微机原理与接口技术

姓名：陈致蓬

单位：中南大学自动化学院

电话： 15200328617

Email ： ZP.Chen@csu.edu.cn

Homepage:

https://www.scholarmate.com/psnweb/homepage

QQ ： 315566683

第 4 章　GPU 的功能和架构

4.1 GPU 基本介绍

4.2 GPU 与 CPU 的区别

4.3 GPU 底层架构

4.1 GPU 基本介绍
4.1.1 GPU 概念简介

GPU ，图形处理器（graphics processing unit），又称显示核心、视觉处理器、显示芯片，是一种专门在个人电脑、工作站、游戏机和一些移动设备（如平板电脑、智能手机等）上做图像和图形相关运算工作的微处理器。

4.1.2 GPU 的诞生

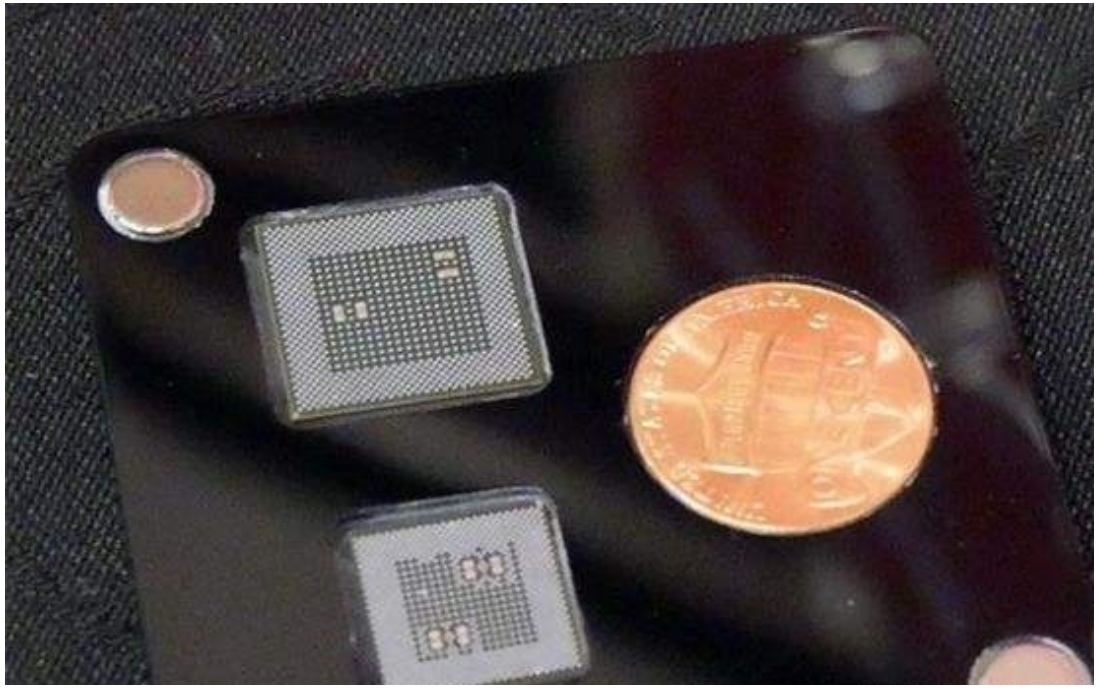
- NVIDIA 公司在 1999 年 8 月 31 日发布 GeForce 256 图形处理芯片时首先提出 GPU 的概念。

- GPU 之所以被称为图形处理器，最主要的原因是因为它可以进行几乎全部与计算机图形有关的数据运算，而这些在过去是 CPU 的专利。

- 目前，计算机图形学正处于前所未有的发展时期。近年来，GPU 技术以令人惊异的速度在发展。渲染速率每 6 个月就翻一番。性能自 99 年，多年来提高了上千倍！与此同时，不仅性能得到了提高，计算质量和图形编程的灵活性也逐渐得以改善。

4.1.3 GPU 宏观物理形态

- 由于纳米工艺的引入，GPU 可以将数以亿记的晶体管和电子器件集成在一个小小的芯片内。从宏观物理结构上看，现代大多数桌面级 GPU 跟硬币同数量级大小。



4.1.3 GPU 宏观物理形态

- 当 GPU 结合散热风扇、 PCI 插槽、 HDMI 接口等部件之后，就组成了显卡。

显卡不能独立工作，需要装载在主板上，结合 CPU 、内存、显存、显示器等硬件设备，组成完整的 PC 机

4.1.4 GPU 的功能

✂图形绘制

GPU 最传统、最基础、最核心的功能。为大多数 PC 桌面、移动设备、图形工作站提供图形处理和绘制功能。

✂物理模拟

GPU 硬件集成的物理引擎（PhysX 、 Havok），为游戏、电影、教育、科学模拟等领域提供了成百上千倍性能的物理模拟，使得以前需要长时间计算的物理模拟得以实时呈现。

