CUDA ON ARM PLATFORM—GPU硬件架构

NVIDIA企业级开发者社区 何琨

CUDA并行计算基础

AGENDA

- 课程说明
- GPU架构的基本原理
- GPU硬件平台
- 基于Arm 和GPU的平台架构
- 最新的GPU应用领域

课程说明

- 有志于从事CUDA/GPU系统开发的开发者
- 使用CUDA/GPU并行计算系统的科研工作者
- 计算机，电子，自动化，生医等相关专业的硕士研究生或高年级本科生

- 了解和掌握GPU/CUDA并行计算系统的分析，设计，开发，调试和优化方法
- GPU并行计算系统的分析能力，编程能力，开发经验

- 风格：中英夹杂

NVIDIA.

课程说明

基础知识

- 1) 计算机体系结构基础
- 2) C语言程序设计
- 3)计算机算法基础
- 4)线性代数

课程内容参考

- 1. CUDA C Programming Guide，NVIDIA Corp.
- 2. CUDA Best Practice Guide, NVIDIA Corp.
- 3. CUDA编程—基础与实践 樊哲勇 著
- 4. https://github.com/brucefan1983/CUDA-Programming

NVIDIA

NVIDIA 开发者

15

500k

“Best Places to Work in 2021”
2021年世界上最好的工作场所

GLASSDOOR

“Most Innovative Companies”
最具创新力的公司

FAST COMPANY

“100 Best Companies to Work For”
100个最适宜工作的公司

FORTUNE

“50 Smartest Companies”
50家最聪明的公司

MIT TECH REVIEW

“World’s Best Performing CEO”
世界上最好的CEO

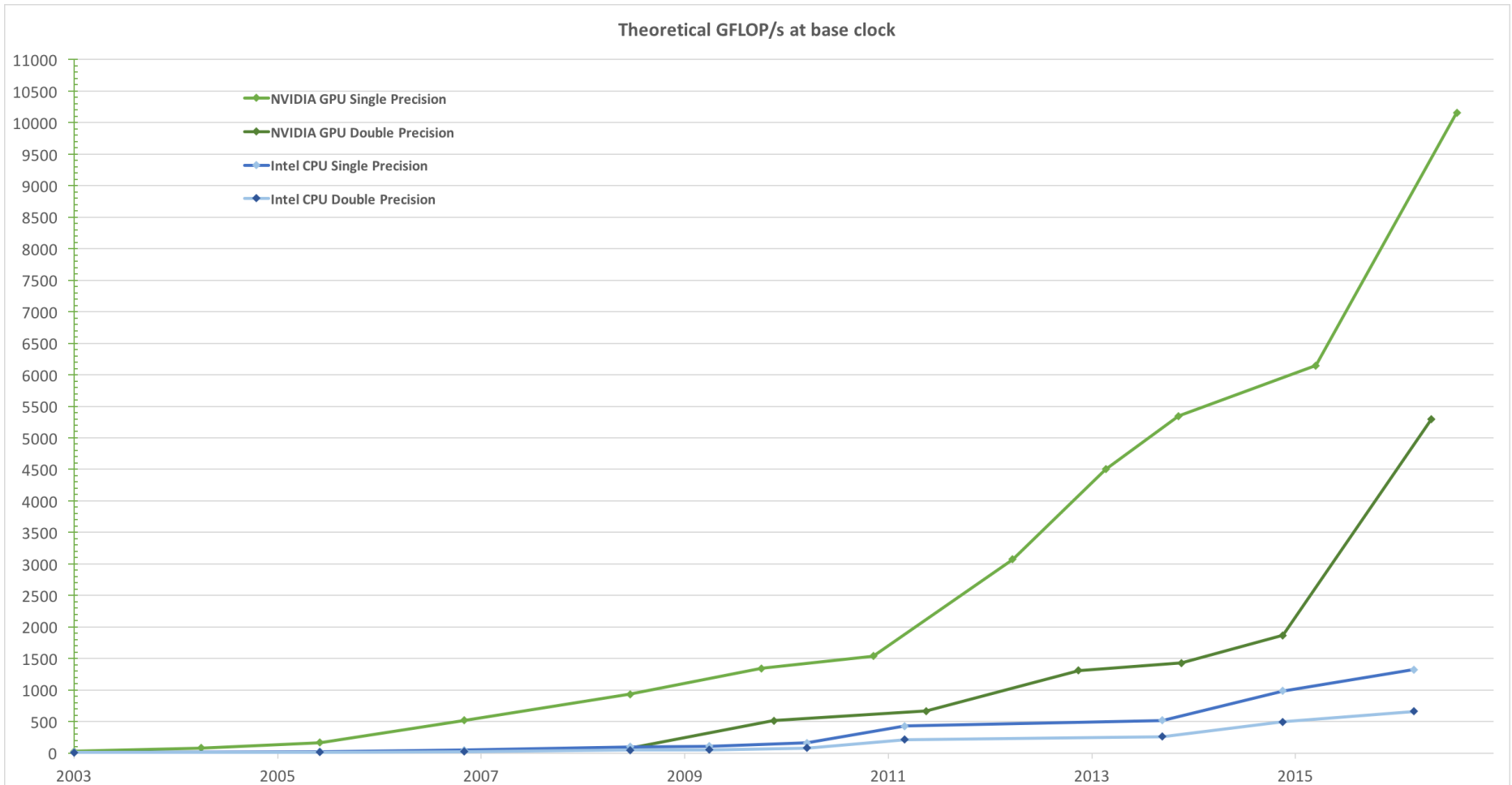
HARVARD BUSINESS REVIEW

“World’s Best CEOs”
世界上最好的CEO

BARRON’S

2006 CUDA问世 | 2008 Tegra发布 | 2009 Fermi发布 | 2011 10亿台设备 | 2012 Kepler发布 | 2014 TK1发布 | 2015 TitainX & TX1 | 2016 Pascal, DGX-1 & drive Px2 | 2017 Volta & TX2 | 2018 Turing | 2019 Nvidia Studio | 2020 Ampere, Mellanox & ARM

GPU计算性能变迁



NVIDIA

GPU Computing Applications

Libraries and Middleware

cuDNN TensorRT	cuFFT cuBLAS cuRAND cuSPARSE	CULA MAGMA	Thrust NPP	VSIPL SVM OpenCurrent	PhysX OptiX iRay	MATLAB Mathematica

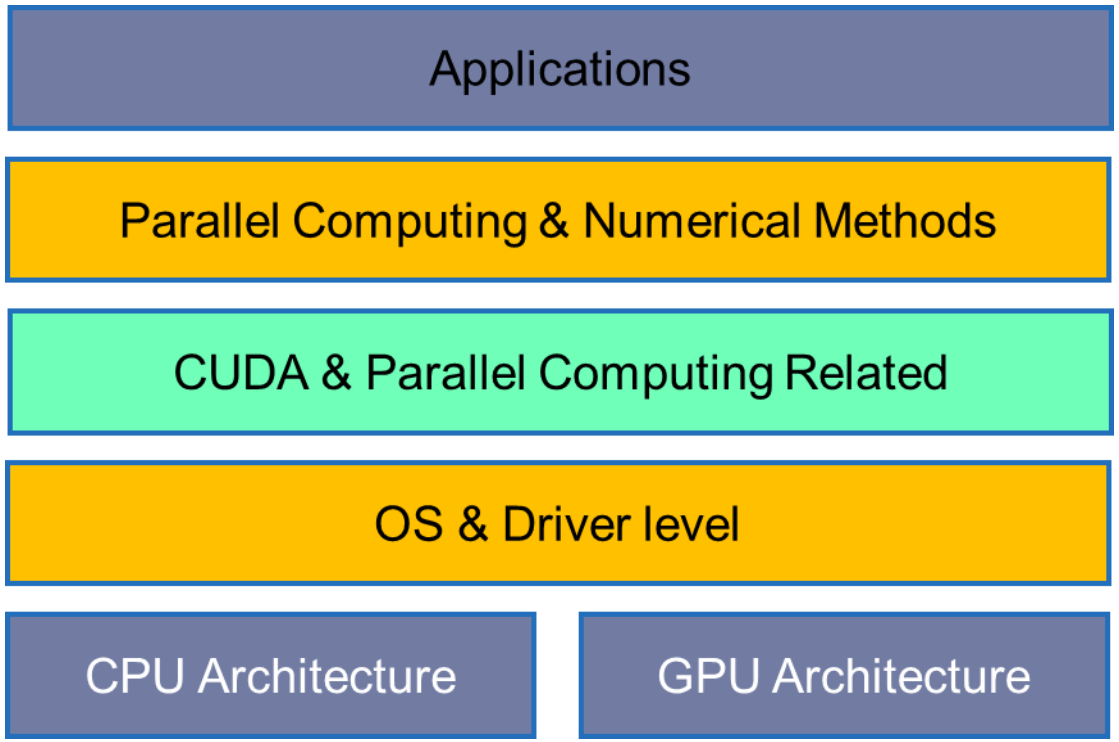
Programming Languages

C	C++	Fortran	Java Python Wrappers	DirectCompute	Directives (e.g. OpenACC)

CUDA-Enabled NVIDIA GPUs

NVIDIA Ampere Architecture (compute capabilities 8.x)				Tesla A Series
NVIDIA Turing Architecture (compute capabilities 7.x)		GeForce 2000 Series	Quadro RTX Series	Tesla T Series
NVIDIA Volta Architecture (compute capabilities 7.x)	DRIVE/JETSON AGX Xavier		Quadro GV Series	Tesla V Series
NVIDIA Pascal Architecture (compute capabilities 6.x)	Tegra X2	GeForce 1000 Series	Quadro P Series	Tesla P Series
				

NVIDIA.


