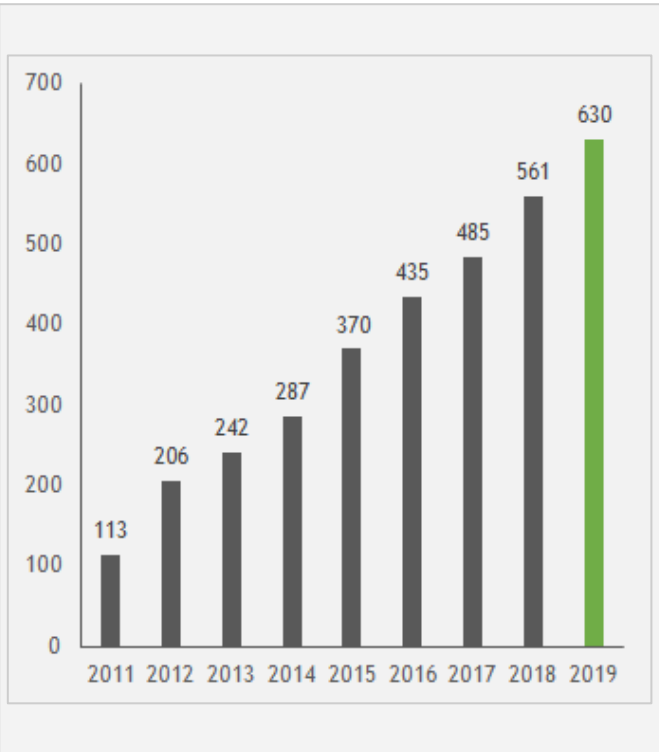
CUDA ON ARM PLATFORM—GPU硬件架构

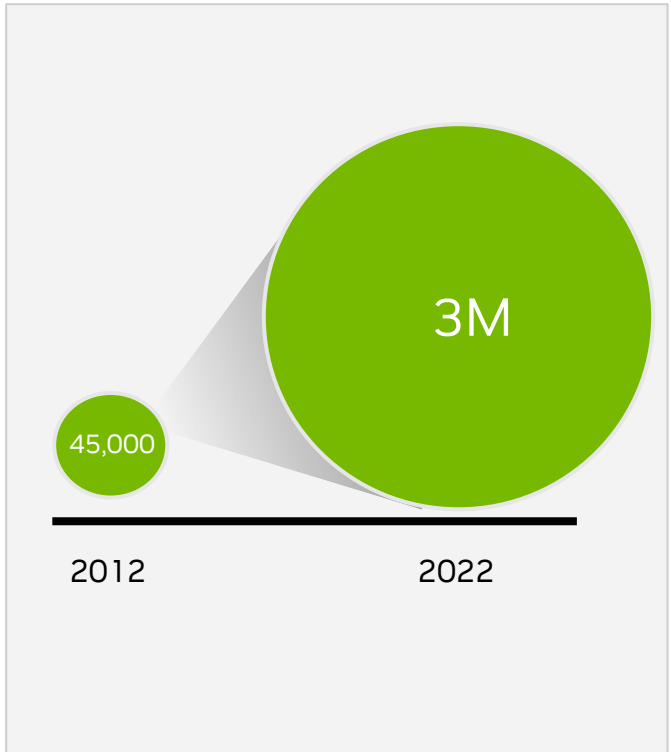
NVIDIA企业级开发者社区 何琨

为什么要学CUDA?

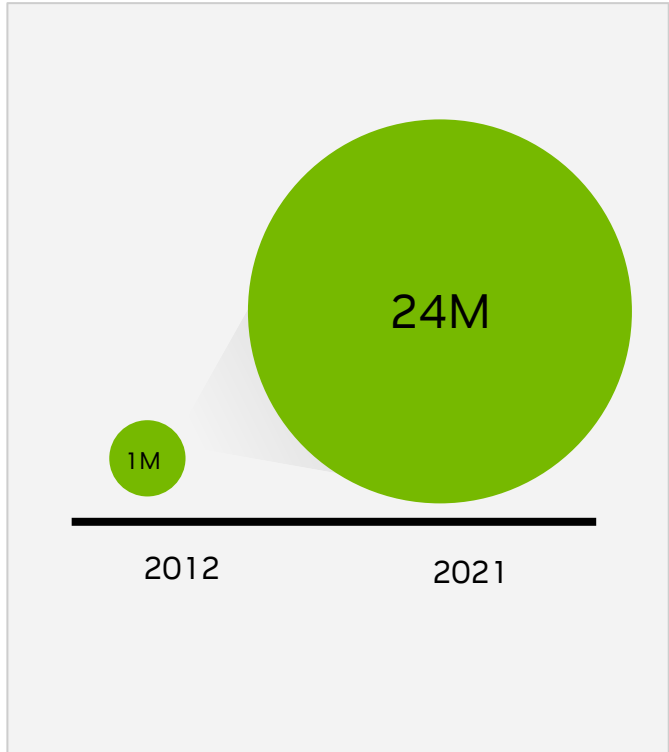
CUDA: 我为什么要学



630+ Applications Accelerated



62x GPU Developers



24X CUDA Downloads

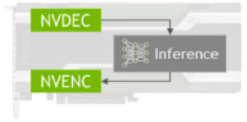
GPU ACCELERATED LIBRARIES

Acceleration for Your Applications

DEEP LEARNING

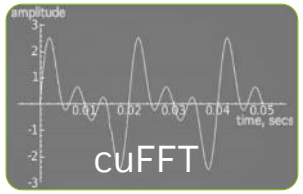
cuDNN


TensorRT

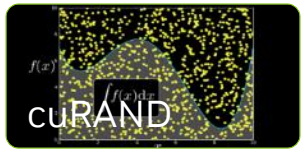


DeepStream SDK

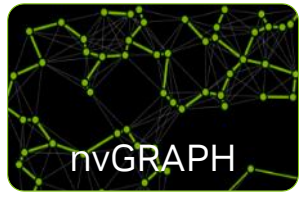
SIGNAL, IMAGE & VIDEO

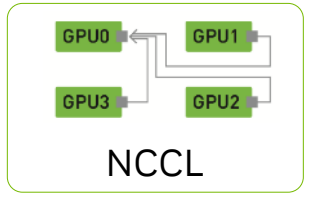


LINEAR ALGEBRA



PARALLEL ALGORITHMS





NCCL

课程说明

什么样的朋友适合来参加这个系列课程?

- 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

学习完这个系列的课程您可以收获什么?

- 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
- 十几个小时在Jetson Nano节点的实践
- 能够通过看API手册之类的，自行进步以从事GPU开发职业

- 风格：中英夹杂

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

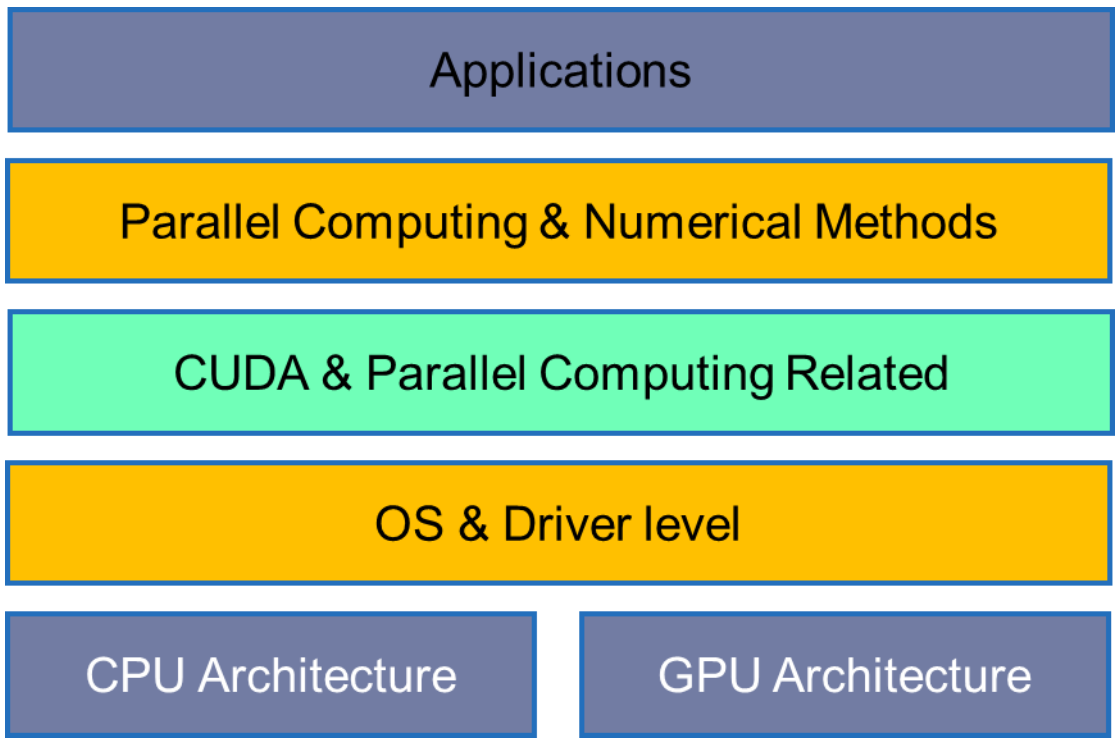
CUDA并行计算基础

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

What is CUDA?