✂海量计算

计算着色器及流输出的出现，为各种可以并行计算的海量需求得以实现，例如 CUDA 。

✂AI 运算

近年来，人工智能的崛起推动了 GPU 集成了 AI Core 运算单元，反哺 AI 运算能力的提升，给各行各业带来了计算能力的提升。

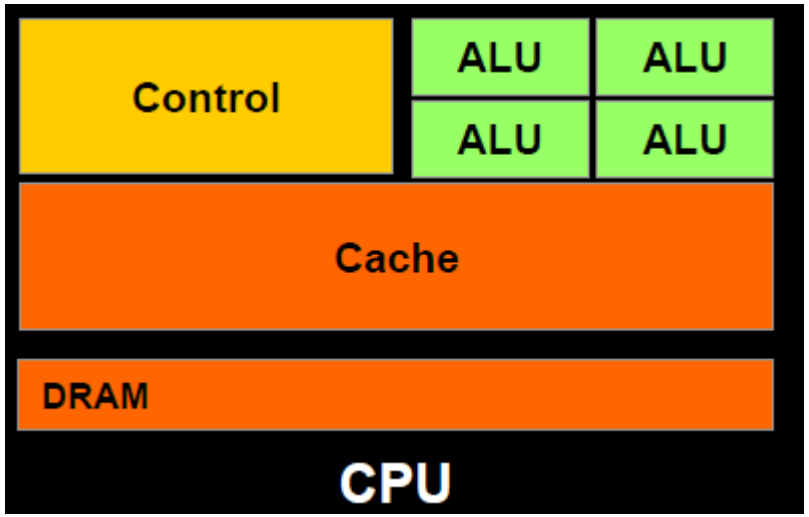
✂其它计算

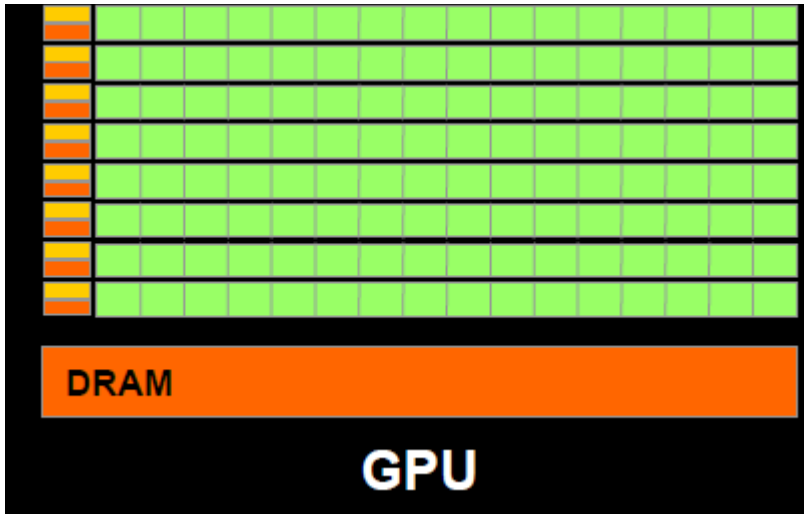
音视频编解码、加解密、科学计算、离线渲染等都离不开现代 GPU 的并行计算能力和海量吞吐能力。

4.1.5 GPU 历代架构发展

- Fermi 费米架构，包含 GTX4 、 GTX5 两个系列
- Kepler 开普勒架构， GTX6 系列
- Maxwell 麦克斯韦架构， GTX9 系列和 GTX750
- Pascal 帕斯卡架构， GTX10 系列
- Turing 图灵架构， RTX20 系和 GTX16 系列
- Ampere 安培架构， RTX30 系列
- Adm 安达架构， RTX40 系列