Applications
Parallel Computing & Numerical Methods
CUDA & Parallel Computing Related
OS & Driver level
CPU Architecture
GPU Architecture

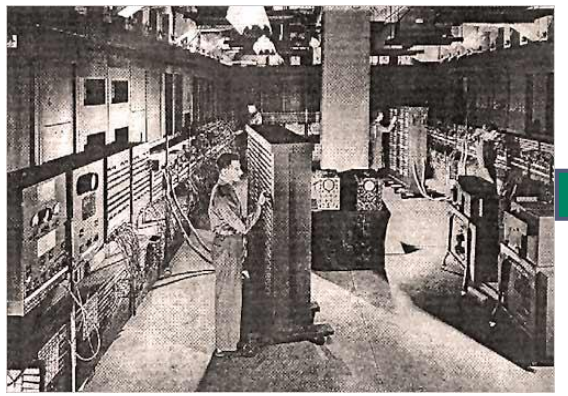
开始课程之前的问题....

1) 为什么我们要使用计算机?

为了更好地解决计算问题

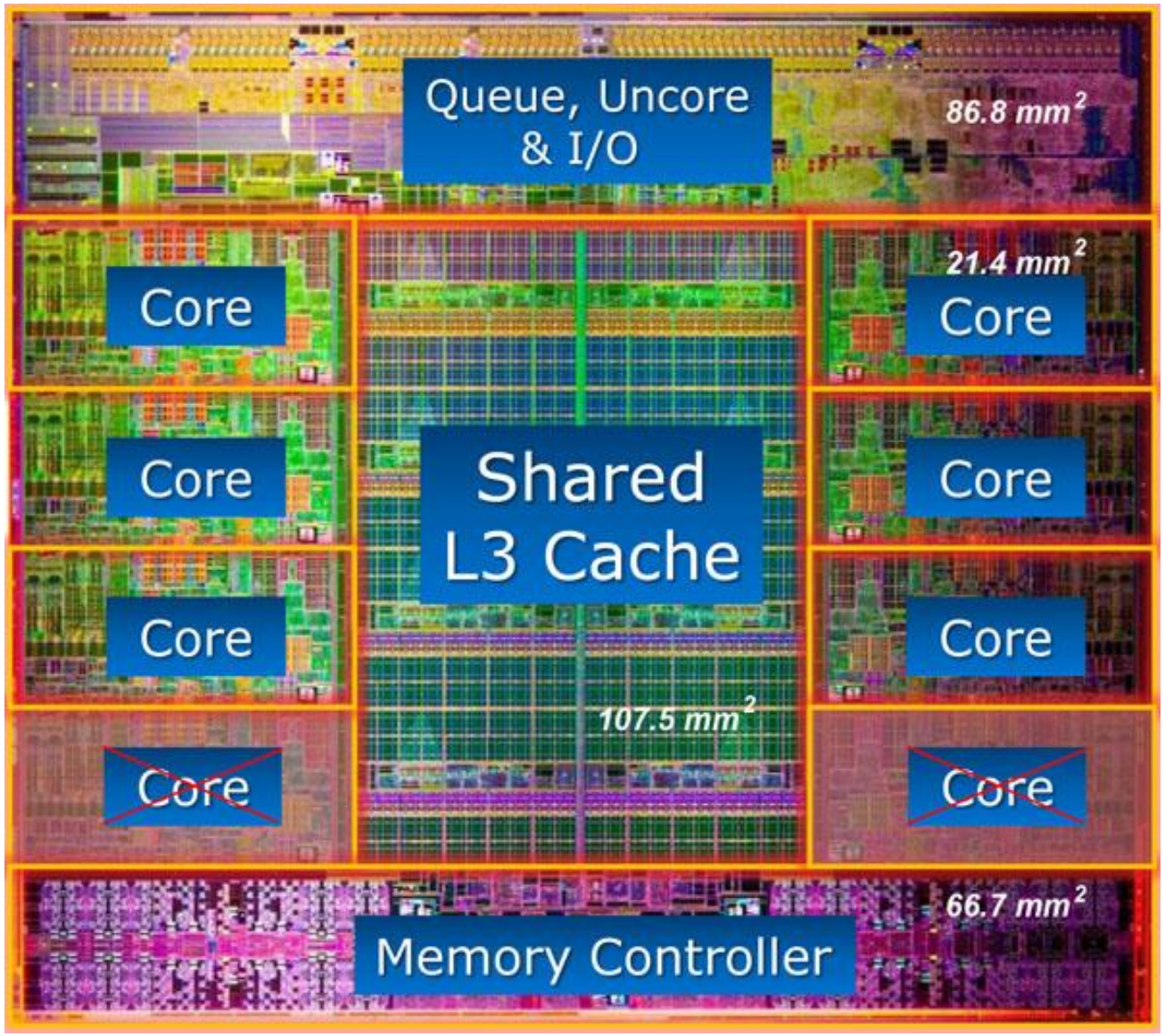
2) 你需要什么样的计算机? 畅想...

速度无穷快(1Phz?)
无穷多内存(1EB?)
智能化的接口
(人工智能)



NVIDIA

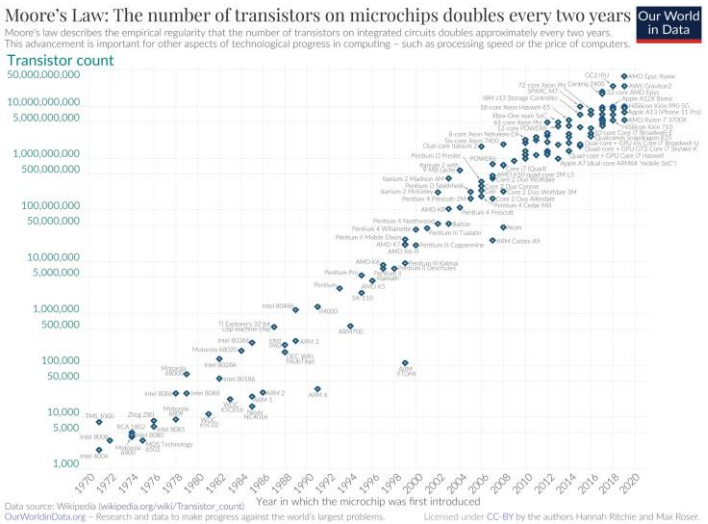
CPU架构

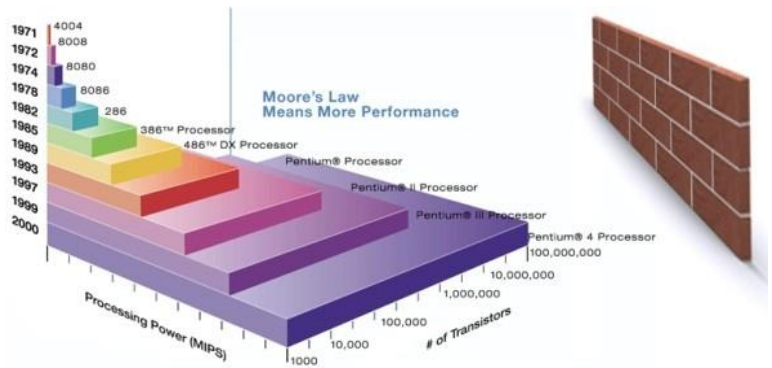

Queue, Uncore
& I/O
86.8 mm²
Core
21.4 mm²
Core
Core
Shared
L3 Cache
Core
Core
Core
107.5 mm²
Core
Core
Memory Controller
66.7 mm²

摩尔定律 MOORE'S LAW

芯片的集成密度每2年翻翻，成本下降一半.