- CUDA
  - Compute Unified Device Architecture
- CUDA C/C++
  - 基于C/C++的编程方法
  - 支持异构编程的扩展方法
  - 简单明了的APIs，能够轻松的管理存储系统
- CUDA支持的编程语言：
  - C/C++/Python/Fortran/Java/……
- 重点, 难点?


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

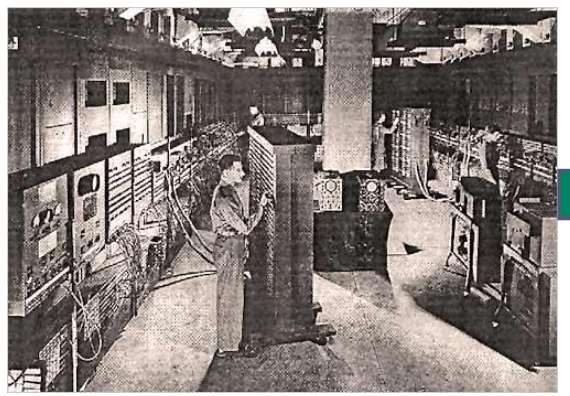
开始课程之前的问题....

1) 为什么我们要使用计算机?

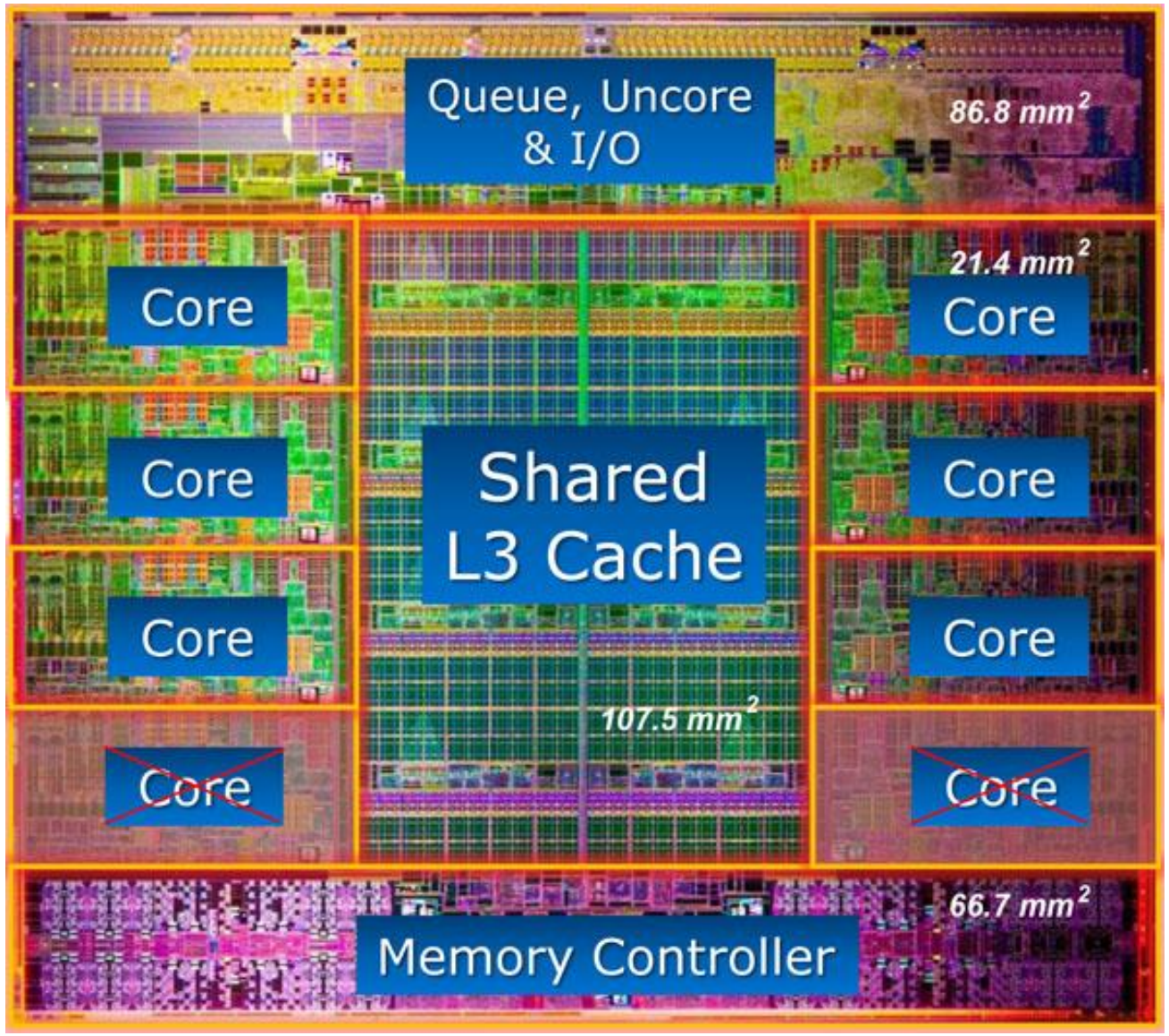
为了更好地解决计算问题

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)



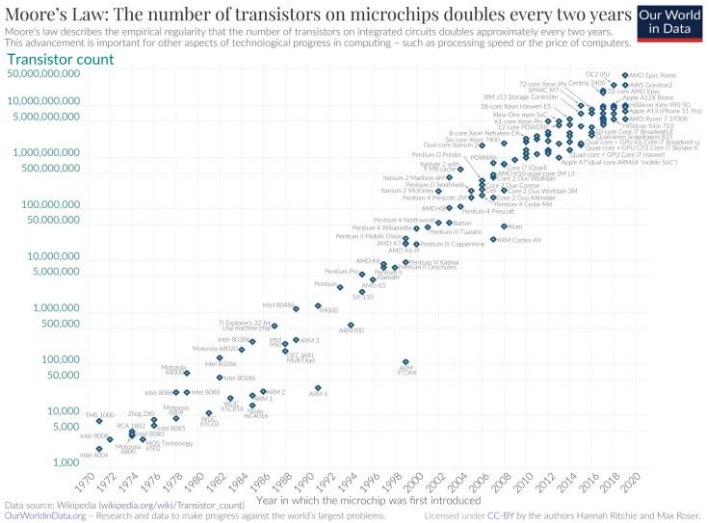
CPU架构

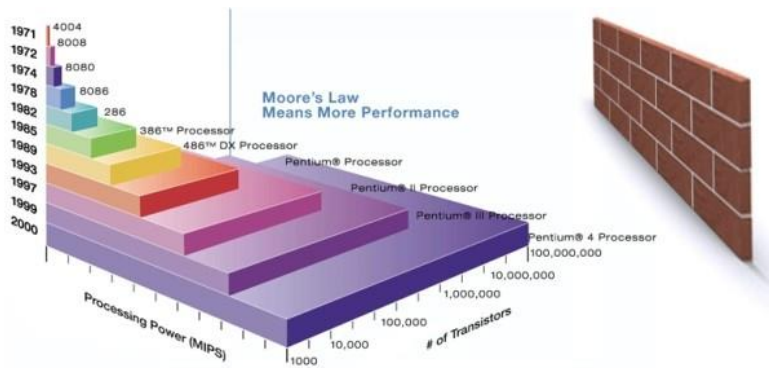

Queue, Uncore
& I/O
86.8 mm²
Core
21.4 mm²
Core
Core
Shared
L3 Cache
Core
Core
Core
107.5 mm²
Core
Core
Memory Controller
66.7 mm²

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”





