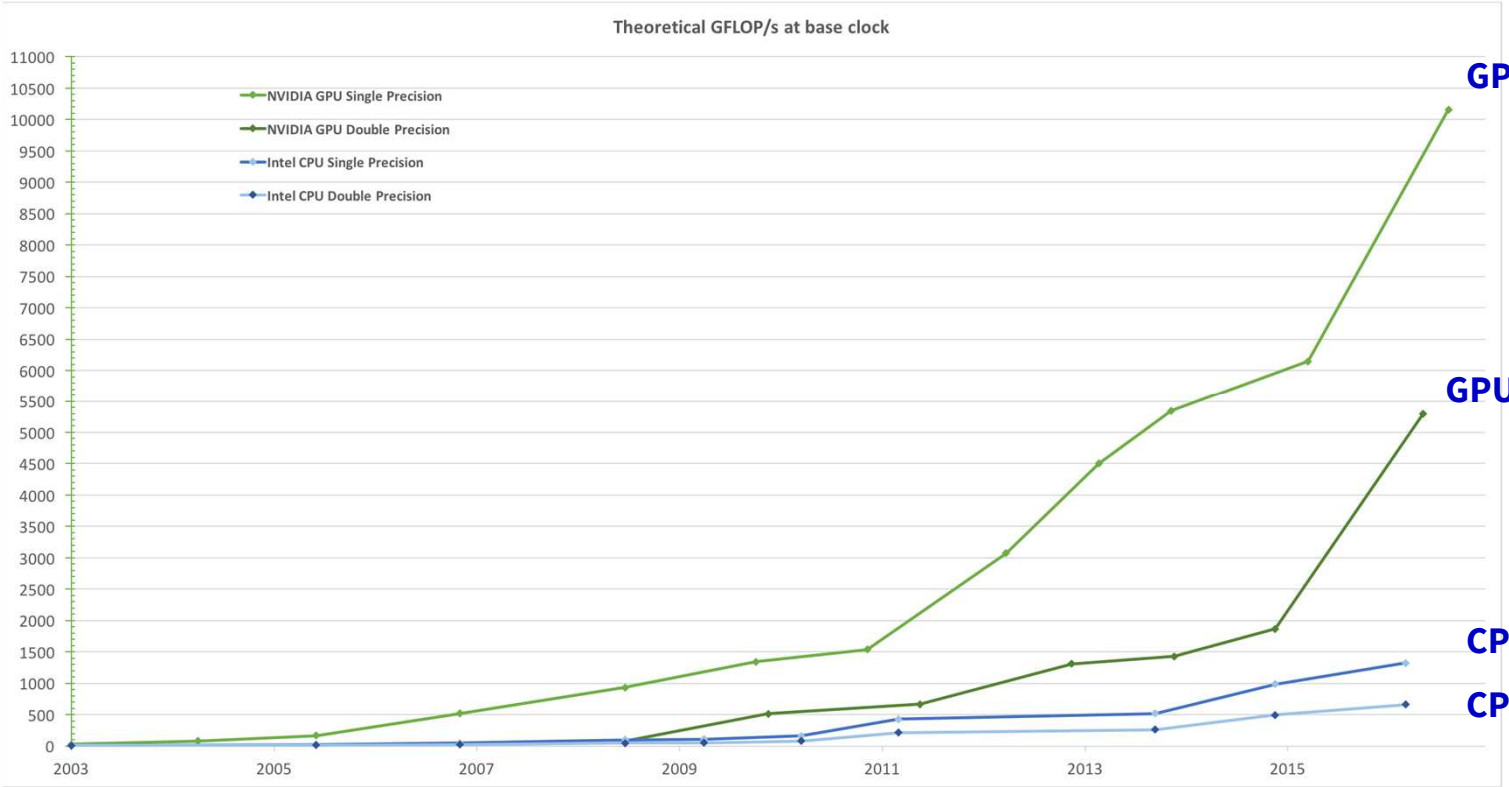
- **CPU : 순차 처리에 적합**
  - sequential processing
- **예: 워드 프로세싱**
  - 사용자 입력 → **화면 반응**까지
  - GPU 보다 최소한 10배 빠름



- **GPU : 대규모 병렬 처리에 적합**
  - massively parallel processing
- **예: 빅 데이터의 통계 처리**
  - 대용량 입력을 한번에 실행
  - **대규모 계산 필요**
  - CPU보다 1,000배 이상 빠름