"The complexity for minimum component costs has increased at a rate of roughly a factor of two per year"





intel.com

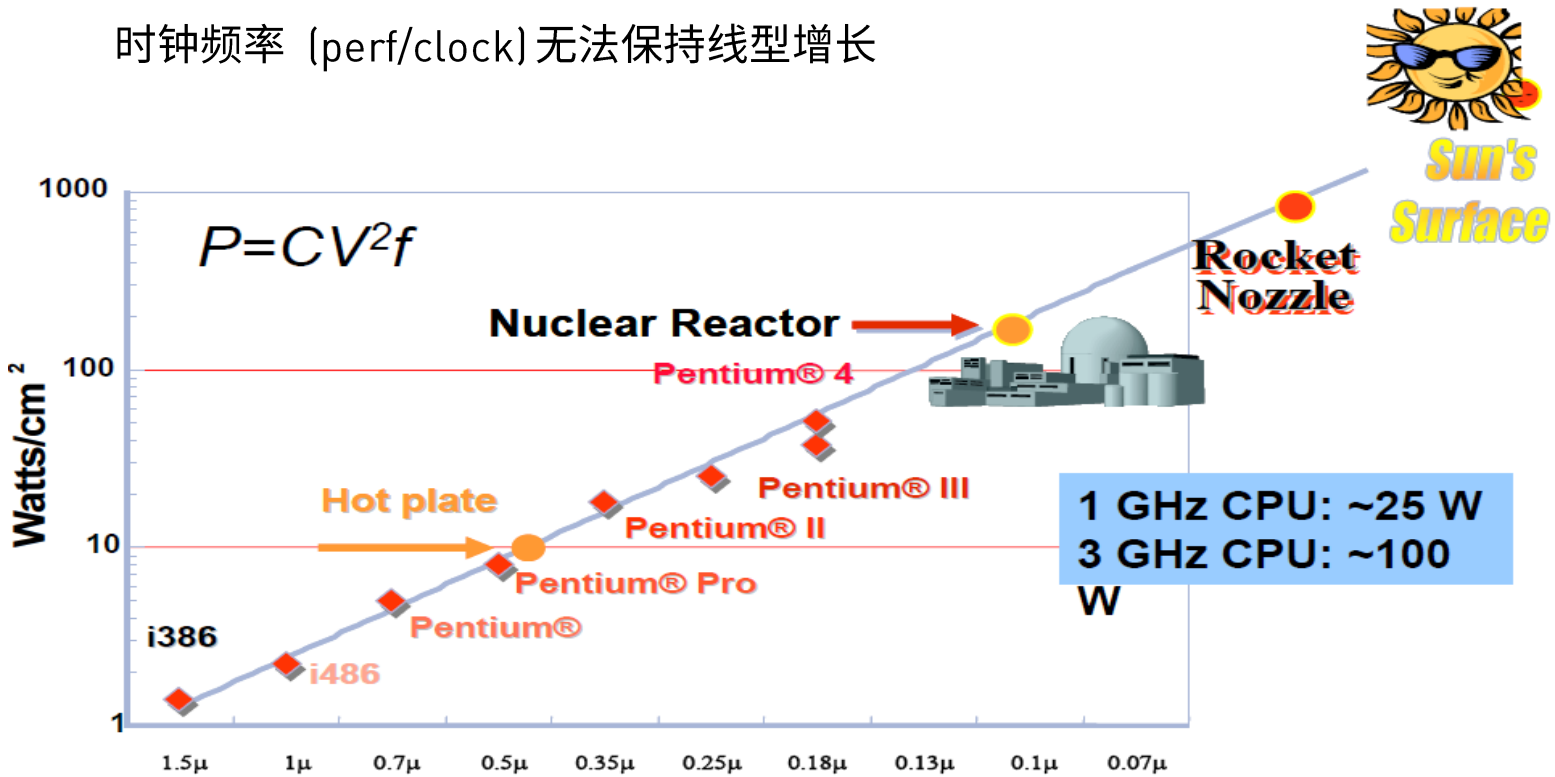
12

NVIDIA

可惜世间总是太多无奈…

常规传统单核处理器遇到物理约束

时钟频率（perf/clock）无法保持线型增长



时钟频率墙

存储器墙

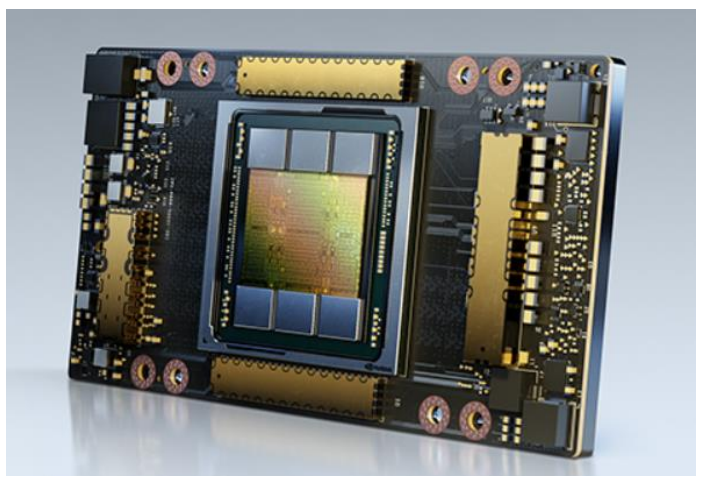
免费的午餐要消失了!!

