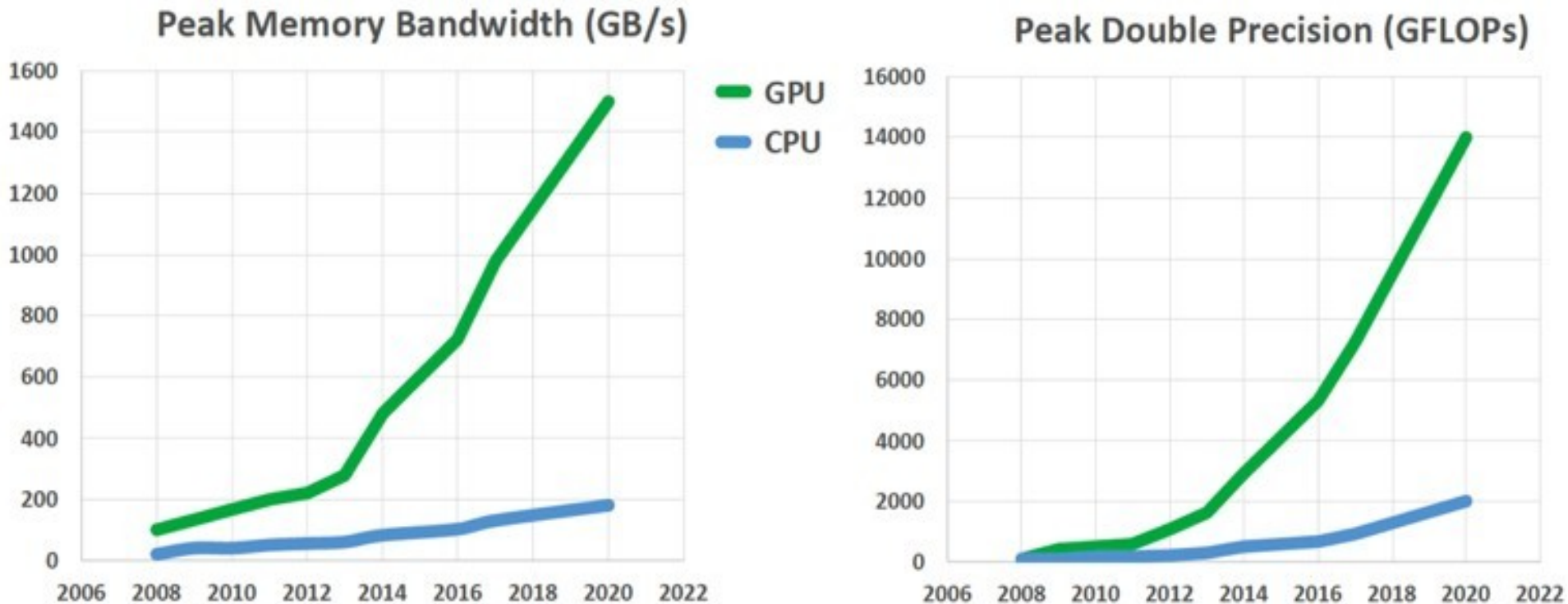


GPU Computing

More on GPU

## Peak Floating Point Operations per Second And Peak Memory Bandwidth for CPU and GPU

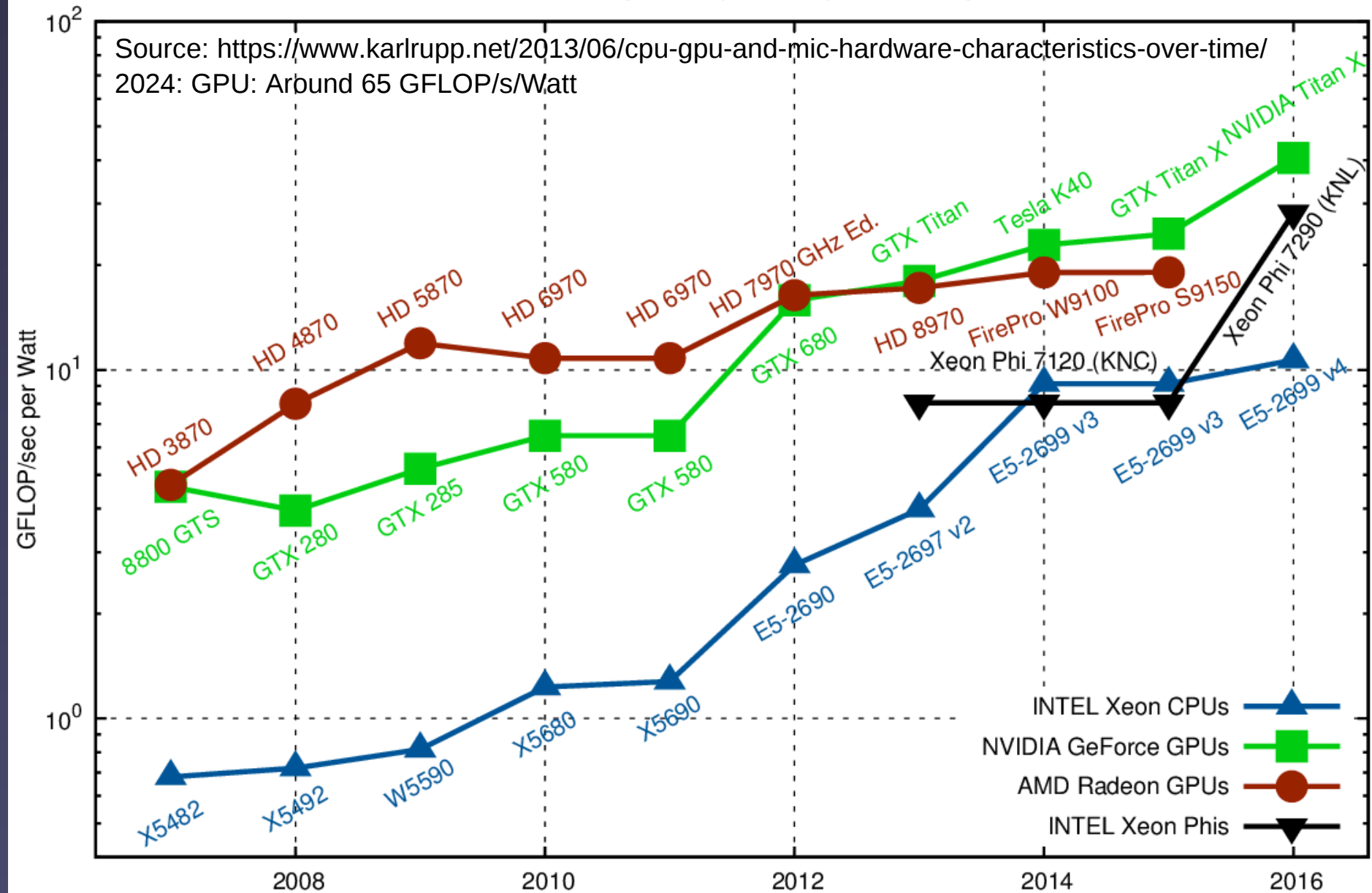


Chip to chip comparison of peak memory bandwidth in GB/s and peak double precision