대규모 병렬 컴퓨팅 MPC massively parallel processing

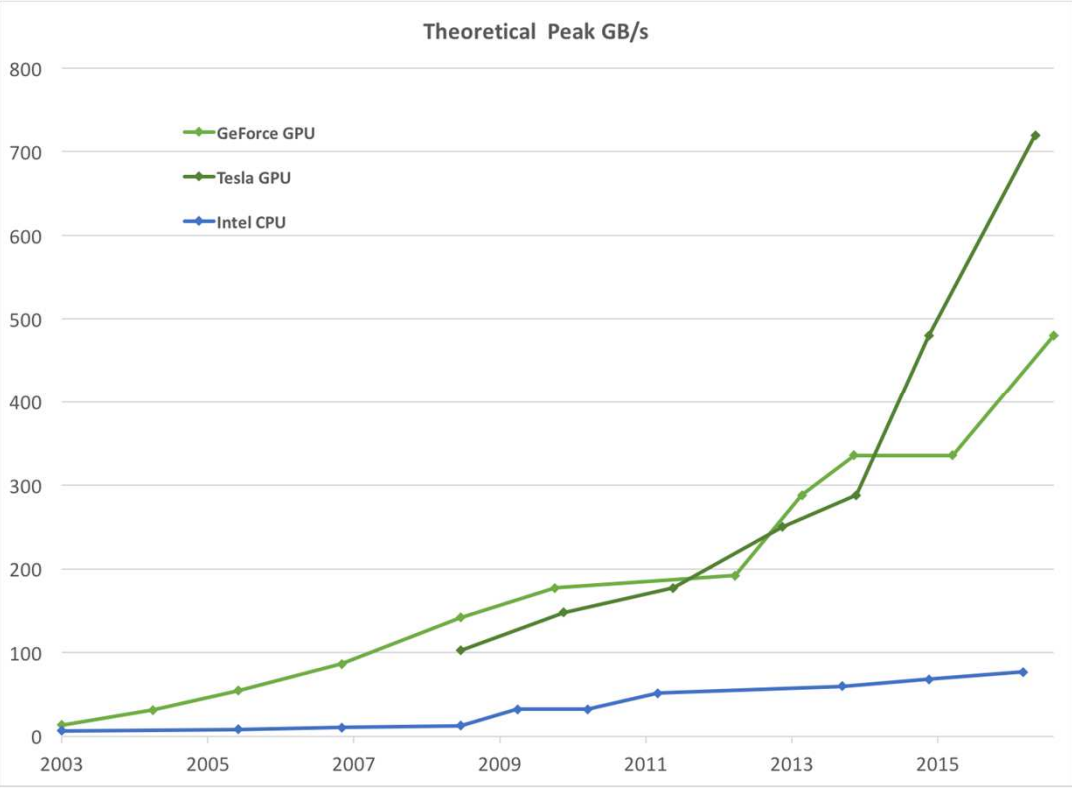
- 조용한 혁명: TFLOPs (GPU) vs. GFLOPs (CPU)



Copied from "Figure 1. Floating-Point Operations per Second for the CPU and GPU" in CUDA Toolkit Documentation, v10.0, NVIDIA, under their EULA

대규모 병렬 컴퓨팅 MPC 계속

- 10배 정도의 전송 속도 bandwidth 차이



Tesla GPU ↔ VRAM (video memory)와 데이터 전송

GeForce GPU ↔ VRAM (video memory)와 데이터 전송

Intel CPU ↔ RAM (main memory)와 데이터 전송

Copied from "Figure 2. Memory Bandwidth for the CPU and GPU" in CUDA Toolkit Documentation, v10.0, NVIDIA, under their EULA

대규모 병렬 컴퓨팅 MPC 계속

- **병렬 컴퓨팅의 특성**
  - **프로그래머: 알고리즘을 병렬로 재설계해야**
  - **대규모 데이터**를 제공한다고 가정 → **빅 데이터** big data → **데이터 사이언스** data science
    - (**데이터 규모 ≫ GPU 코어 숫자**) 라고 가정

- **GPU 기술**
  - GeForce RTX 3090 기준
  - 10,496 cores + 24GB memory
  - 높은 처리량: 29 ~ 36 TFLOPS
  - 높은 전송속도: 936 GB/s
  - 높은 보급률

대규모 병렬 처리 모델

- **모델 model = 디바이스 + 프로그래밍 언어 + 컴파일러 + 라이브러리 + …**
- **OpenMP 오픈엠피 Open Multi-Processing**
  - 멀티 코어 CPU 용 – Visual Studio 에서 사용 가능
  - 최근에 GPU로 확장 중
- **CUDA 쿠다 Compute Unified Device Architecture**
  - NVIDIA GPU 전용 → 현재는 **클라우드 컴퓨팅 cloud computing** 사용 가능
- **OpenCL 오픈씨엘 Open Computing Language**
  - CPU / GPU / FPGA 모두 제공
  - Apple, Intel, AMD/ATI, NVIDIA, …
  - 범용성을 추구 – 좀더 복잡한 모델, 교육용으로는 레벨이 높음

내용 contents

- **병렬 컴퓨팅 parallel computing 의 도입 과정**
- **CPU / GPU의 설계 철학 design philosophy**
  - 반응 시간 latency 우선
  - 처리량 throughput 우선
- **대규모 병렬 컴퓨팅 massively parallel computing**
  - MPC = massively parallel computing

대규모 병렬 컴퓨팅 소개

폰트 정상 출력 → 큰 교자 타고 혼례 치른 날
정참판 양반댁 규수 큰 교자 타고 혼례 치른 날
정참판 양반댁 규수 큰 교자 타고 혼례 치른 날
본고딕 Noto Sans KR

The quick brown fox jumps over the lazy dog
The quick brown fox jumps over the lazy dog
The quick brown fox jumps over the lazy dog
Source Sans Pro