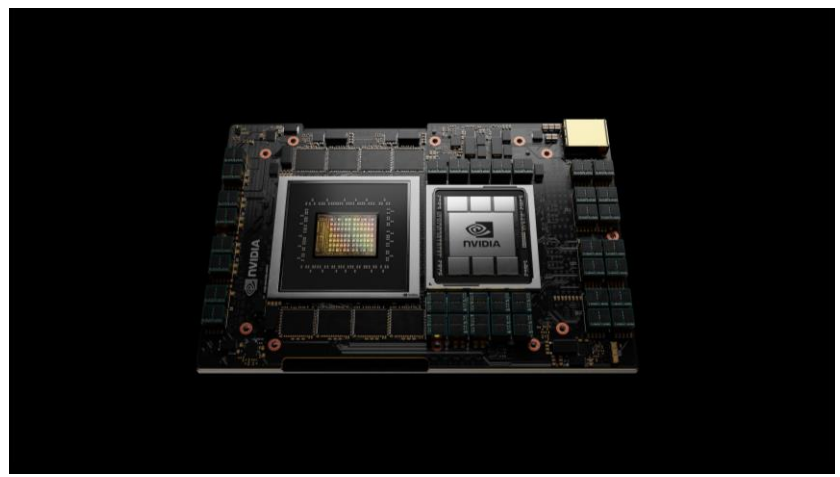
NVIDIA

现在的CPU系统已经遇到各种瓶颈

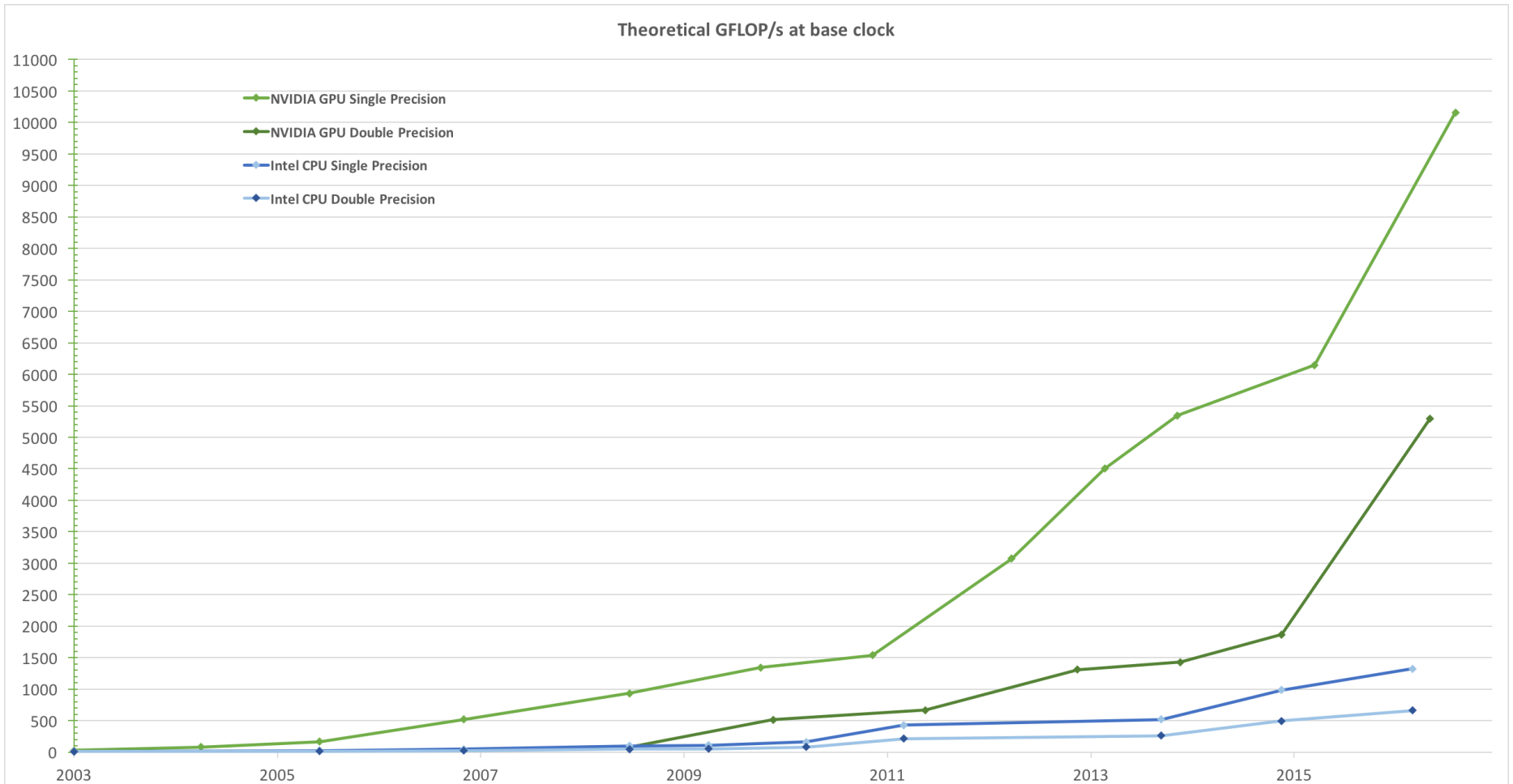
只能向多核及并行系统发展

顺势而生的GPU – Graphics Processing Unit



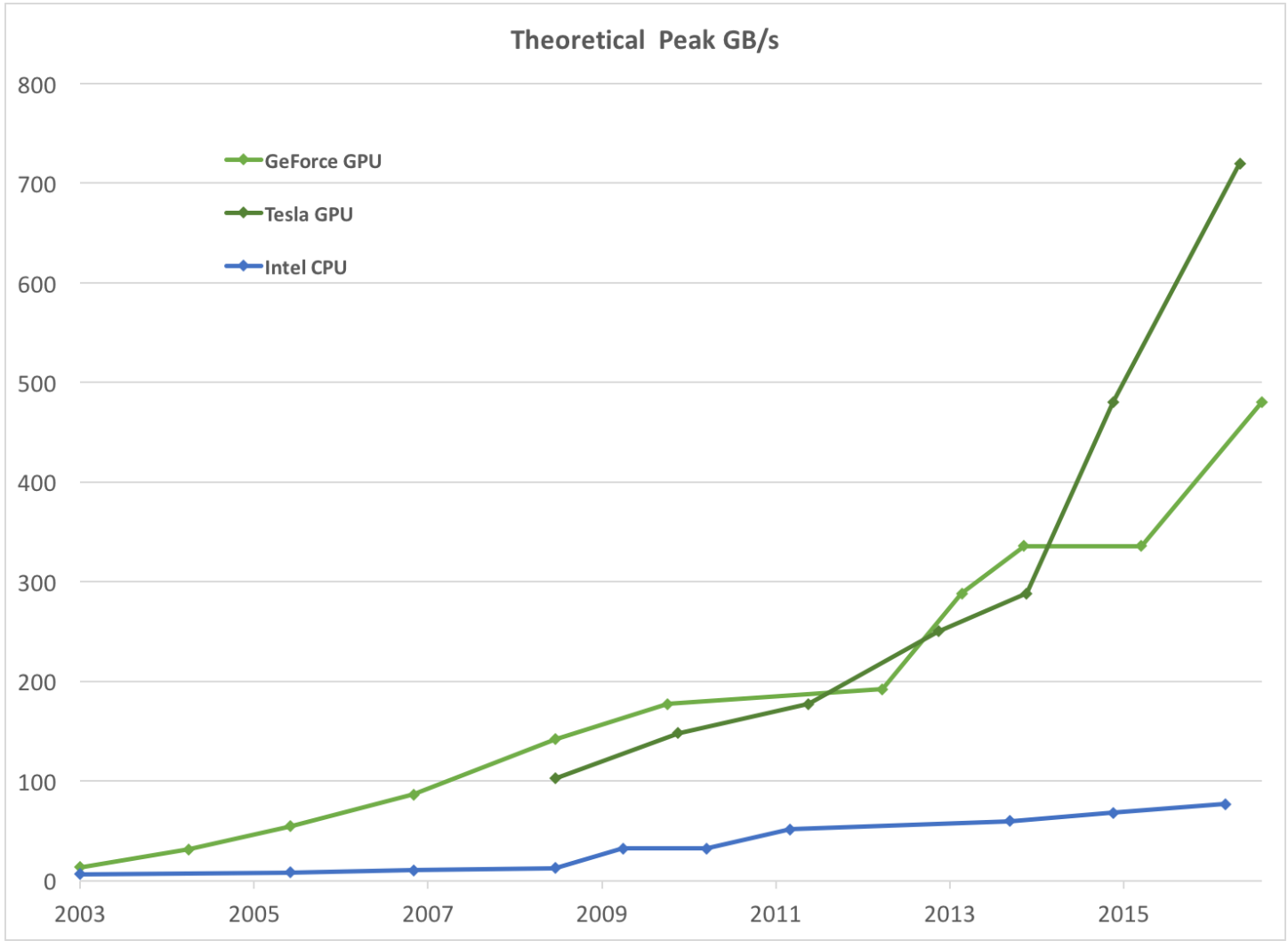


GPU计算性能



NVIDIA

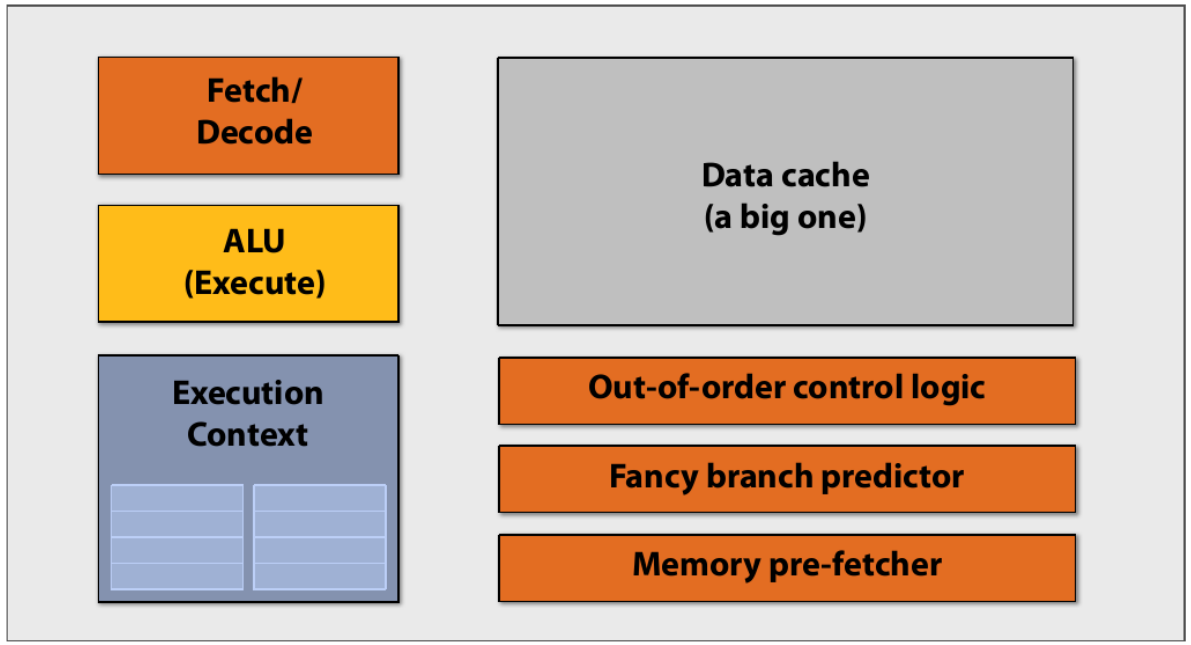
GPU数据传输带宽



NVIDIA

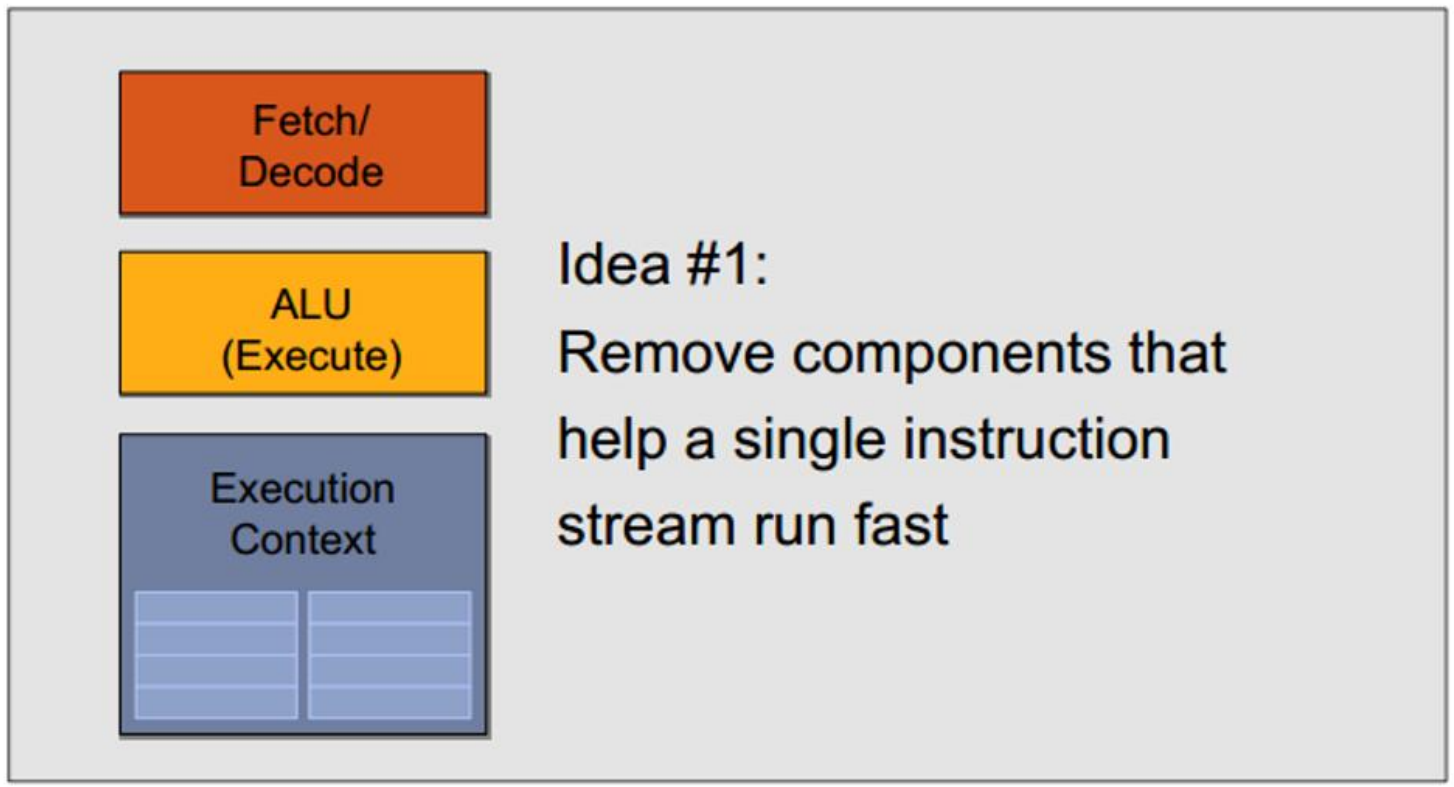
CPU类型的内核

“CPU-style” cores



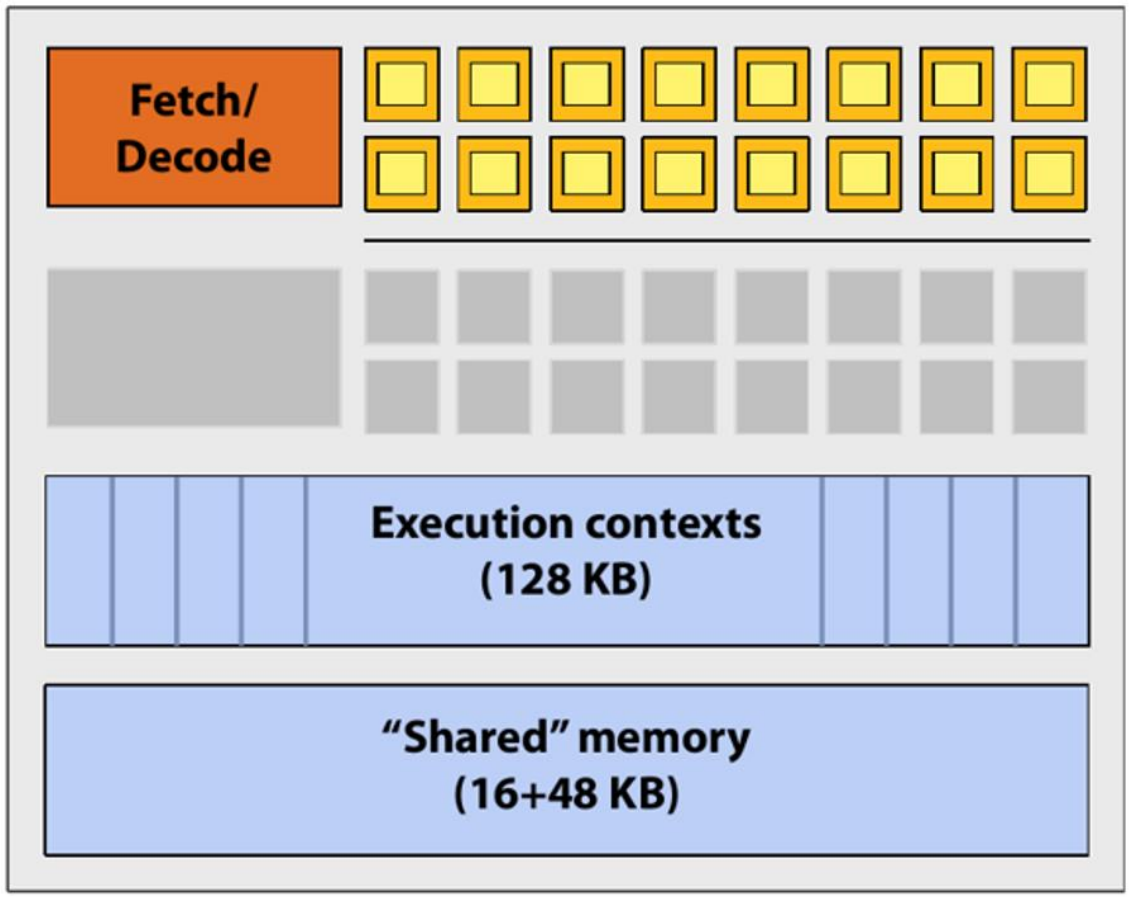