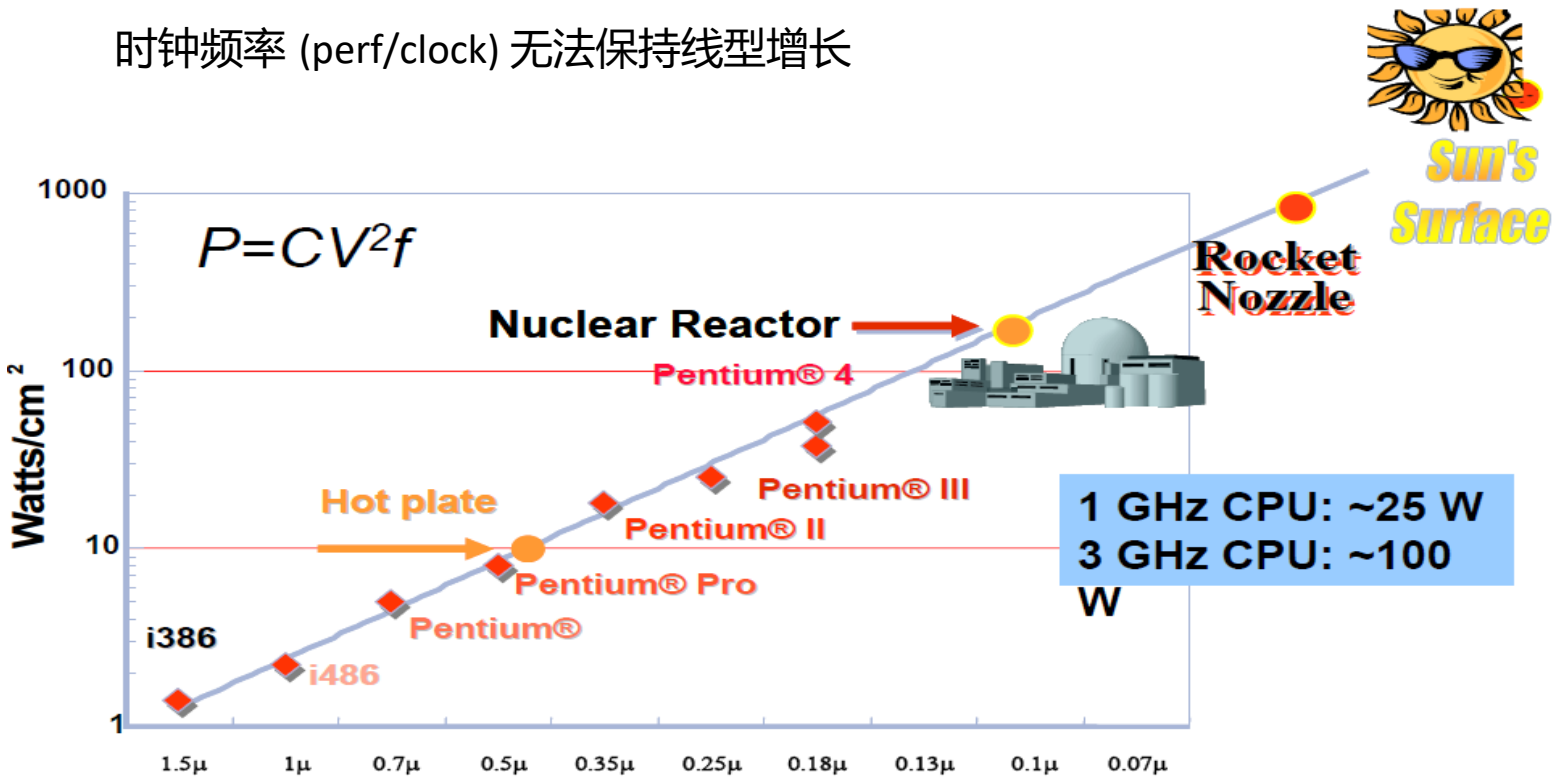
intel.com

12

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!!

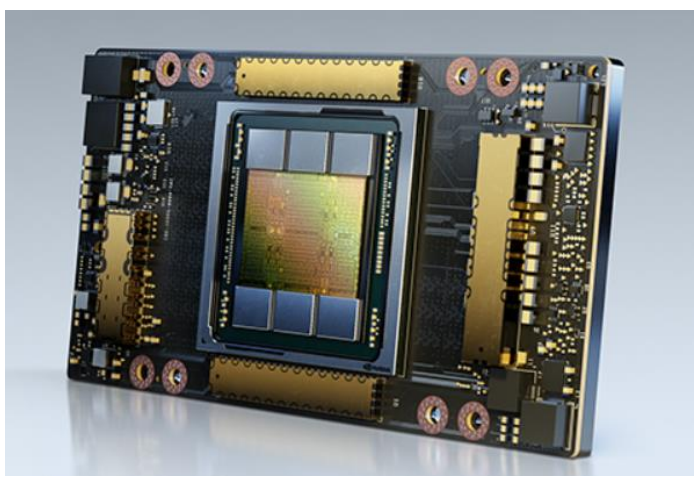
GPU VS CPU

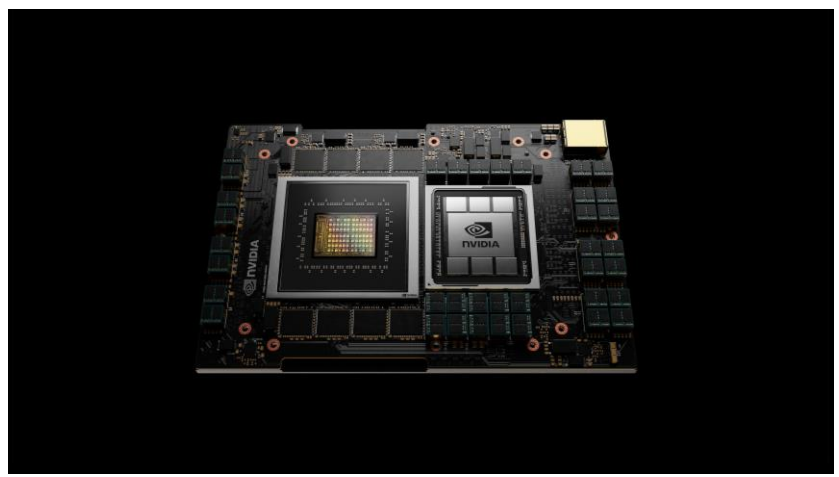
- CPU’s issue :
  - 大部分的能量都花在了控制上而不是计算上
  - X86 ISA 不够高效
- Wall :
  - Power Wall
  - Memory Wall
  - ILP Wall