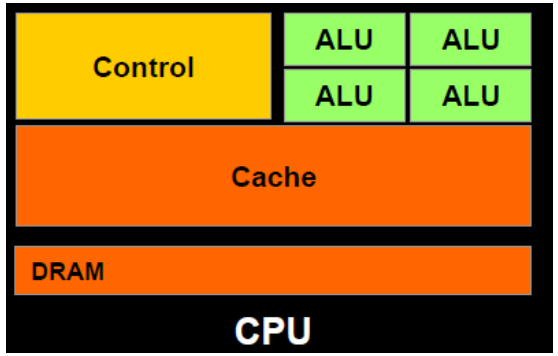
4.2 GPU 与 CPU 的区别

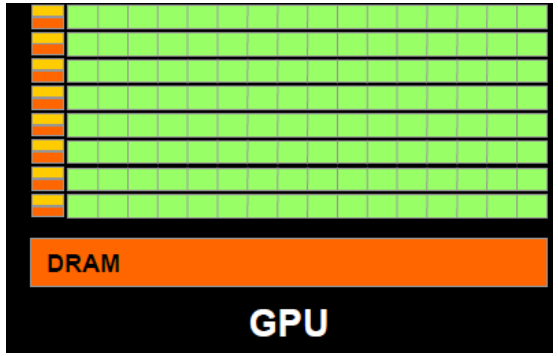




- 绿色方框： **computational units**(可计算单元) 或称之为 **computational cores**(计算核心)
- 橙色方框： **memories** （内存）
- 黄色方框： **control units** （控制单元）

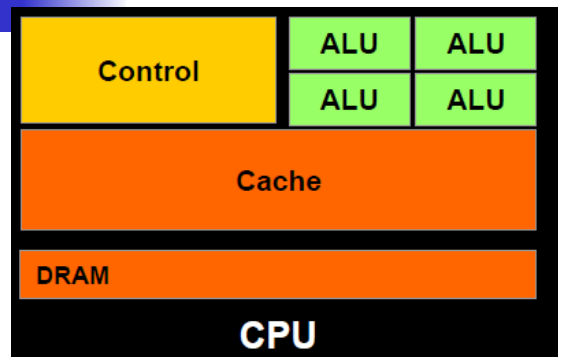
4.2.1 Computational units(cores) 的区别



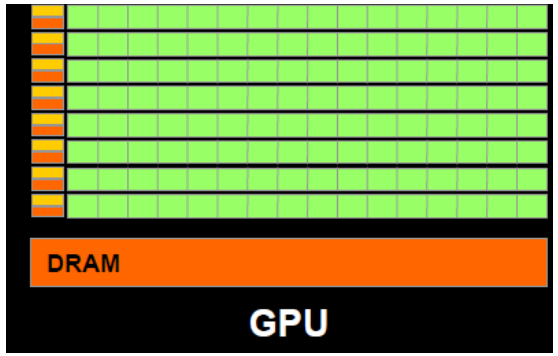


- **CPU 的 Computational units 计算能力强、数量少；**
- **GPU 的 Computational units 计算能力弱、数量多；**

4.2.1 Computational units(cores) 的区别

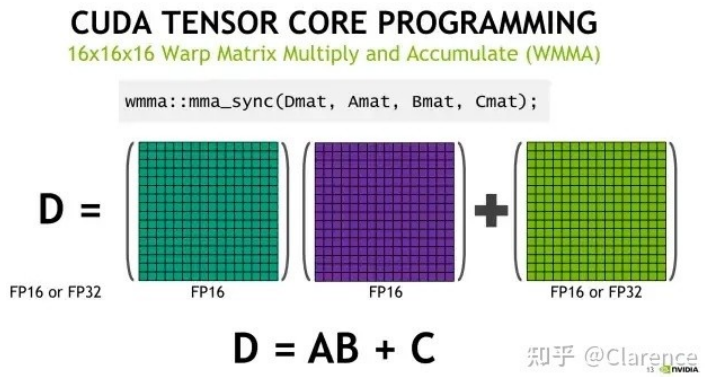


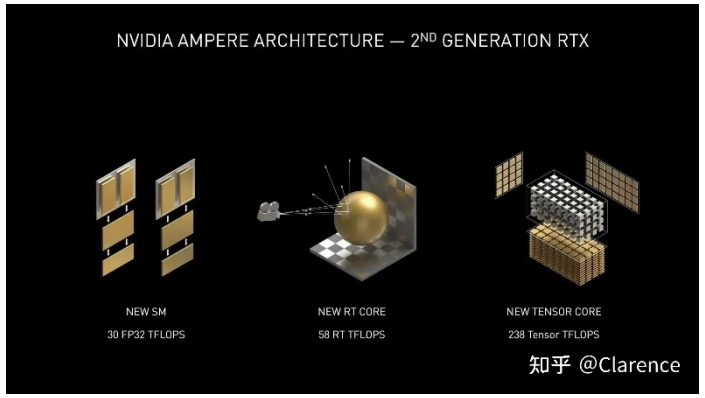
- CPU 的优势在于核心的处理能力、计算能力强，拥有 "out-of-order exectutions" （乱序执行）功能；



- GPU 的优势在于大规模并行处理数据的能力，采用 SIMD 编程方式，各个 Core 的计算操作在相同的时间内进行；(MAD 与 FMA) MAD 指令实际是计算 A*B+C 的值，区别