NVIDIA

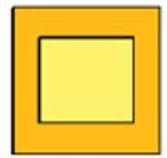
精简、减肥之后



NVIDIA

重新组合在一起!



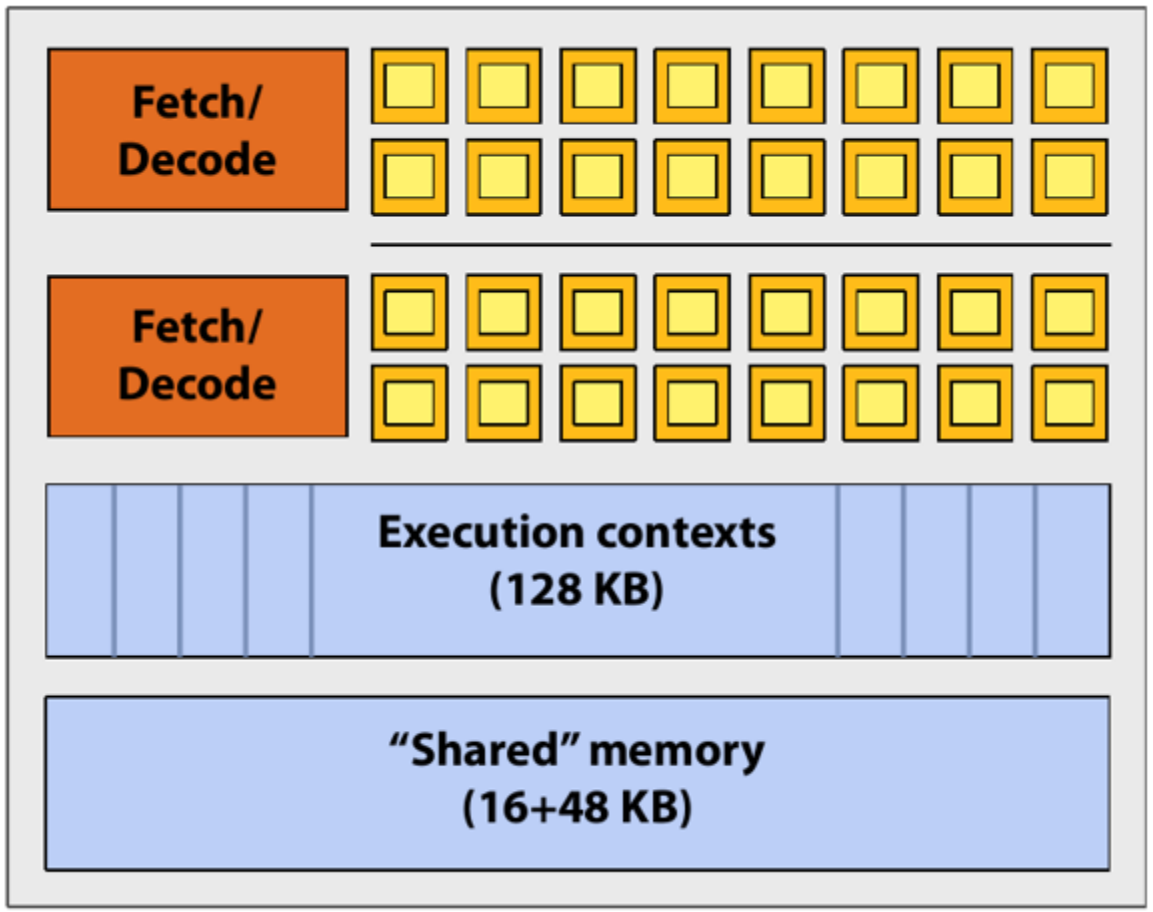


= SIMD function unit,
control shared across 16 units
(1 MUL-ADD per clock)

- Groups of 32 [fragments/vertices/CUDA threads] share an instruction stream
- Up to 48 groups are simultaneously interleaved
- Up to 1536 individual contexts can be stored

NVIDIA

CUDA CORE!