gigaflops for GPUs and CPUs since 2008. Data for Nvidia "Volta" V100 and Intel "Cascade Lake" Xeon SP are used for 2019 and projected into 2020. From:

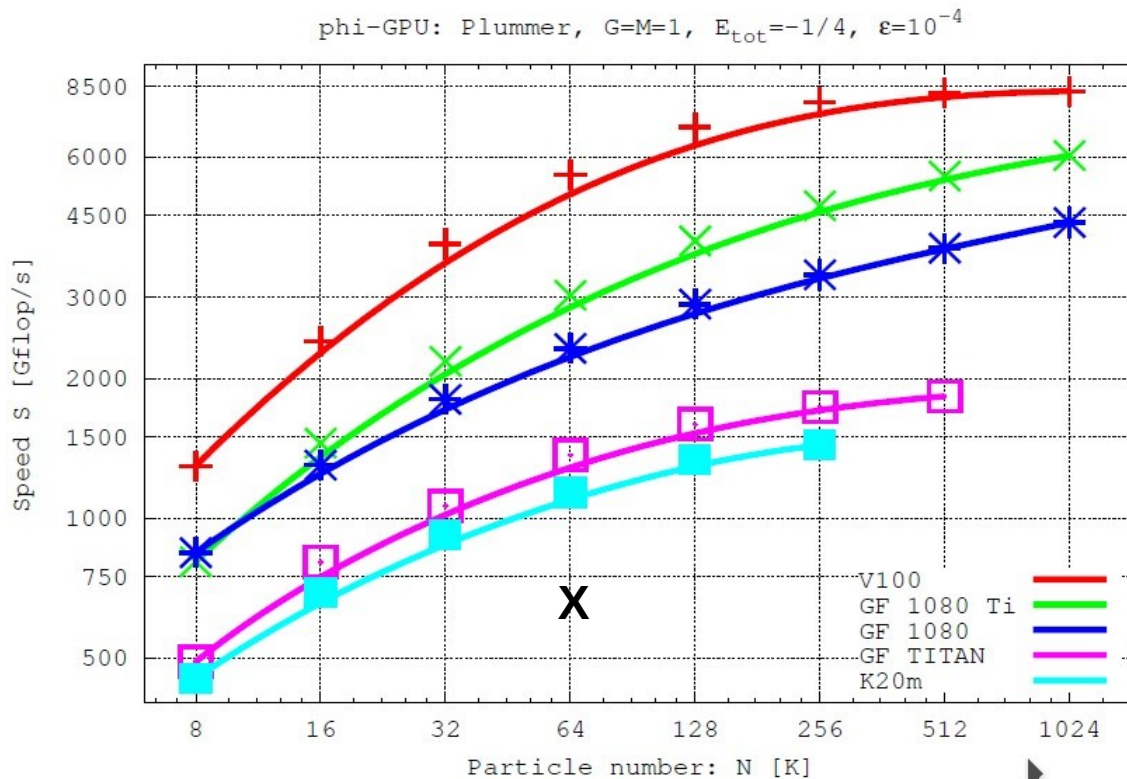
<https://www.nextplatform.com/2019/07/10/a-decade-of-accelerated-computing-augurs-well-for-gpus/>