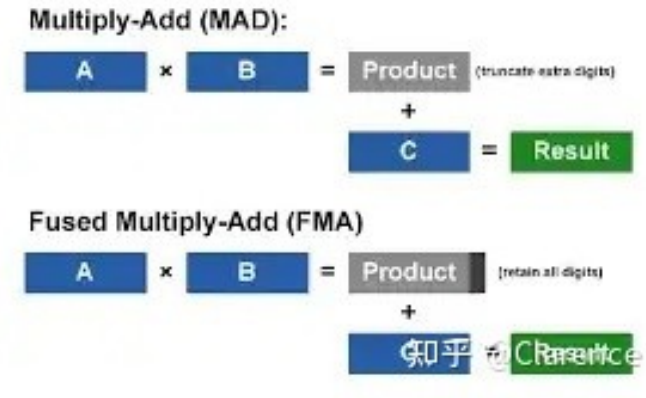
4.2.1 Computational units(cores) 的区别



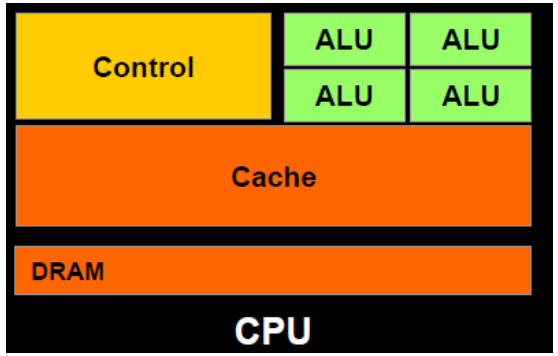


- **GPU 的 core 只能做一些最简单的浮点运算，例如 multiply-add(MAD) 或者 fused multiply-add(FMA) 指令；**

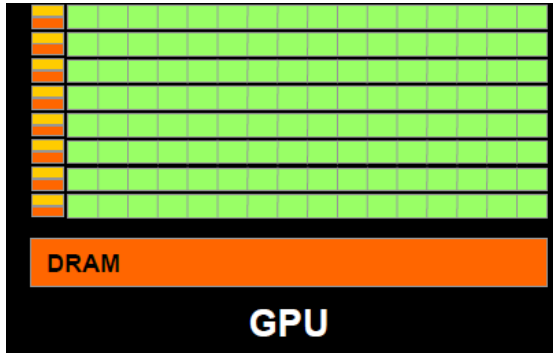
- **核心区别 MAD 与 FMA 的不同，目的都是求解 A*B+C**



4.2.2 Memory 的区别



- CPU 的 memory 系统一般基于 DRAM ，其内存系统中的 cache 可减少 CPU 访问 DRAM 的时间；



- GPU 中的内存（DRAM）被称为全局内存或者 GMEM ，其内存大小比 CPU 的 DRAM 小的多；

- GPU 左上角的小橙色块是 GPU 的 cache 段，GPU 与 CPU 缓存机制不同（在 4.3 中说

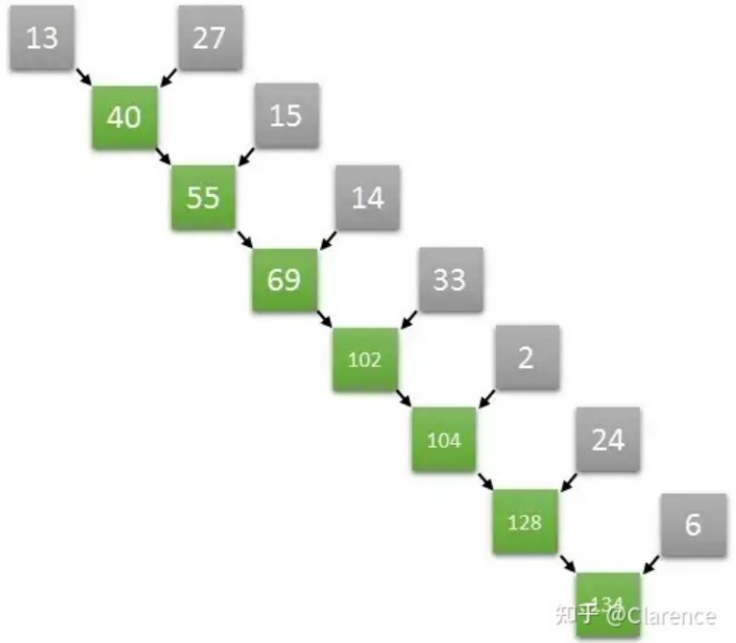
4.3 GPU 底层架构

- 在特定的应用场合，多个 core 之间是不需要的通讯的，例如在图像缩放中， core 与 core 之间不需要任何协作，他们的任务是完全独立，因此简单的并行操作即可。