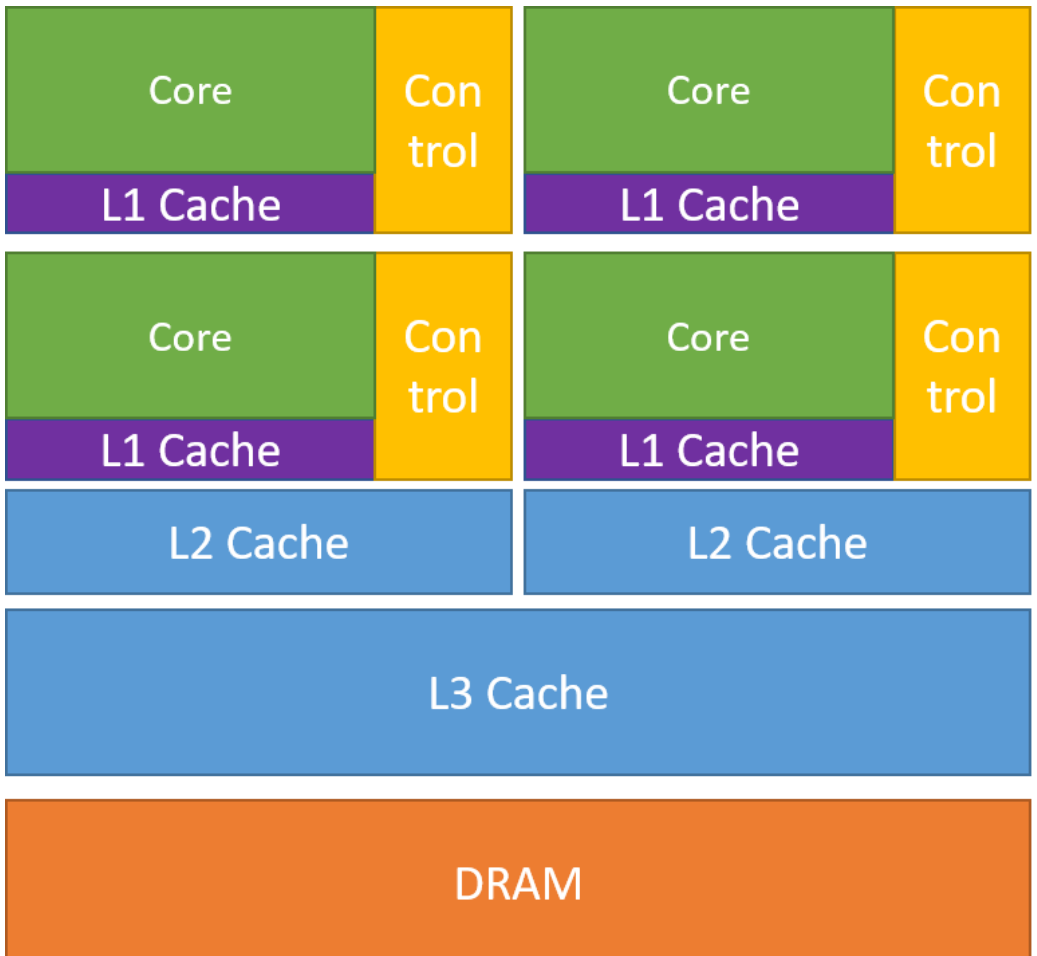


= **CUDA core**
(1 MUL-ADD per clock)

- The **SM** contains 32 **CUDA cores**
- Two **warps** are selected each clock (decode, fetch, and execute two **warps** in parallel)
- Up to 48 warps are interleaved, totaling 1536 **CUDA threads**

NVIDIA

芯片结构

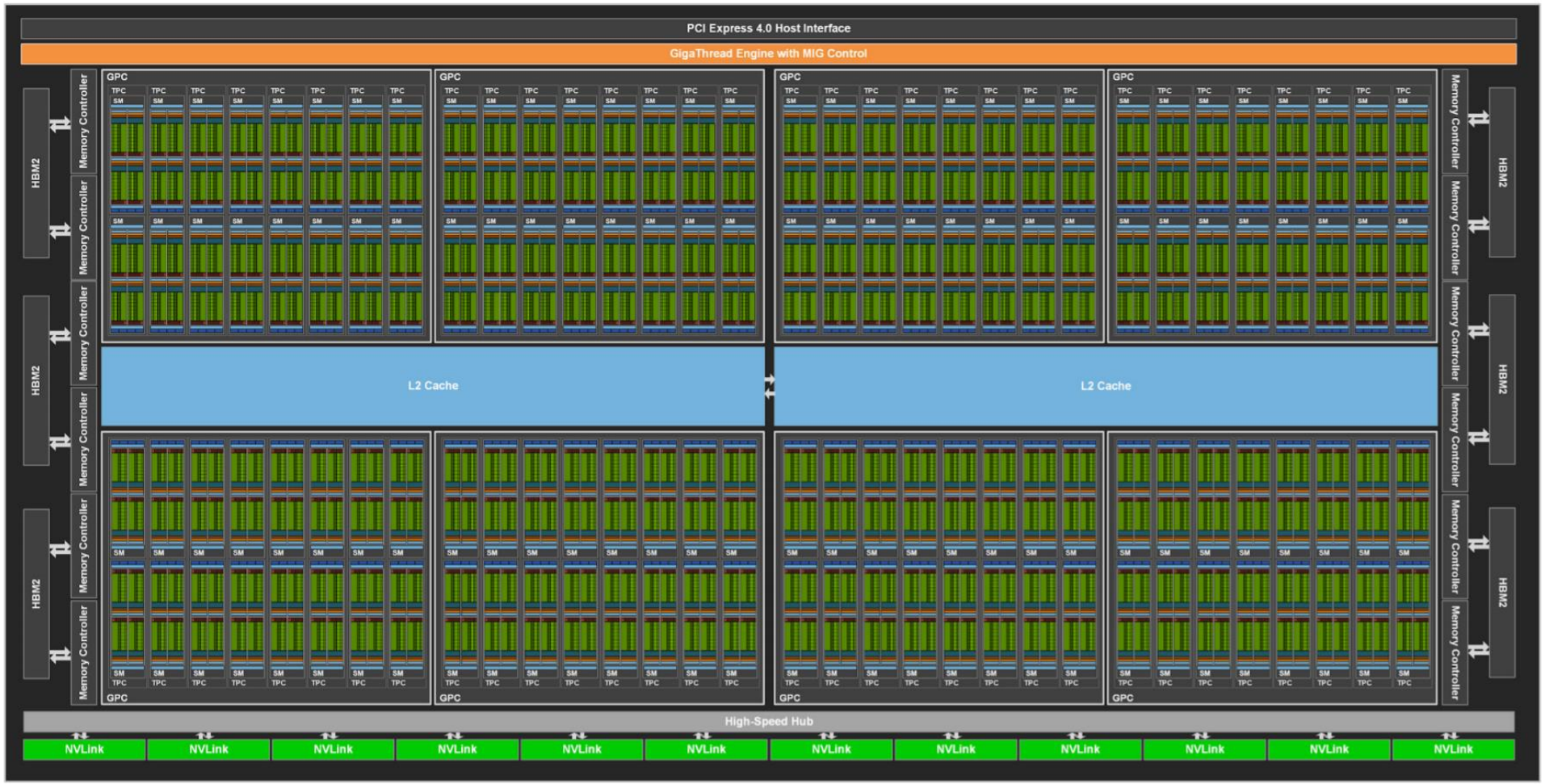


CPU

GPU

NVIDIA

GPU结构---GA100



NVIDIA

GPU结构---GA100

· 8 GPCs, 8 TPCs/GPC, 2 SMs/TPC, 16 SMs/GPC, 128 SMs per full GPU

· 64 FP32 CUDA Cores/SM, 8192 FP32 CUDA Cores per full GPU

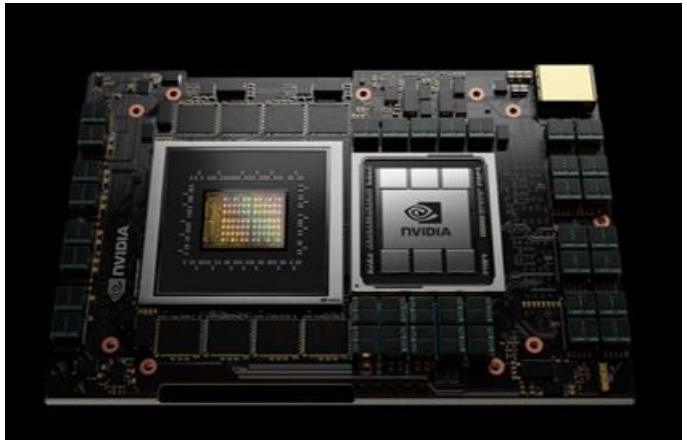
· 4 third-generation Tensor Cores/SM, 512 third-generation Tensor Cores per full GPU

· 6 HBM2 stacks, 12 512-bit memory controllers

NVIDIA.