### Theoretical Peak Floating Point Operations per Watt, Single Precision

Source: <https://www.karlrupp.net/2013/06/cpu-gpu-and-mic-hardware-characteristics-over-time/>  
 2024: GPU: Arbund 65 GFLOP/s/Watt



# Kepler, Pascal, Volta, Scaling, it works...



Volta V100

Pascal GF1080

Kepler K20m

Spurzem, Berczik,  
et al., 2013,  
LNCS Supercomputing,  
2013, pp. 13-25,  
Springer.  
(updated unpublished)

**Fig. 4.** Here we report a preliminary result from a benchmark test of our code on one Kepler K20 card; we compare with the performance on Fermi C2050 (used in the Mole-8.5 cluster), and the oldest Tesla C1060 GPU (used in the laohu cluster of 2009) - the latter is used as a normalization reference. We plot the speed ratio of our usual benchmarking simulation used in the previous figures, as a function of particle number. From this we see the sustained performance of a Kepler K20 would be about 1.4 - 1.5 Tflop/s.

**X = first GPU of laohu 2010**



# Graphics Processors (GPU) as General Purpose Supercomputers (GPGPU)



2019:  
kepler wn14  
RTX 2080 Ti

2008...

GeForce 9800 GTX, 128 Stream Proc., 512 MB

GeForce 9800 GX2, 256 Stream Proc., 1 GB

GeForce 9800 GT, 64 Stream Proc., 512 MB

[...]

2009: Tesla ~200 Proc., 4GB

2010: Fermi ~400 Proc., 4GB

2013: Kepler K20, ~2500 Procs., 6GB

2016: Kepler K80, ~5000 Procs.

2017/18: Pascal, Volta, Ampere > 5000 Procs., 40 GB

## NVIDIA RTX A5000

2022: kepler wn15

### RTX A5000

- ▶ Up to 27.8 TFLOPS Single Precision
- ▶ 24 GB GDDR6 w/ECC Memory
- ▶ 4x DP1.4 Display Outputs
- ▶ PCIe-Gen 4
- ▶ Quadro Sync
- ▶ 2-Way NVLink
- ▶ vGPU support