现在的CPU系统已经遇到各种瓶颈

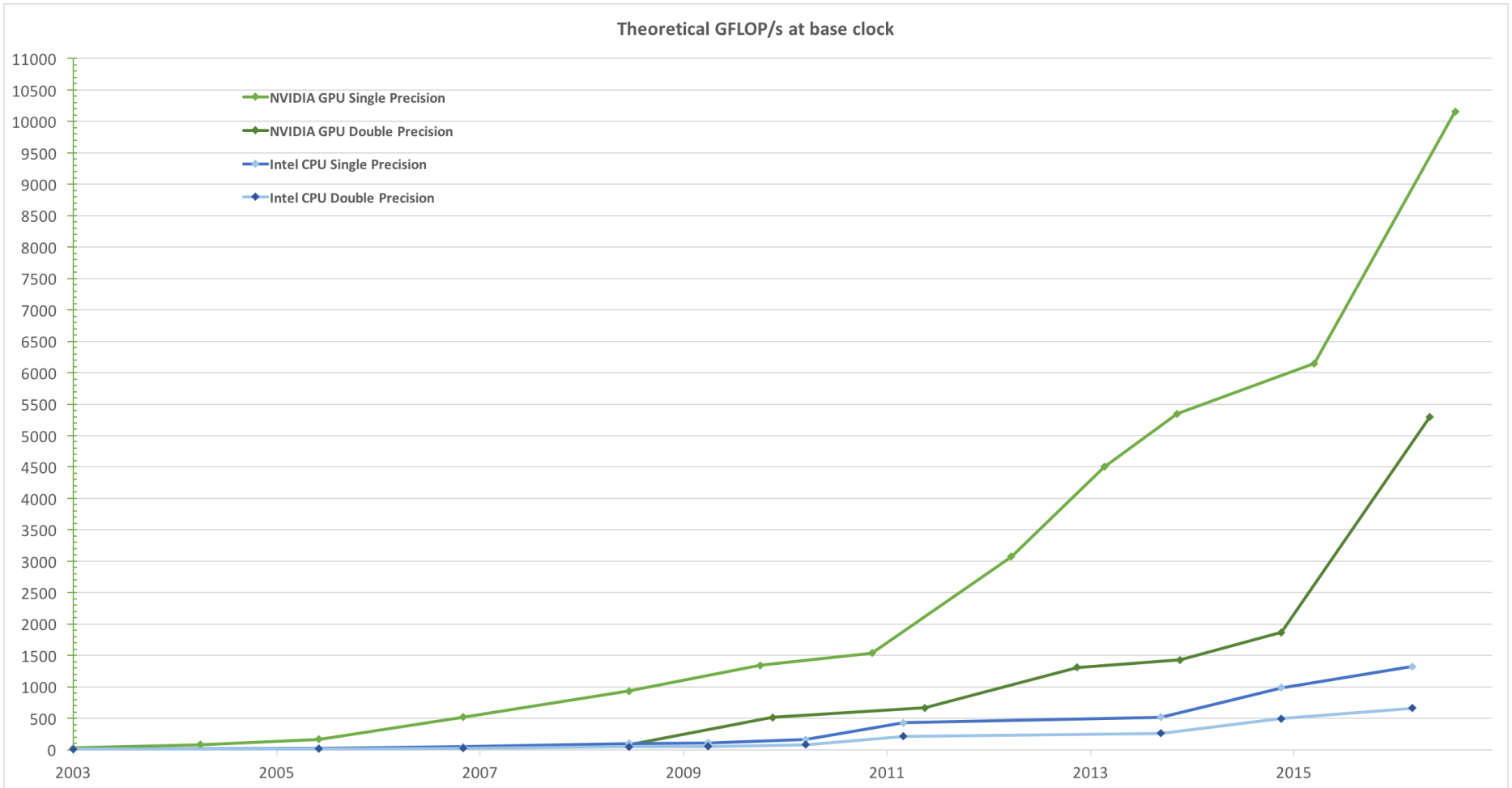
只能向多核及并行系统发展

顺势而生的GPU – Graphics Processing Unit



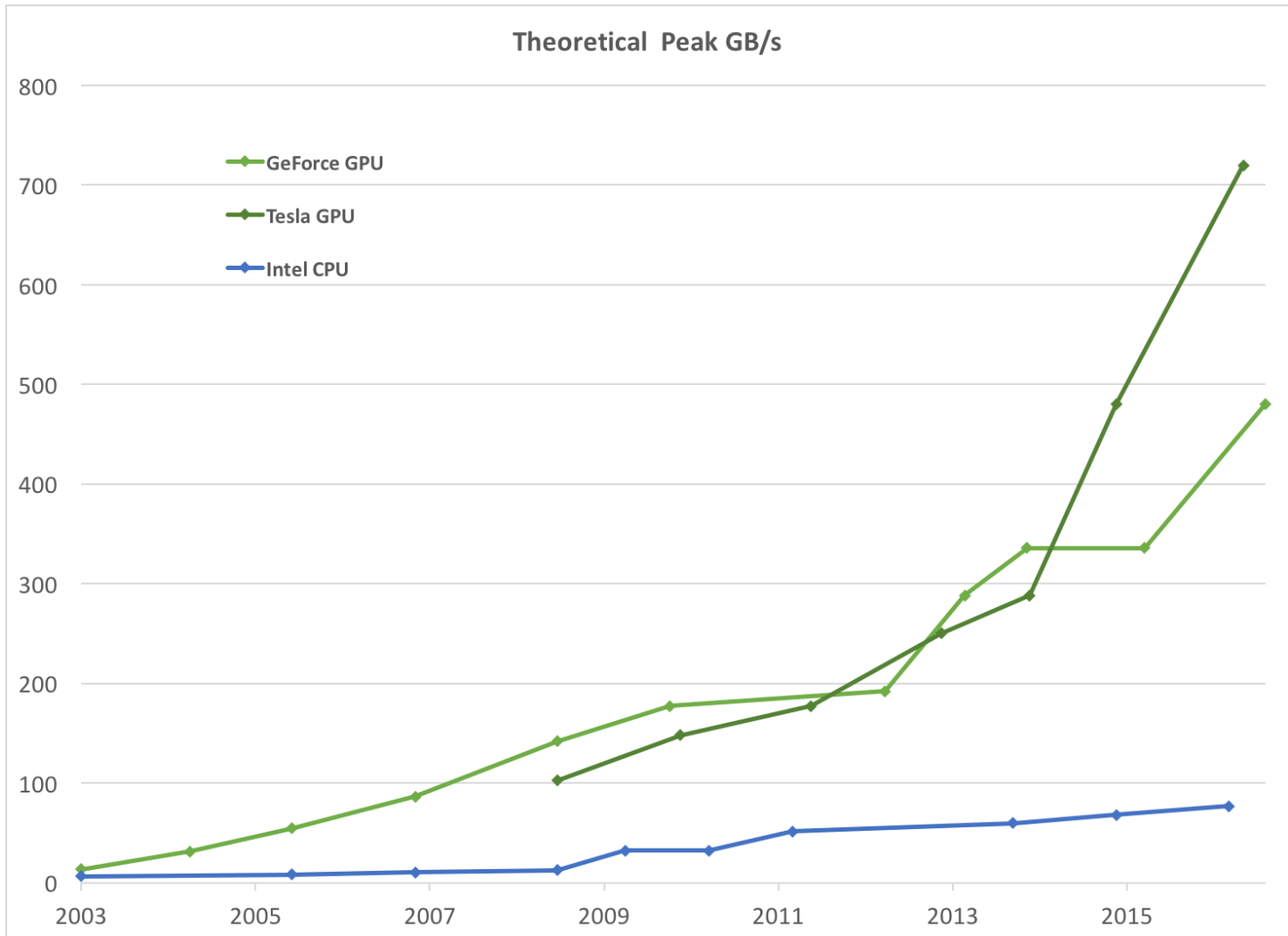


GPU计算性能



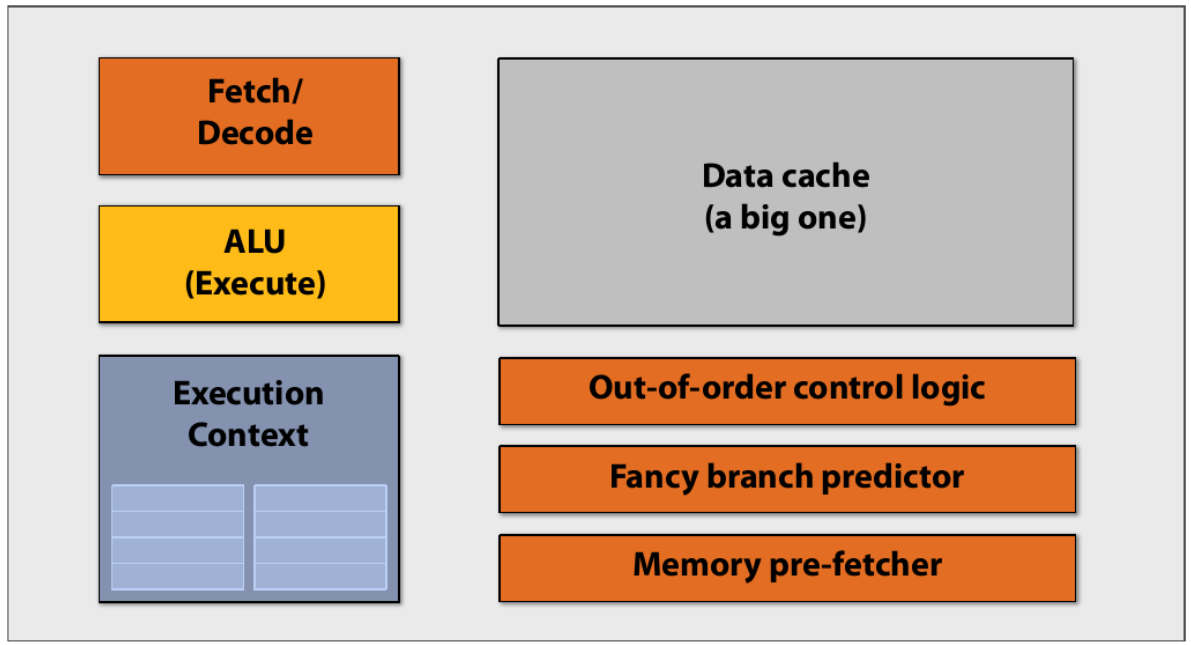
NVIDIA

GPU数据传输带宽

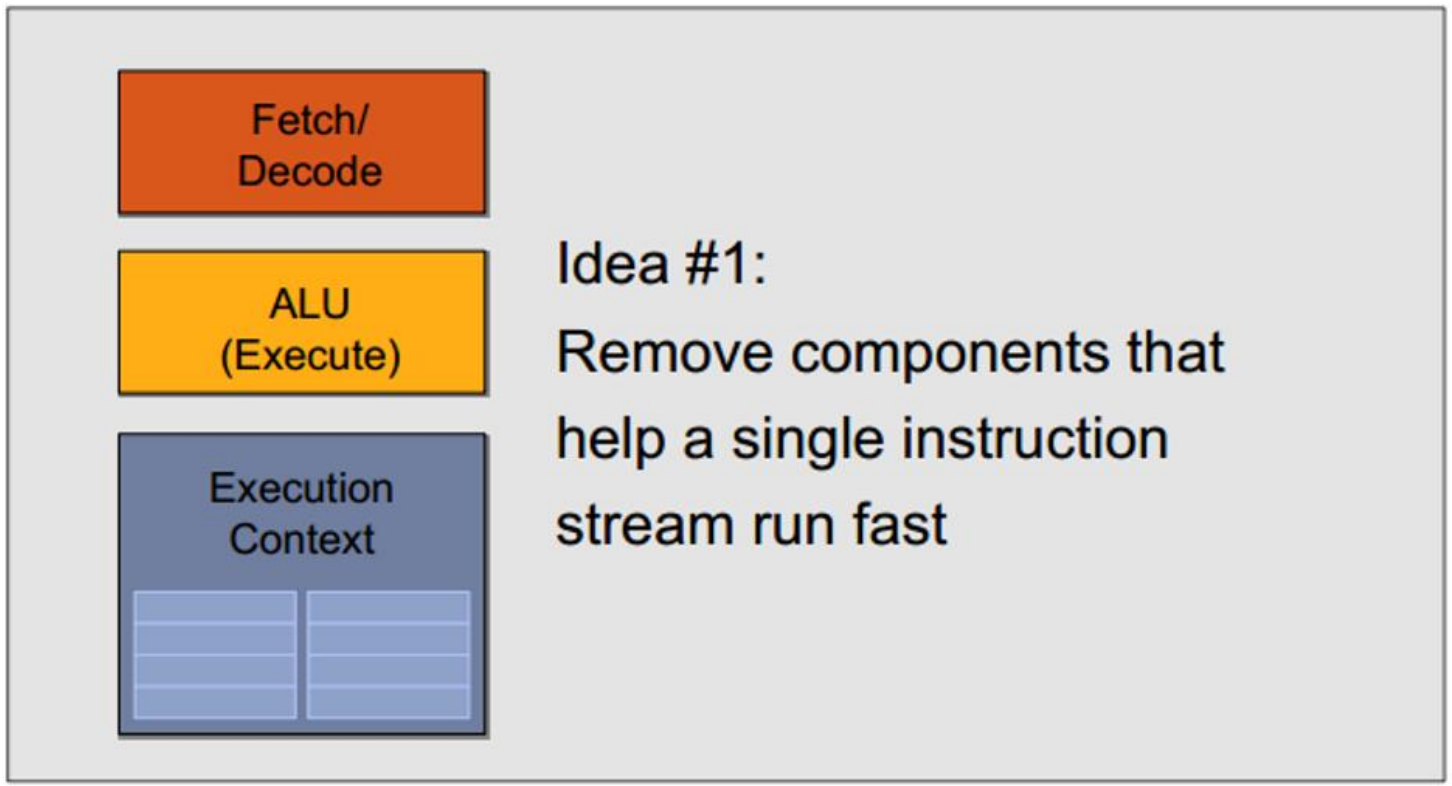