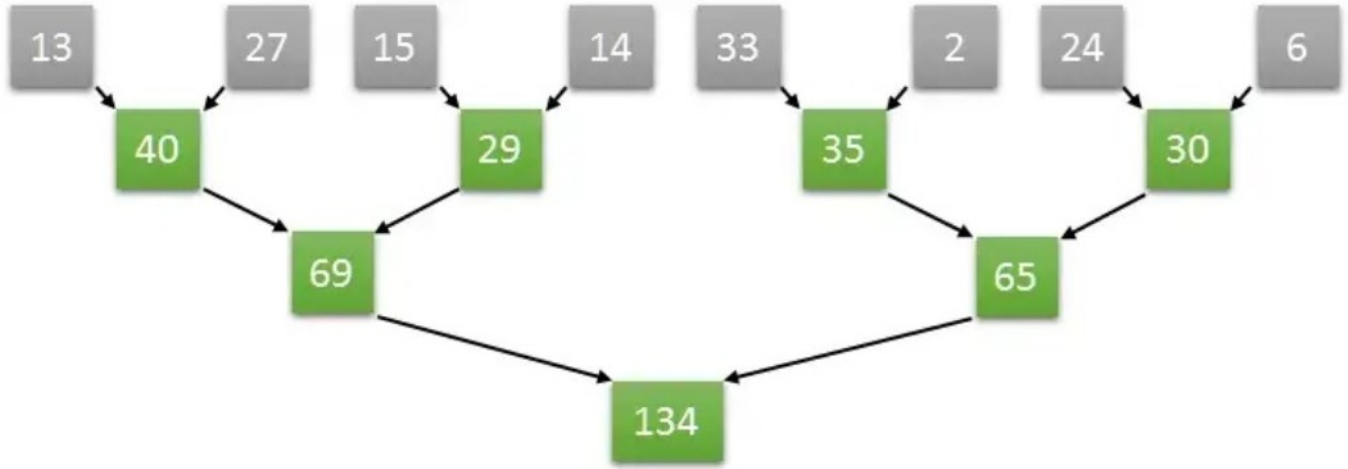
- 但也存在许多场合，需要多个 core 之间相互通讯协作，例如数组求和问题。

4.3.1 GPU 数组求和问题

串行

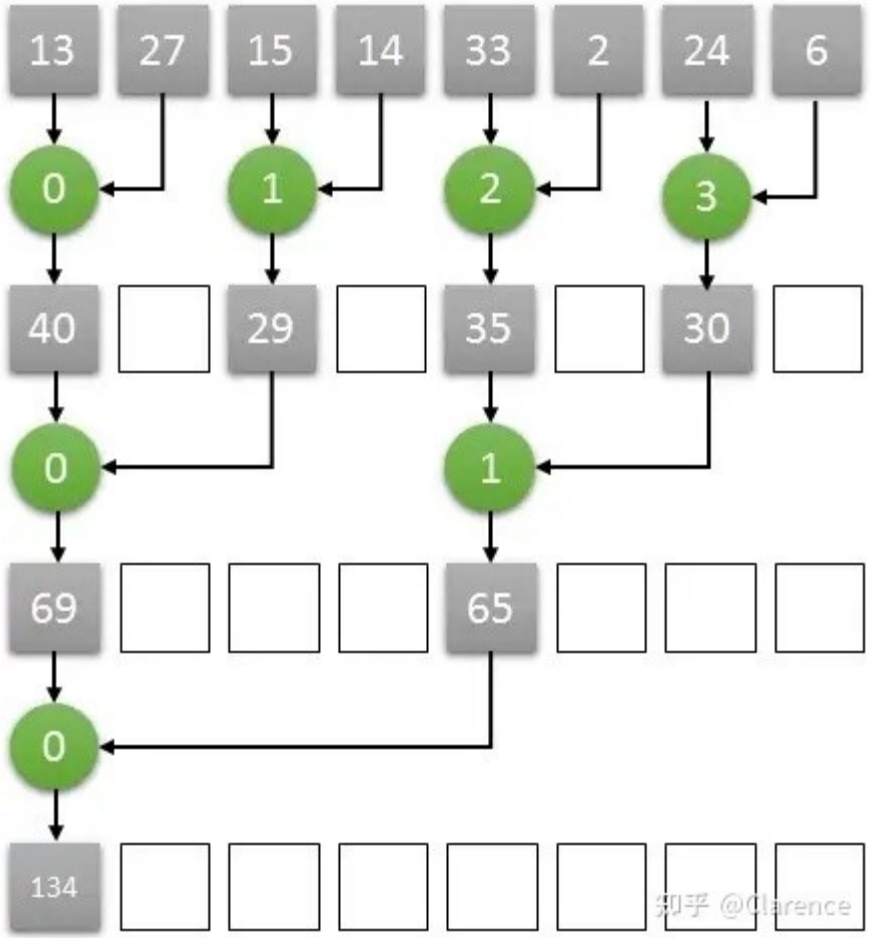


并行



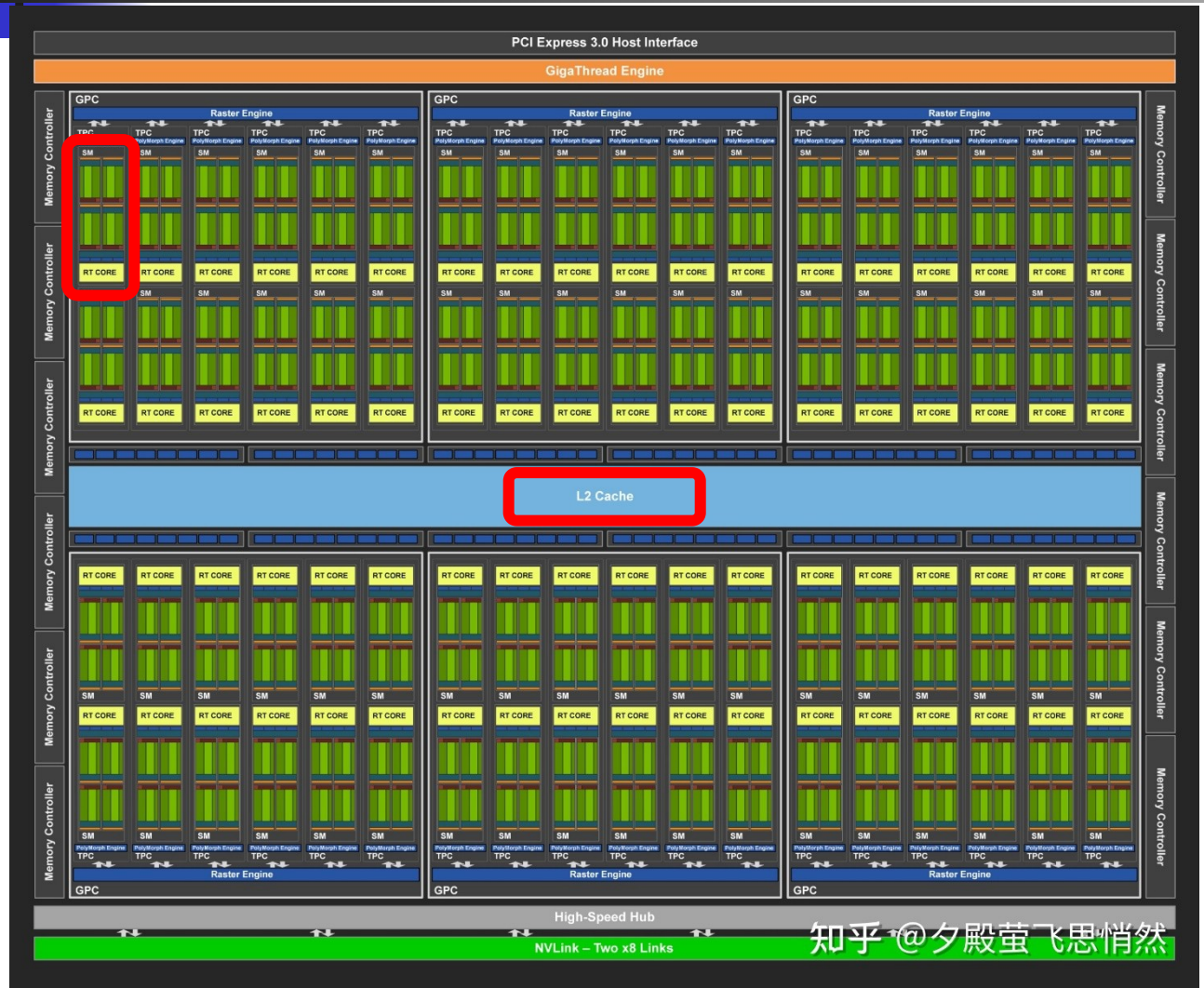
长度为 8 的数组两两并行求和计算，只需要 3 次就可以计算出结果，顺序计算则需要 8 次。如果按照两两并行相加的算法， N 个数字相加，那么仅需要 log2 （ N ）次就可以完成计算。

4.3.1 GPU 数组求和问题



- GPU 只需四个 core 就可以完成长度为 8 的数组求和算法。
- 多个 core 之间需共享一段内存空间，进行读 / 写操作，以此完成数据交互。
- 多个 core 之间协作方法：将各类 GPU 的 core 分类为多个组，形成多个流处理器 (Streaming Multiprocessors) ，简称为 SMs 。