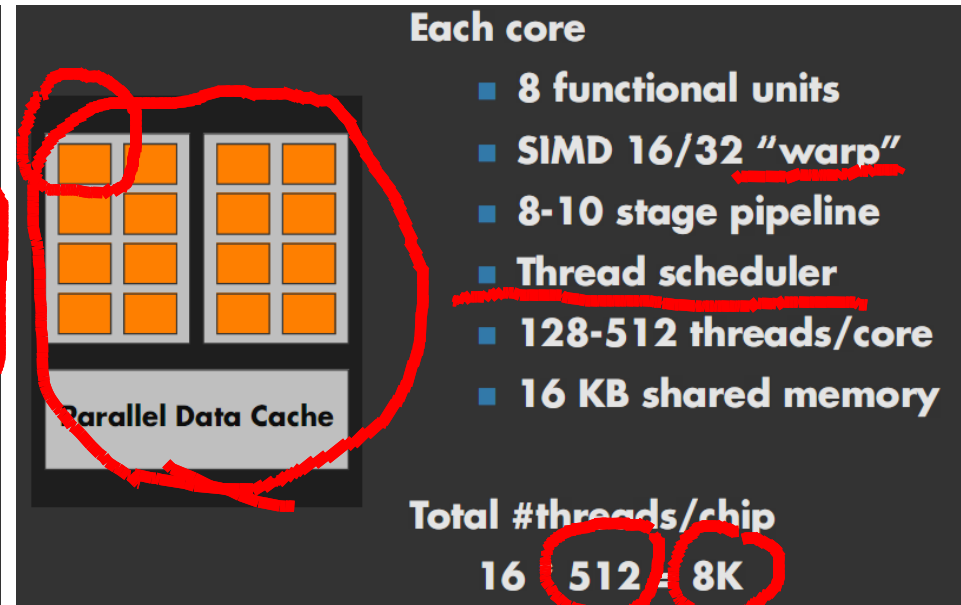
2010: Tesla C1070

Laohu 北京



# Hardware around 2006

architecture still valid – just scaled up (except: tensor cores and fast data links)

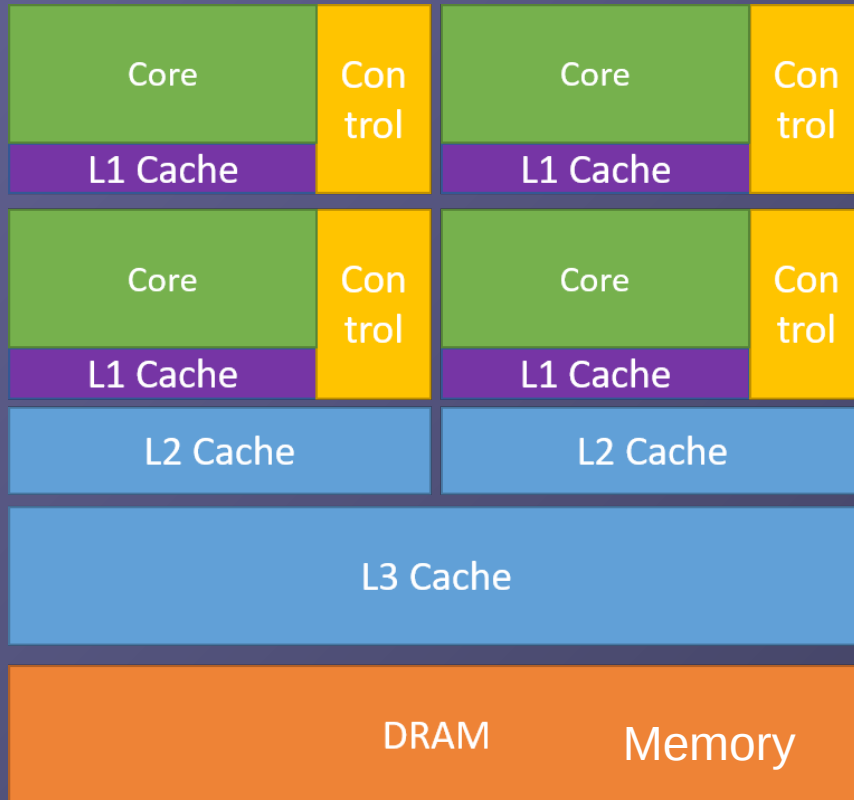


These are the physical parameters! The software ("runtime system") sees a "virtual GPU" which is MUCH larger!!