JETSON NANO	
GPU	128 Core Maxwell 0.472 TFLOPs (FP16)
CPU	4 core ARM A57 @ 1.43 GHz
Memory	4 GB 64 bit LPDDR4 25.6 GB/s
Storage	16 GB eMMC
Video Encode	4K @ 30 | 4x 1080p @ 30 | 8x 720p @ 30 (H.264/H.265)
Video Decode	4K @ 60 | 2x 4K @ 30 | 8x 1080p @ 30 | 16x 720p @ 30 | (H.264/H.265)
Camera	12 (3x4 or 4x2) MIPI CSI-2 DPHY 1.1 lanes (1.5 Gbps)
WiFi/BT	Requires external chip
Display	HDMI 2.0 or DP1.2 | eDP 1.4 | DSI (1 x2) 2 simultaneous
UPHY	1 x1/2/4 PCIE 1 USB 3.0
SATA	None
Other I/Os	1xSDIO / 2xSPI / 3xI2C / UART / I2S / GPIOs
USB OTG	Not supported
Mechanical	69.6mm x 45mm 260 pin edge connector, No TTP

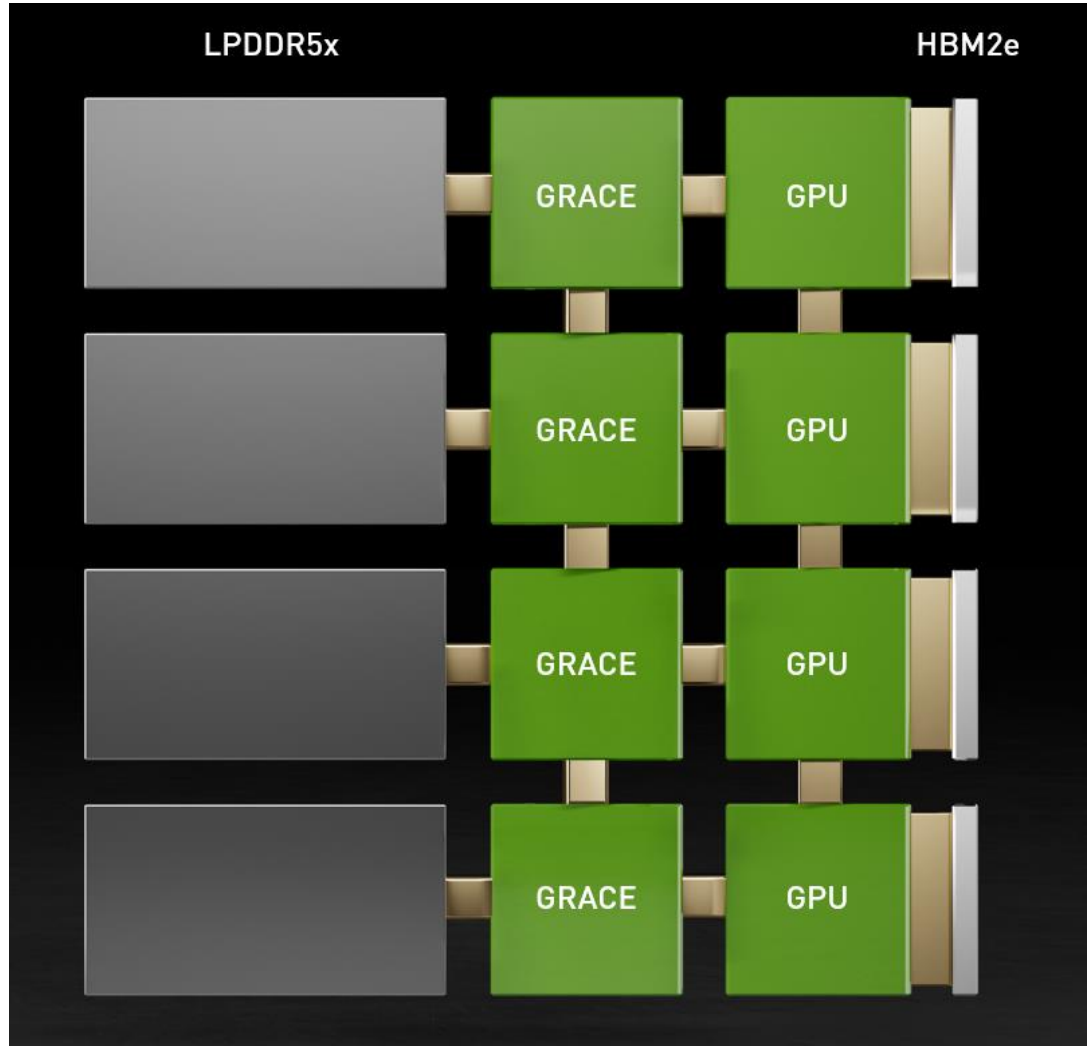
计算能力相当于多台台式机



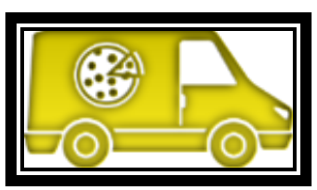
A NEW COMPUTING ARCHITECTURE FOR AI AND DATA SCIENCE

30X Increase System Memory to GPU

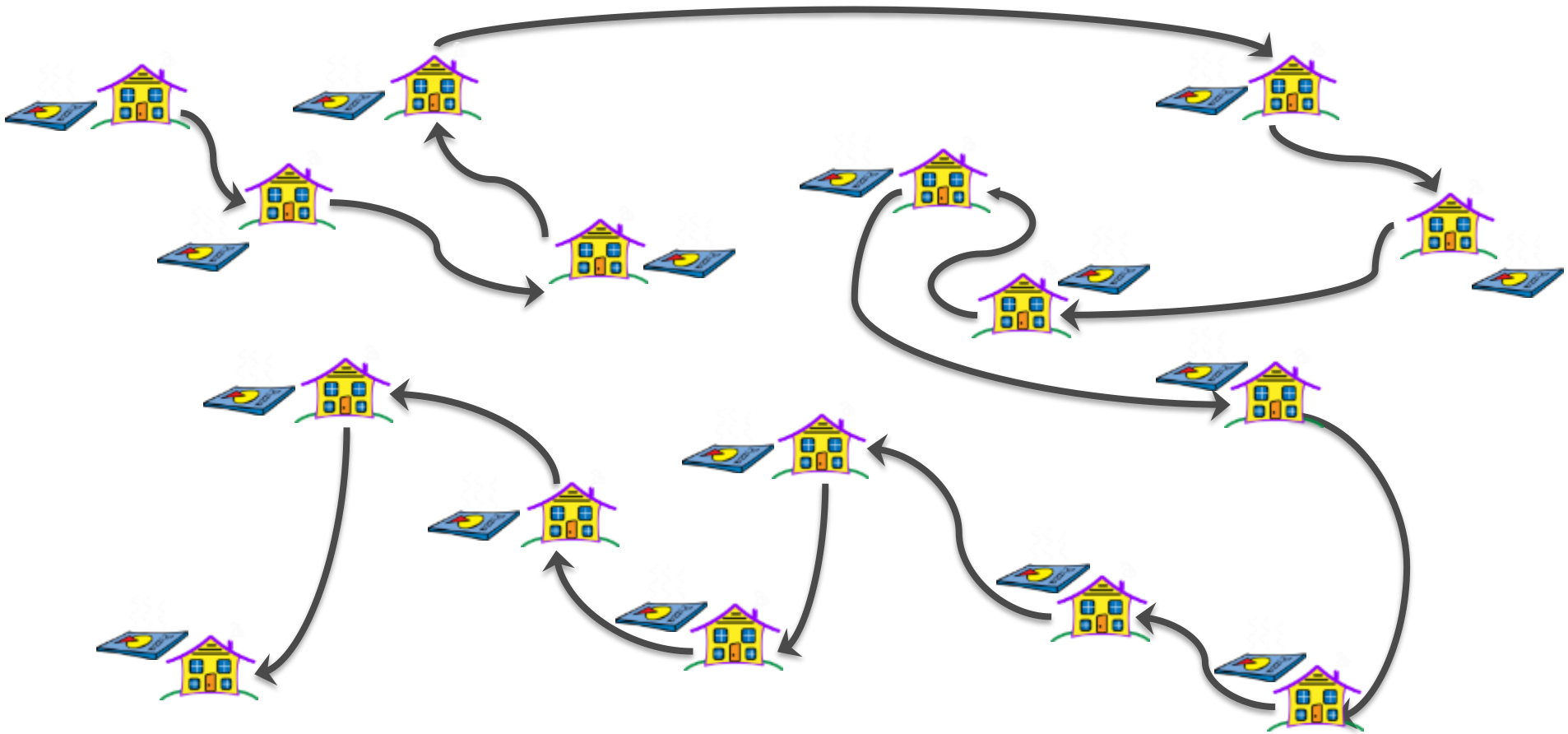
GPU	8,000	GB/sec	
CPU	500	GB/sec	
NVLINK	500	GB/sec	
Mem-to-GPU	2,000	GB/sec	30X