CPU类型的内核

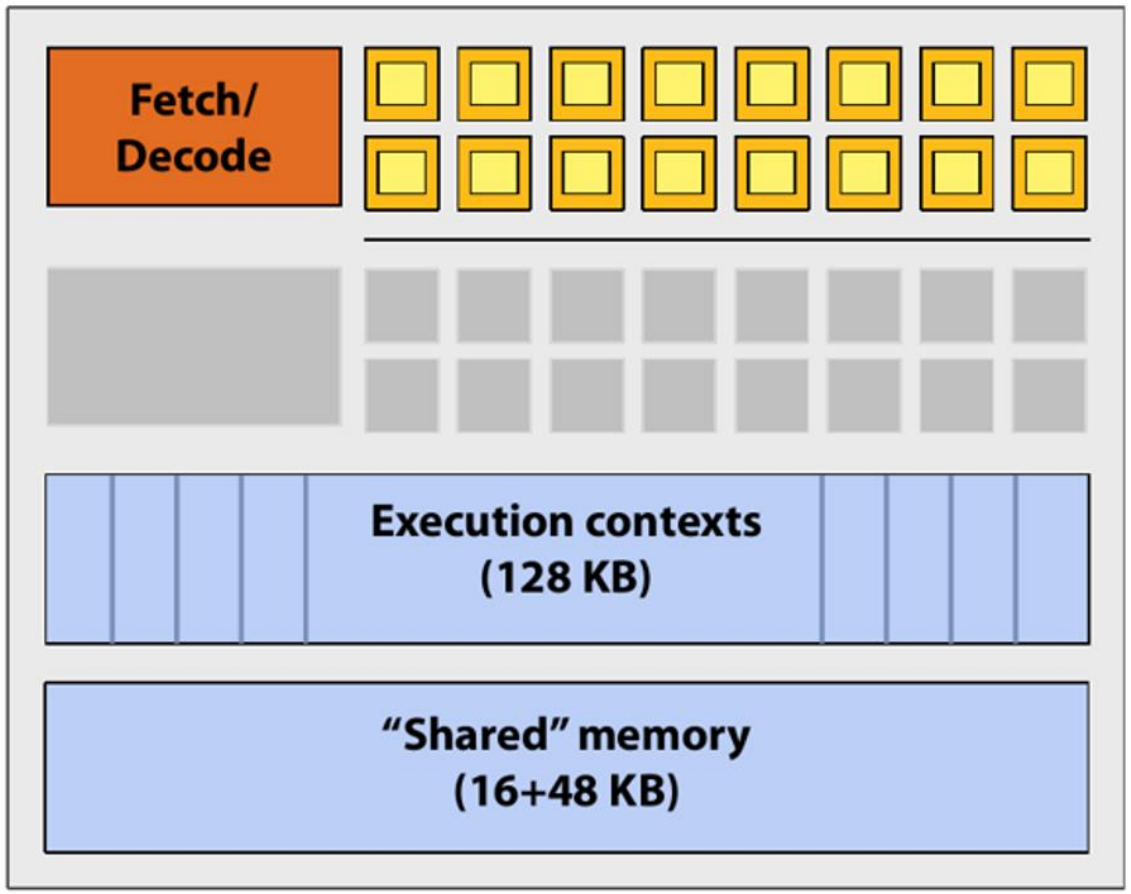
“CPU-style” cores

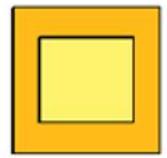


精简、减肥之后



重新组合在一起!

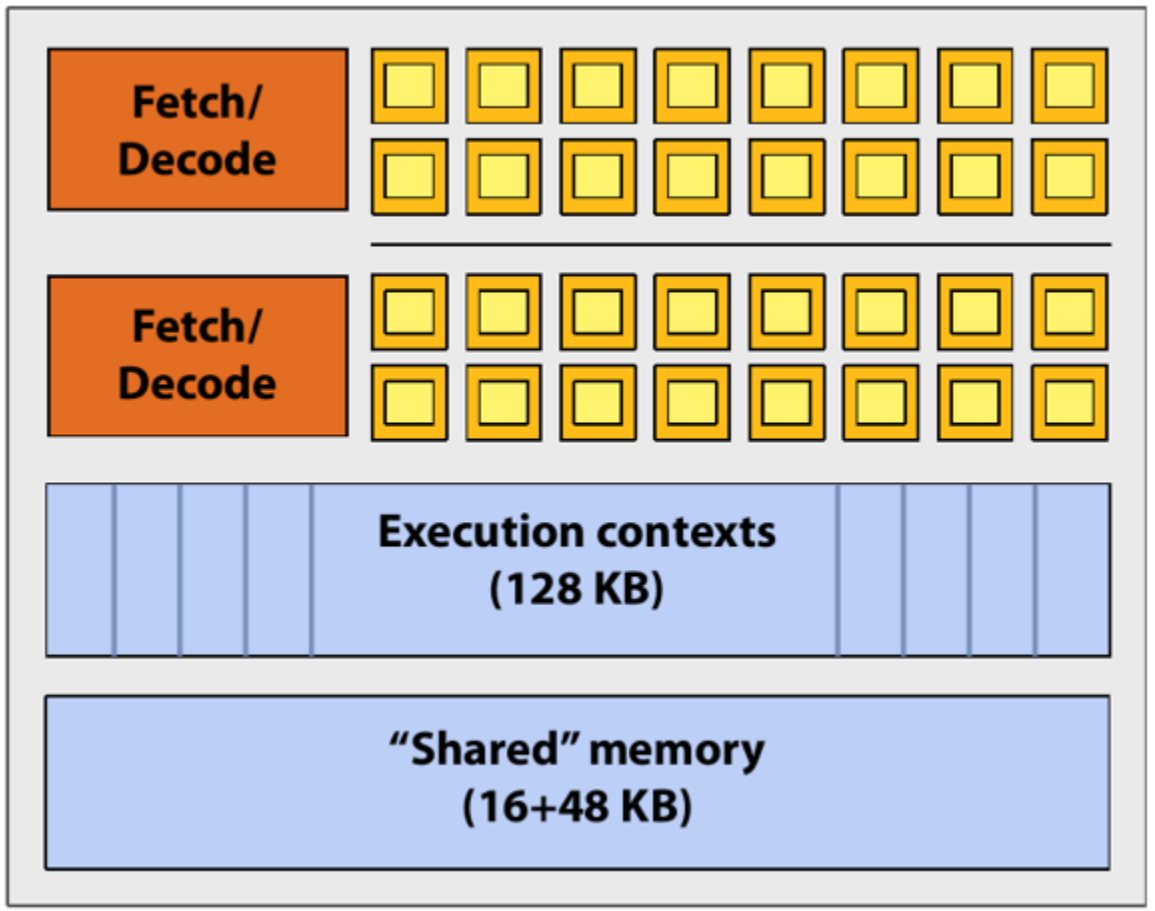




= SIMD function unit,
control shared across 16 units
(1 MUL-ADD per clock)