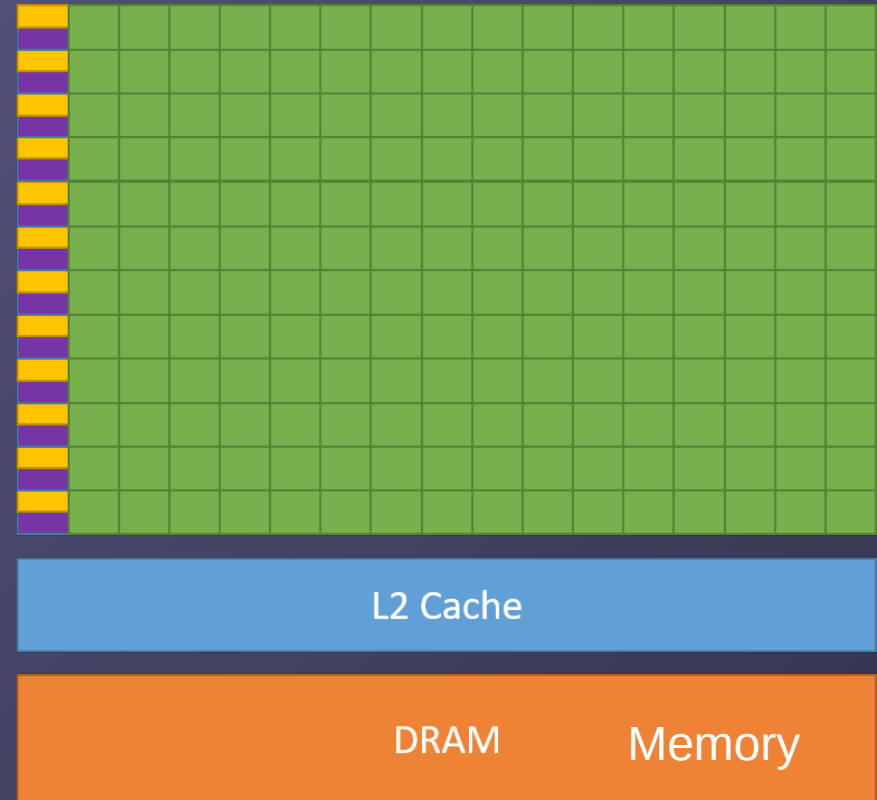
**GeForce 8800 GTX:**

**575 MHz \* 128 processors \* 2 flop/inst \* 2 inst/clock = 333 Gflops**

# CPU and GPU; from CUDA NVIDIA Developer Zone at <http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html>



CPU

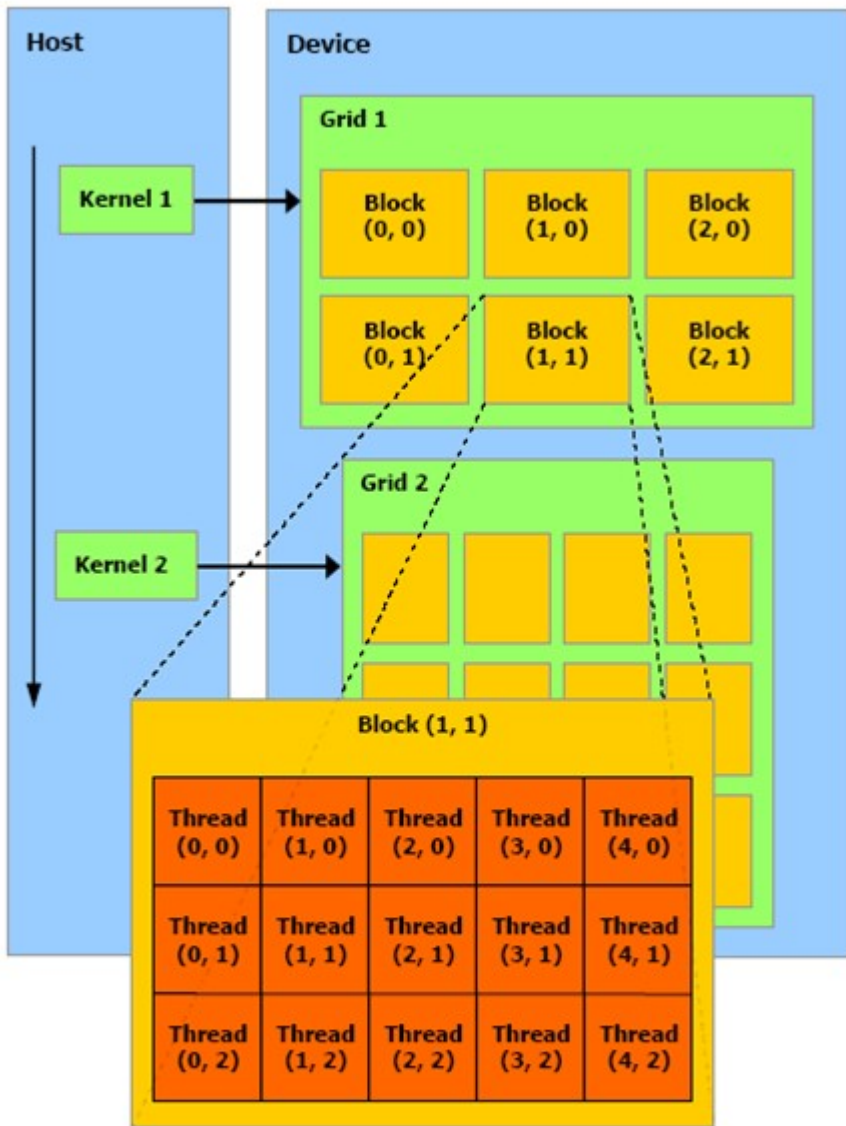


GPU

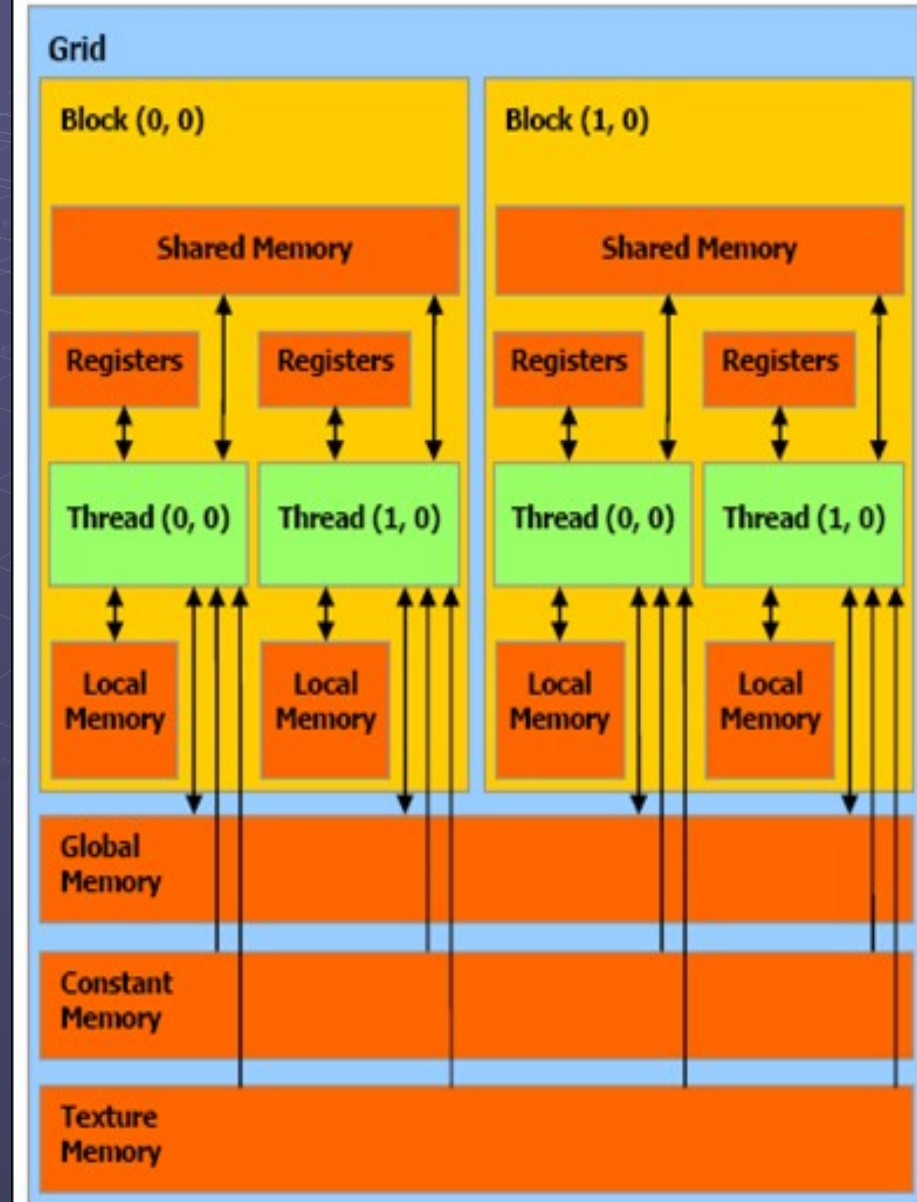
**“The GPU devotes more transistors to computing”  
“favours data parallel operations”**

# GPU Structure

<https://docs.nvidia.com/cuda/parallel-thread-execution/index.html>



The host issues a succession of kernel invocations to the device. Each kernel is executed as a batch of threads organized as a grid of thread blocks