CPU 处理披萨快递



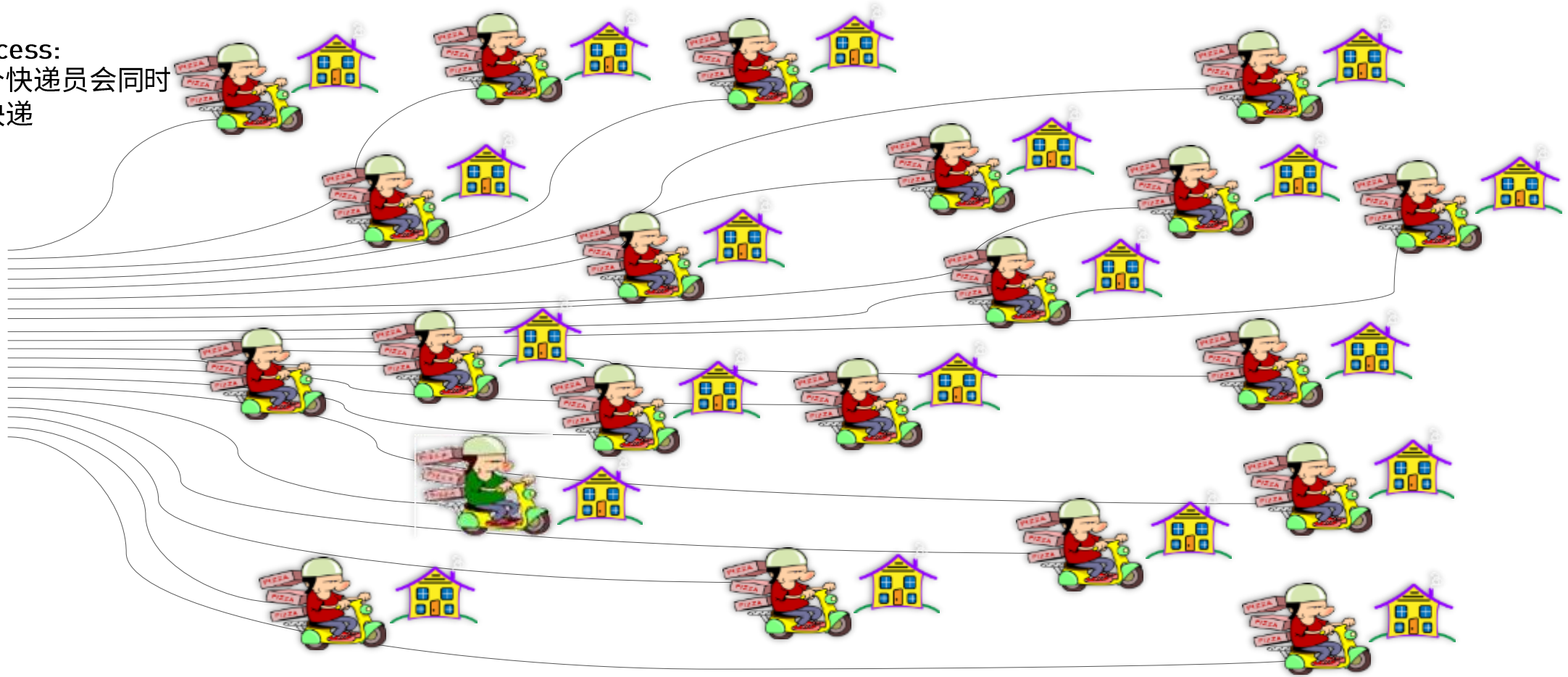
Process:
快递员会一家接着一家的送快递



NVIDIA.

NVIDIA GPU 披萨快递

Process:
多个快递员会同时送快递



NVIDIA

CUDA并行计算模式

- 一句话：并行计算是同时应用多个计算资源解决一个计算问题
  - 涉及多个计算资源或处理器
  - 问题被分解为多个离散的部分，可以同时处理（并行）
  - 每个部分可以由一系列指令完成
- 最好是计算密集的任务
  - 通信和计算开销比例合适
  - 不要受制于访存带宽

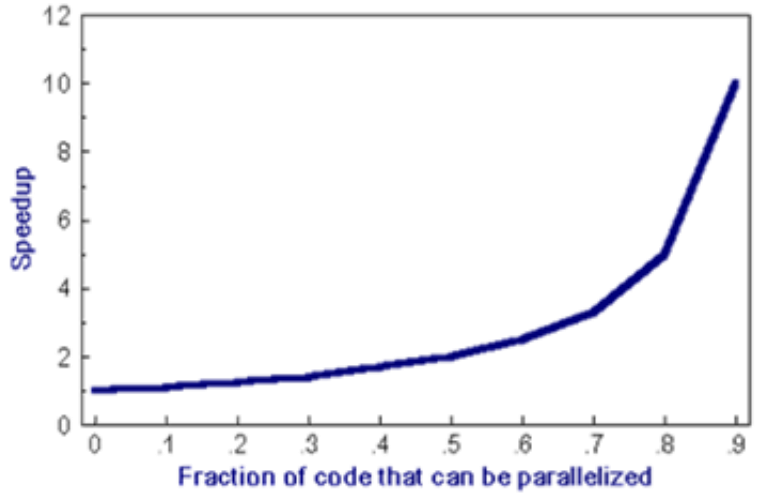
- 现在这些都能在基于ARM平台的Jetson NANO上完成！

NVIDIA.

Amdahl's Law

- Amdahl's Law 程序可能的加速比取决于可以被并行化的部分。

$$speedup = \frac{1}{1 - P}$$



- 如果没有可以并行化的, P = 0 and the speedup = 1 (no speedup).如果全部都可以并行化, P = 1 and the speedup is infinite (in theory).
- 如果50% 可以并行化, maximum speedup = 2,

Amdahl's Law

- **如果有N个处理器并行处理**

$$\text{speedup} = \frac{1}{\frac{P + S}{N}}$$

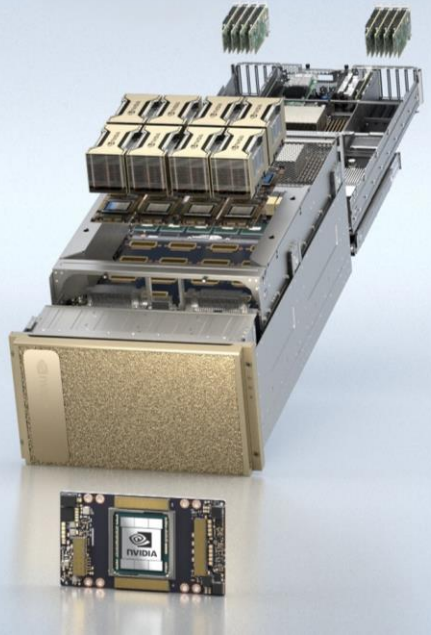
- P = **并行部分**, N = **处理器数量** and S = **串行部分**

Amdahl's Law

- 并行化的可扩展性有极限. For example, at P = .50, .90 and .99 (50%, 90% and 99% of the code is parallelizable)

	speedup		
N	P = .50	P = .90	P = .99
10	1.82	5.26	9.17
100	1.98	9.17	50.25
1000	1.99	9.91	90.99
10000	1.99	9.91	99.02

“从发明GPU来加速游戏，到把GPU改造成我们所见过的最多样化和最强大的协处理器，这是一条漫长的道路”



英伟达开创了加速计算的先机，以解决普通电脑无法解决的问题。我们为我们这个时代的达芬奇和爱因斯坦制造电脑，这样他们就能看到并创造未来。

加速计算需要的不仅仅是一个强大的芯片。我们通过全栈发明——从芯片和系统到它们运行的算法和应用程序——实现了令人难以置信的加速。

更多资源:

https://developer.nvidia-china.com

何琨-Ken

北京 密云

扫一扫上面的二维码图案，加我微信

https://www.nvidia.cn/developer/community-training/


NVIDIA