- 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

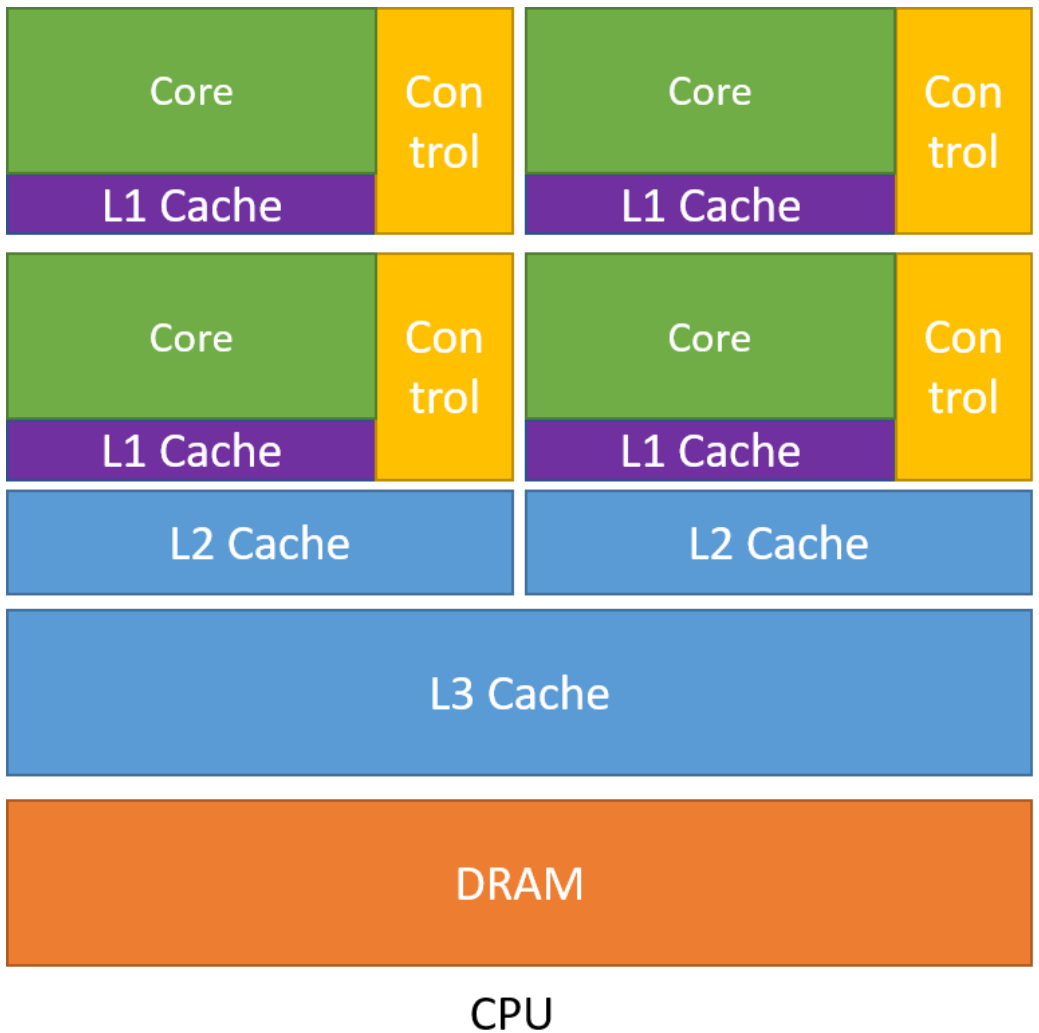
CUDA CORE!

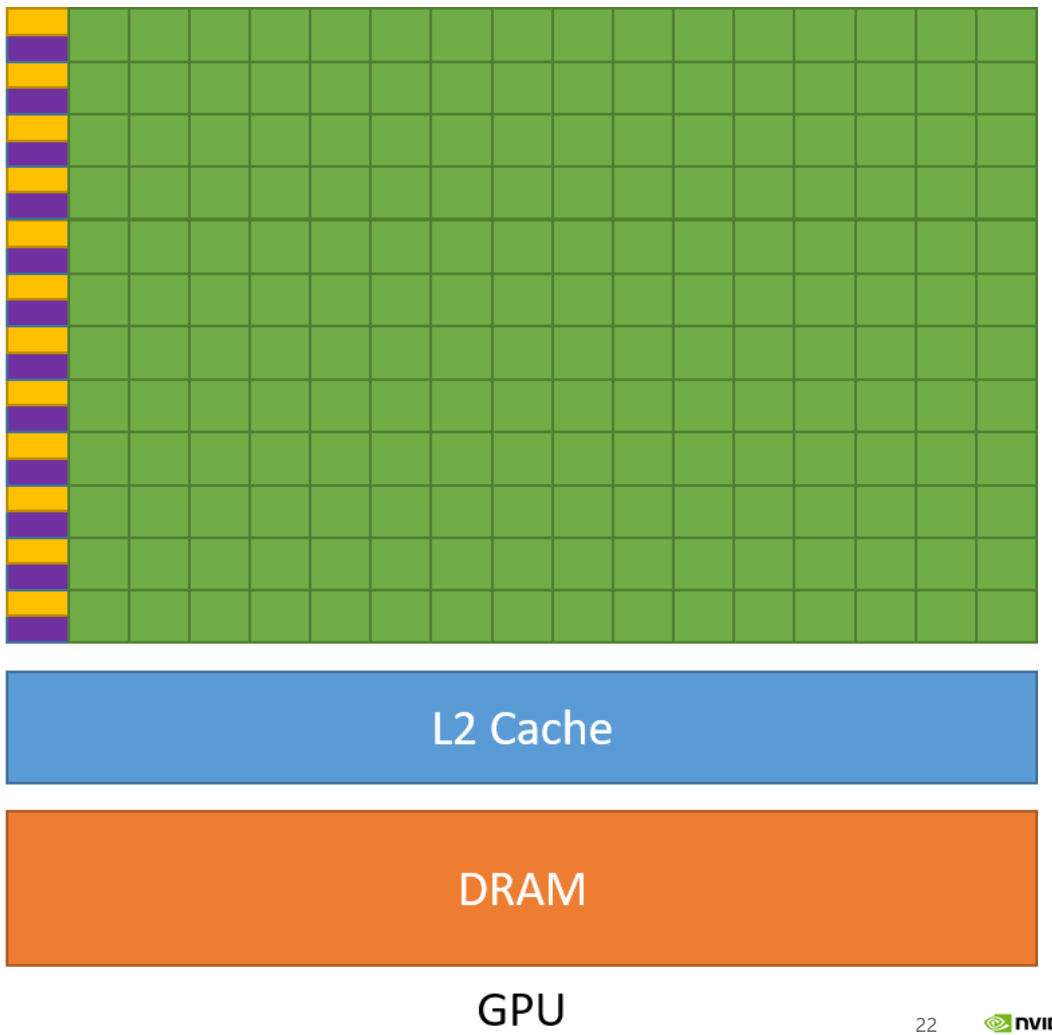


= **CUDA core**
(1 MUL-ADD per clock)

- 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**

芯片结构


Core
Con trol
L1 Cache
Core
Con trol
L1 Cache
Core
Con trol
L1 Cache
Core
Con trol
L1 Cache
L2 Cache
L2 Cache
L3 Cache
DRAM
CPU


L2 Cache
DRAM
GPU

GPU结构---GA100


PCI Express 4.0 Host Interface
GigaThread Engine with MIG Control
GPC
TPC
SM
L2 Cache
L2 Cache
Memory Controller
HBM2
High-Speed Hub
NVLink

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

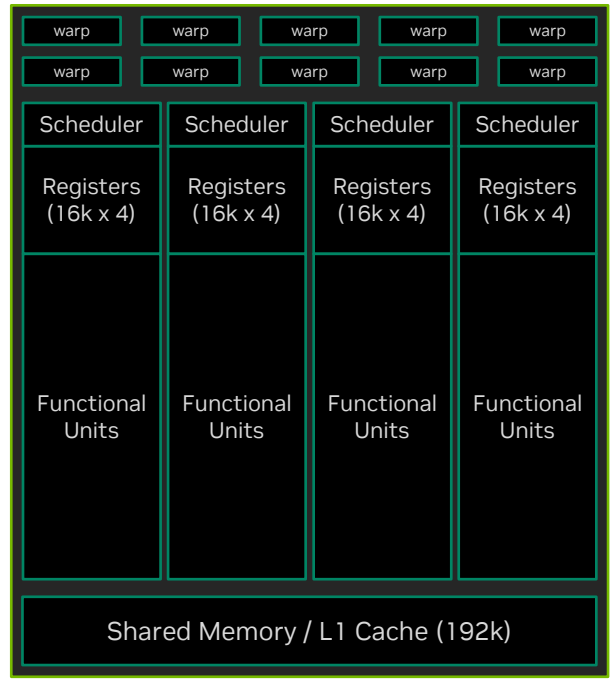
· 4 third-generation Tensor Cores/SM, 512 third-generation Tensor Cores per full GPU