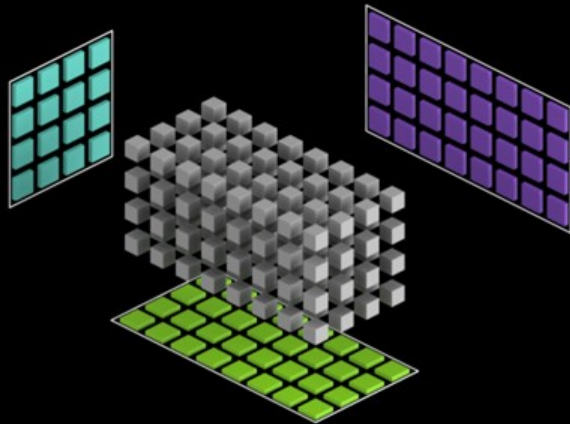


# New feature in Volta, Ampere, Turing: Tensor Cores

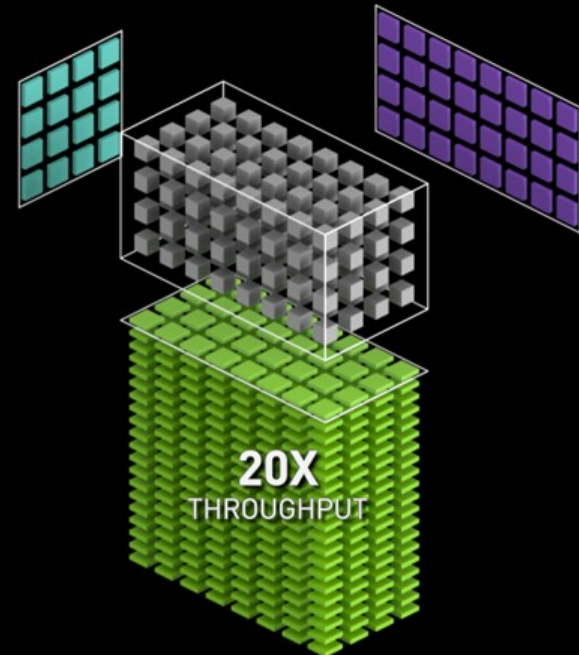
<https://www.nvidia.com/en-us/data-center/tensor-cores/>

**FP64 Tensor Cores:** “A100 brings the power of [Tensor Cores to HPC](#), providing the biggest milestone since the introduction of double-precision GPU computing for HPC. By enabling matrix operations in FP64 precision, a whole range of [HPC applications](#) that need double-precision math can now get a 2.5X boost in performance and efficiency compared to prior generations of GPUs.” (Quote from NVIDIA webpages)

NVIDIA V100 FP32



NVIDIA A100 Tensor Core TF32 with Sparsity



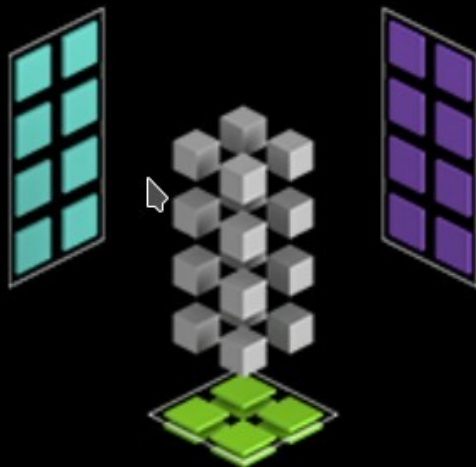
# New feature in Volta, Ampere, Turing: Tensor Cores

<https://www.nvidia.com/en-us/data-center/tensor-cores/>

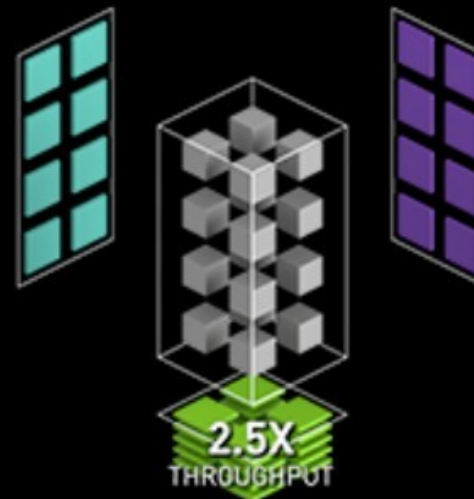
**FP64 Tensor Cores:** “A100 brings the power of [Tensor Cores](#) to HPC, providing the biggest milestone since the introduction of double-precision GPU computing for HPC. By enabling matrix operations in FP64 precision, a whole range of [HPC applications](#) that need double-precision math can now get a 2.5X boost in performance and efficiency compared to prior generations of GPUs.” (Quote from NVIDIA webpages)