4.3.2 GPU 总体架构（以图灵架构为例）



绿色的块即为 SM （一组 core 的集合）

4.3.3 SM （以图灵架构为例）

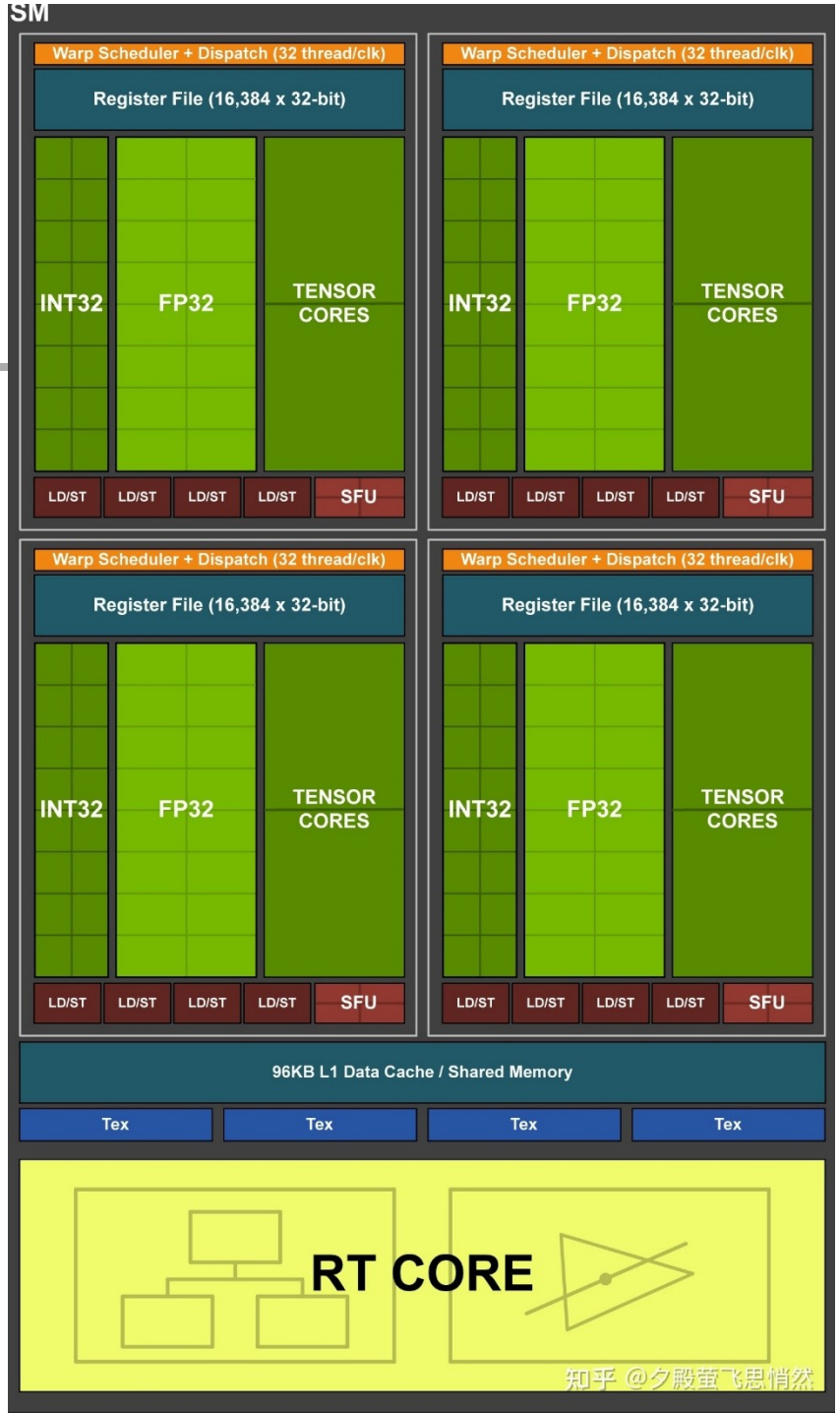
其存在不同类型的 CORE 。主要分为 INT32,FP32 ， TENSOR CORES 等。