· 6 HBM2 stacks, 12 512-bit memory controllers

GPU结构---GA100

	Per SM	On A100
Total Threads	2048	221,184
Total Warps	64	6,912
Active Warps	4	432
Waiting Warps	60	6,480
Active Threads	128	13,824
Waiting Threads	1,920	207,360

A100 Streaming Multiprocessor (SM)



64 warps/SM

4x concurrent warp exec

64k x 4-byte registers

192KB L1/shared memory (configurable split)

The GPU can switch from one warp to the next in a **single clock cycle**

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

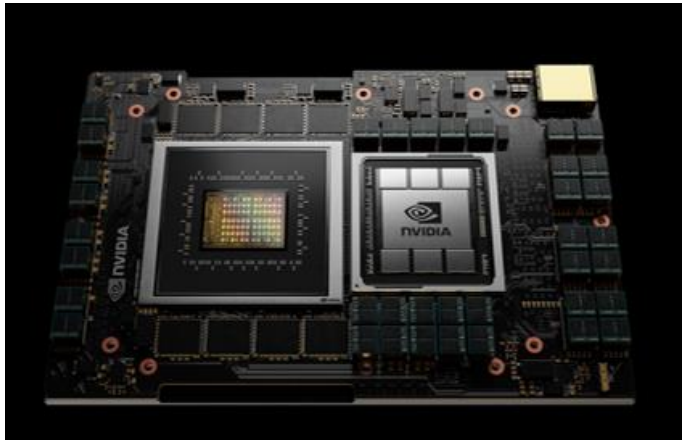
ARM CPU + 内置NVIDIA GPU

计算能力相当于多台台式机

Jetson Xavier NX 开发者套件技术规格	
GPU	NVIDIA Volta™ 架构 搭载 384 个 NVIDIA CUDA® 核心和 48 个 Tensor Core
CPU	6 核 NVIDIA Carmel ARM®v8.2 64 位 CPU 6 MB L2 + 4 MB L3
DLA	2x NVDLA
视觉加速器	2x PVA
显存	8 GB 128 位 LPDDR4x 59.7GB/s
存储	microSD（不包括卡）
视频编码	2x 4K60 | 4x 4K30 | 10x 1080p60 | 22x 1080p30 (H.265) 2x 4K60 | 4x 4K30 | 10x 1080p60 | 20x 1080p30 (H.264)
视频解码	2x 8K30 | 6x 4K60 | 12x 4K30 | 22x 1080p60 | 44x 1080p30 (H.265) 2x 4K60 | 6x 4K30 | 10x 1080p60 | 22x 1080p30 (H.264)
摄像头	2 个 MIPI CSI-2 D-PHY 通道
连接	千兆位以太网，M.2 Key E（包括 WiFi/BT），M.2 Key M (NVMe)
显示器	HDMI 和 DP
USB	4x USB 3.1、USB 2.0 Micro-B
其他	GPIO、I2C、I2S、SPI、UART
规格尺寸	103 毫米 x 90.5 毫米 x 34 毫米