TF32 FP64 FP16 INT8

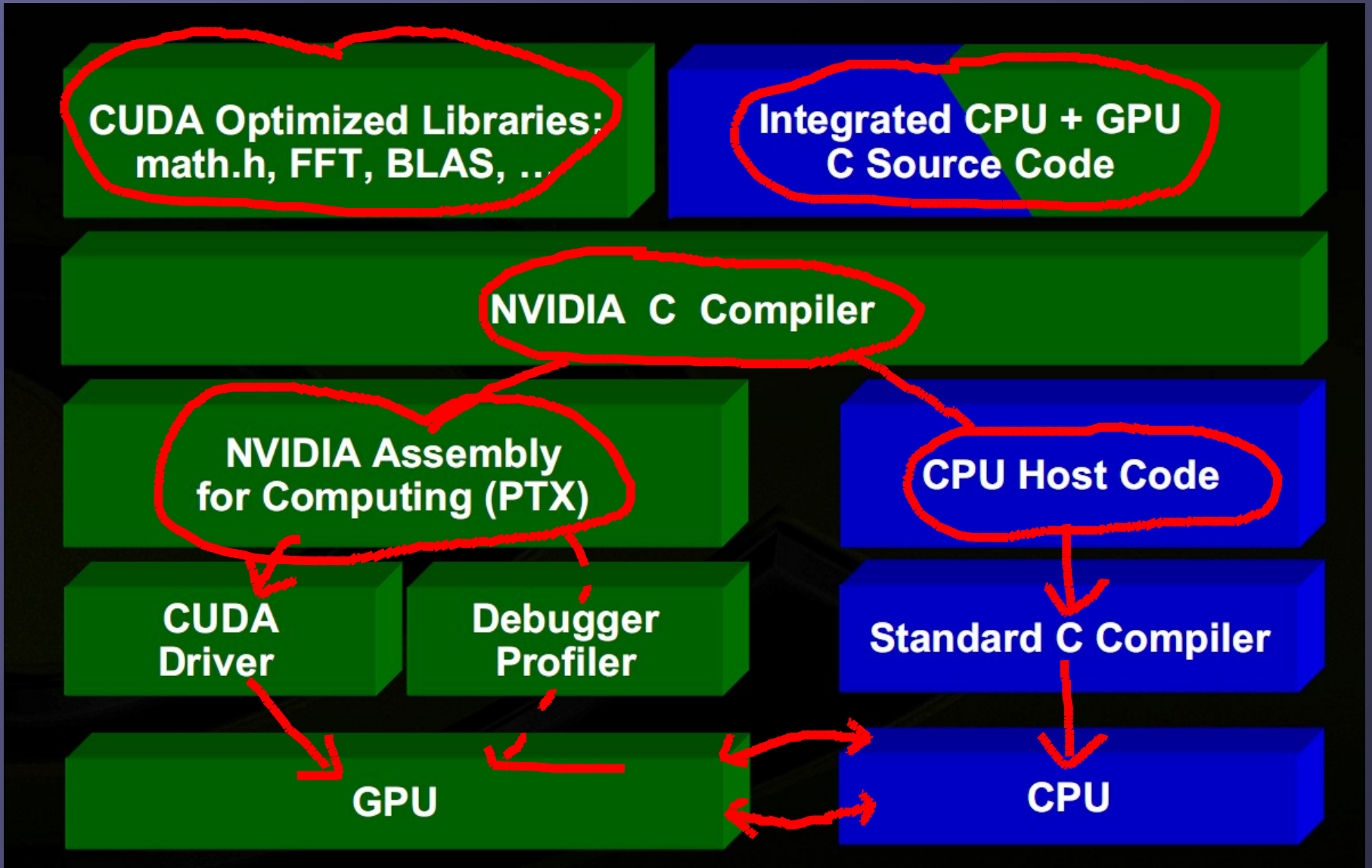
NVIDIA V100 FP64



NVIDIA A100 Tensor Core FP64



# CUDA



# 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)
---	-----	---------	----------------------------	---------------	------------------------------

Ampere and Volta:  
Tensor Cores/NVLink



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) Kepler (3.x)	Tegra X2 Tegra K1	GeForce 1000 Series GeForce 700/800	Quadro P Series Quadro K	Tesla P Series Tesla K
	 Embedded	 Consumer Desktop/Laptop	 Professional Workstation	 Data Center



# Python + CUDA = PyCUDA



- ▶ All of CUDA in a modern scripting language
- ▶ Full Documentation
- ▶ Free, open source (MIT)
- ▶ Also: PyOpenCL

- ▶ CUDA C Code = Strings
- ▶ Generate Code Easily
  - ▶ Automated Tuning
- ▶ Batteries included: GPU Arrays, RNG, ...
- ▶ Integration: numpy arrays, Plotting, Optimization, ...