FP32 Cores ，执行单进度浮点运算；

FP64 Cores ，执行双进度浮点运算；

Integer Cores ，执行整数操作；

Tensor Cores ，执行张量积算加速常见的深度学习操作；

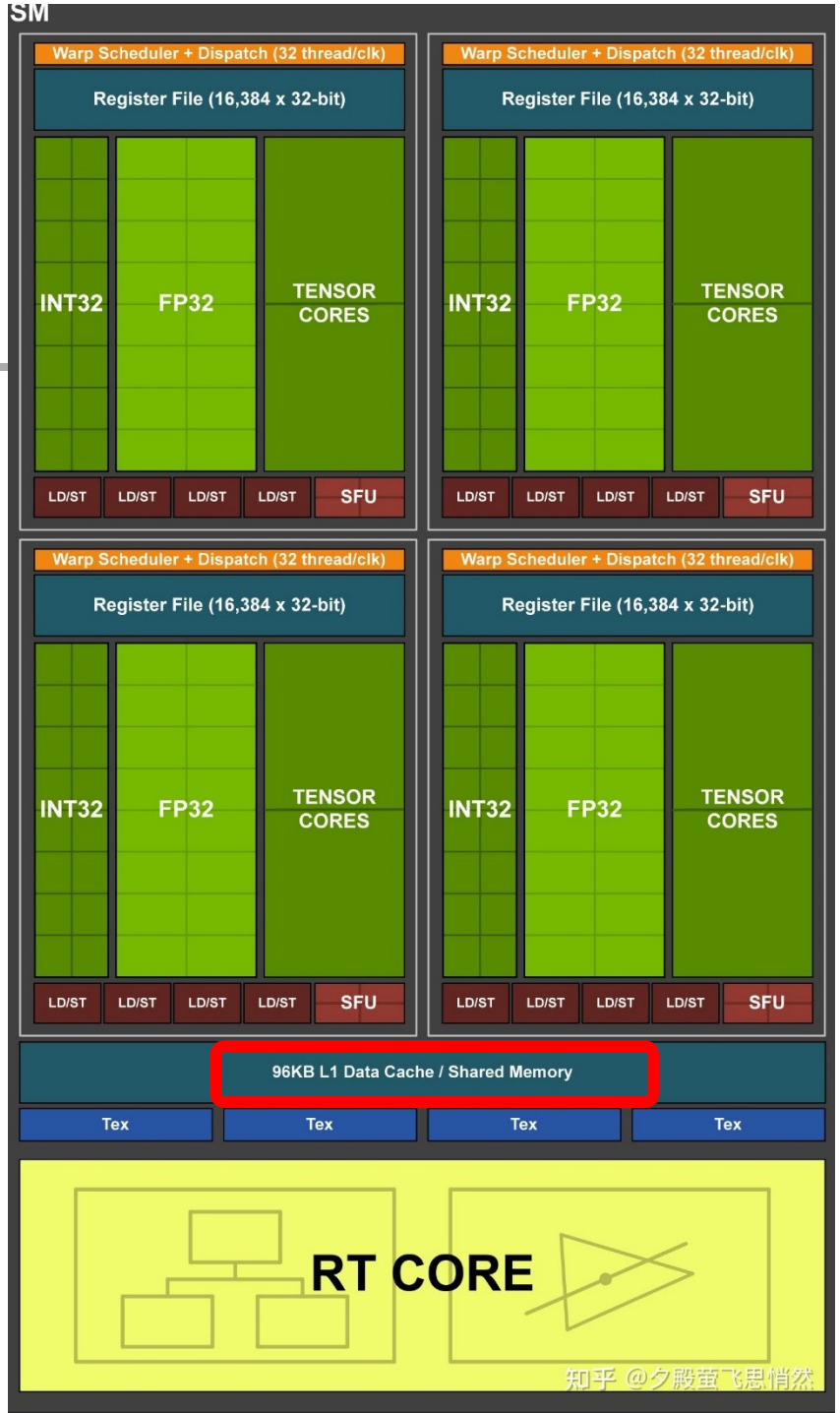


4.3.3 SM （以图灵架构为例）

GPU 的缓存机制及多个 core 之间的通信协作

在四个 SM 块的底部有一个 L1 Cache ，允许各个 Core 访问，在 L1 Cache 中每个 SM 都有一块专用的共享内存。 L1 cache 大小有限，速度非常快，比访问 GMEM 快得多。

L1 Cache 拥有两个功能，一个是用于 SM 上 Core 之间相互共享内存，另一个则是普通的 cache 功能。当 Core 需要协同工作，编译器编译后的指令会将部分结果储存在共享内存中，以便于不同的 core 获取到对应数据。当用做普通 cache 功能的时候，当 core 需要访问 GMEM 数据的时候，首先会在 L1 中查找，如果没找到，则回去 L2 cache 中寻找，如果 L2 cache 也没有，则会从 GMEM 中获取数据， L1 访问最快 ， L2 其次， GMEM 最慢。



4.3.4 GPU 未来硬件升级

- 更多运算单元；
- 更多存储空间；
- 更高并发；
- 更高带宽；
- 更低延时；
- …… ……

谢谢大家！