ARM CPU + 内置NVIDIA GPU

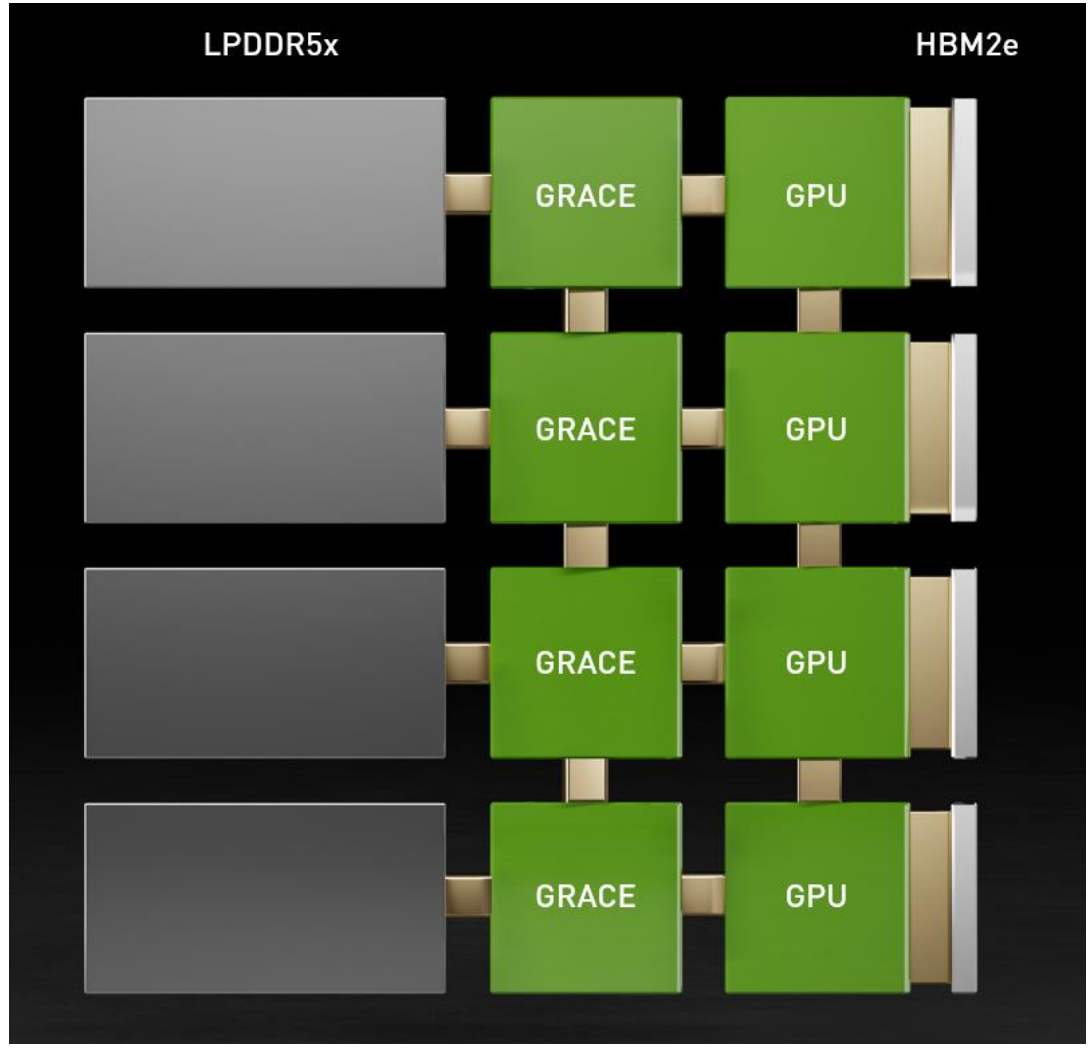
计算能力相当于多台台式机



A NEW COMPUTING ARCHITECTURE FOR AI AND DATA SCIENCE

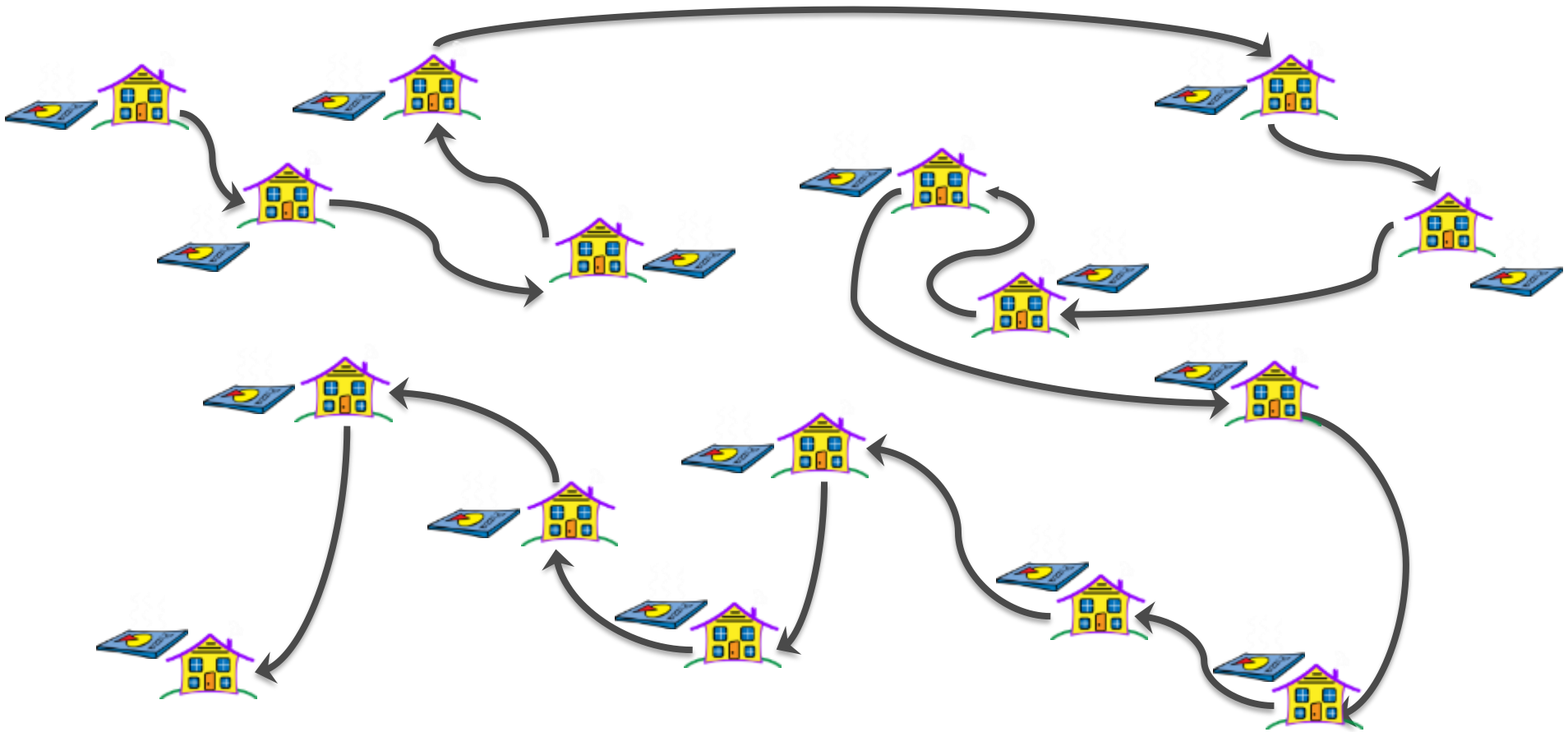
30X Increase System Memory to GPU

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



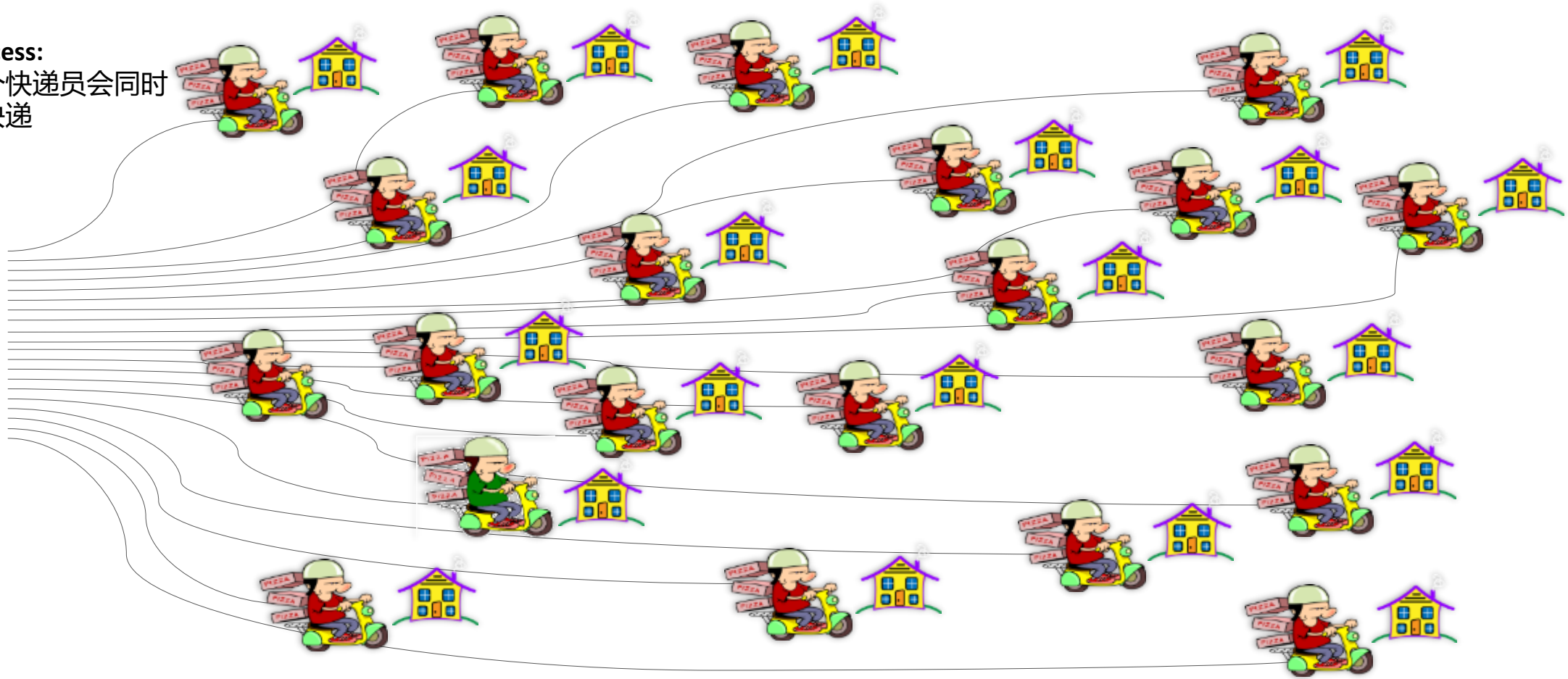
CPU 处理披萨快递

Process:
快递员会一家接着一家的送快递



NVIDIA GPU 披萨快递

Process:
多个快递员会同时送快递



Synchronous

Asynchronous

CUDA并行计算模式

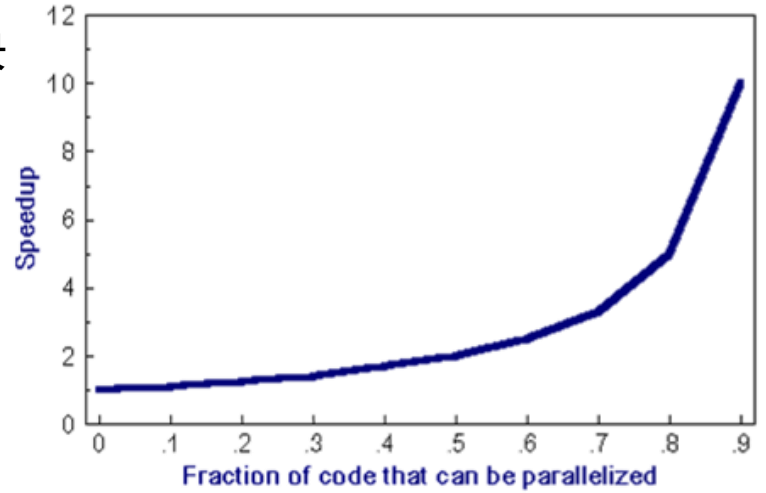
- 一句话：并行计算是同时应用多个计算资源解决一个计算问题
  - 涉及多个计算资源或处理器
  - 问题被分解为多个离散的部分，可以同时处理（并行）
  - 每个部分可以由一系列指令完成
- 最好是计算密集的任务
  - 通信和计算开销比例合适
  - 不要受制于访存带宽

- 单一指令或少指令的大数据集任务
- 现在这些都能在基于ARM平台的Jetson NANO上完成！

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

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

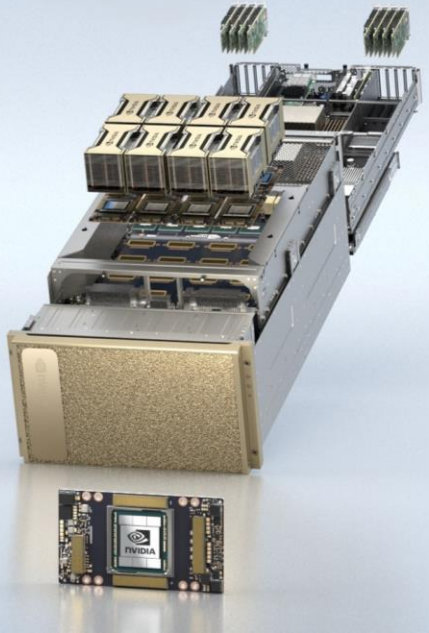
- P = 并行部分, N = 处理器数量 and S = 串行部分

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.cn/zh-cn/community-training


NVIDIA