

# General Introduction to GPUs



Norwegian research infrastructure services

Hicham Agueny, PhD  
Scientific Computing Group  
IT-department, UiB

**One-day Program (10:00-15:45)**

**Slides and exercices**

<https://github.com/HichamAgueny/GPU-course.git>

## **Heterogenous computing with OpenACC**

### **I. General introduction to GPU**

- I-1. State-of-the-art supercomputers**
- I-2. Introduction to GPU-hardware (CPU vs GPU)**
- I-3. GPU-programming models and compiler supports**

**Break (11:00-11:15)**

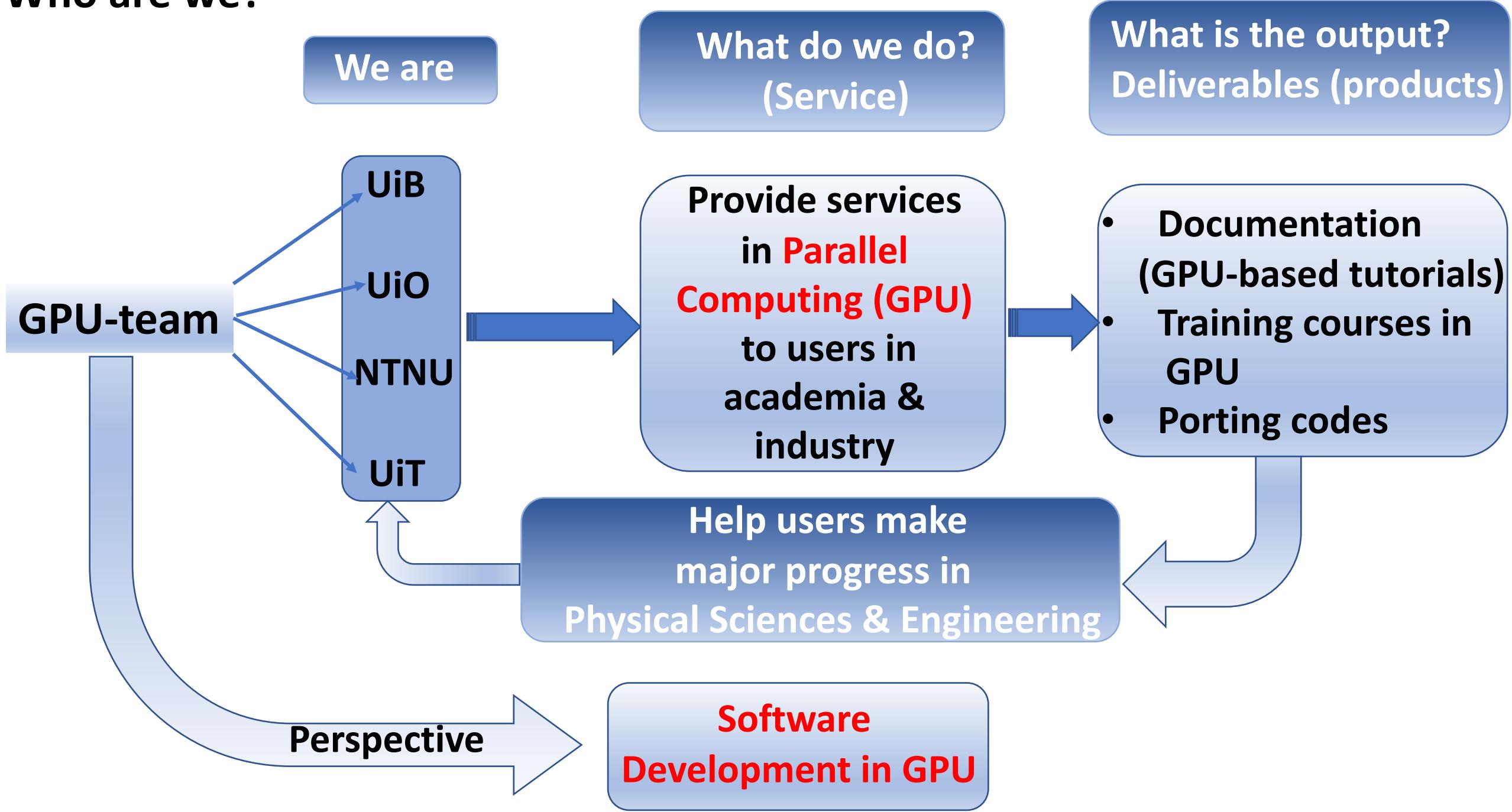
### **II. Showcase: OpenACC and functionality**

- II-1. Synchronous OpenACC**
- II-2. Asynchronous OpenACC**
- II-3. Code-profiling (Gprof+Nsight)**

**Lunch (12:00-13:00)**

### **III. Hands-on + discussion (2h)**

# Who are we?

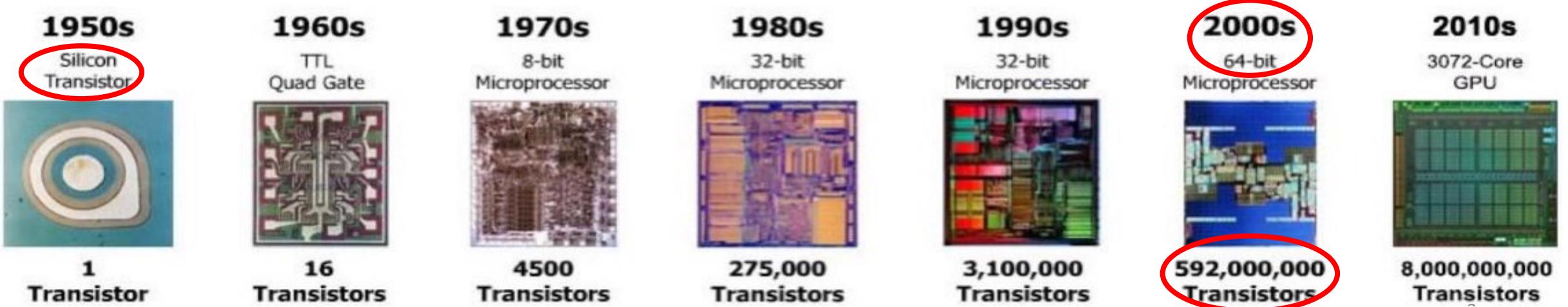


# Overview

- **State-of-the-art supercomputers**
- **CPU & GPU architectures** GPU: Graphics Processing Unit  
CPU: Central Processing Unit
- **GPU-programming models and compiler supports**
- **Parallelism in heterogenous systems**

# **State-of-the-art supercomputers**

# Evolution of Processors: last 70 years



- Transistors are the **building block of a processor**.
- **Transistors** are fabricated using a concept based on **metal-oxide-silicon**.
- **Clock rate**: Transistors can switch on/off (one-cycle) from million times per second (MHz) up to billion times per second (GHz).
- The performance of a processor depends on how fast transistors can switch on/off.

64-bit in 2000s was a big step in performing precise calculations



Mohamed Attala

Invented in 1959 & In production Until 2018



Dawon Kahng

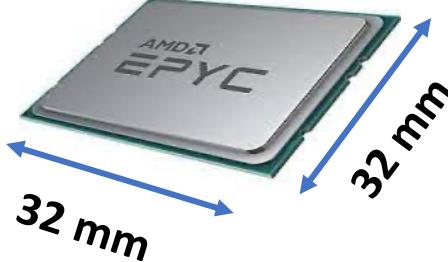
# Evolution of transistors in processor

S Our World  
in Data

AMD CPU

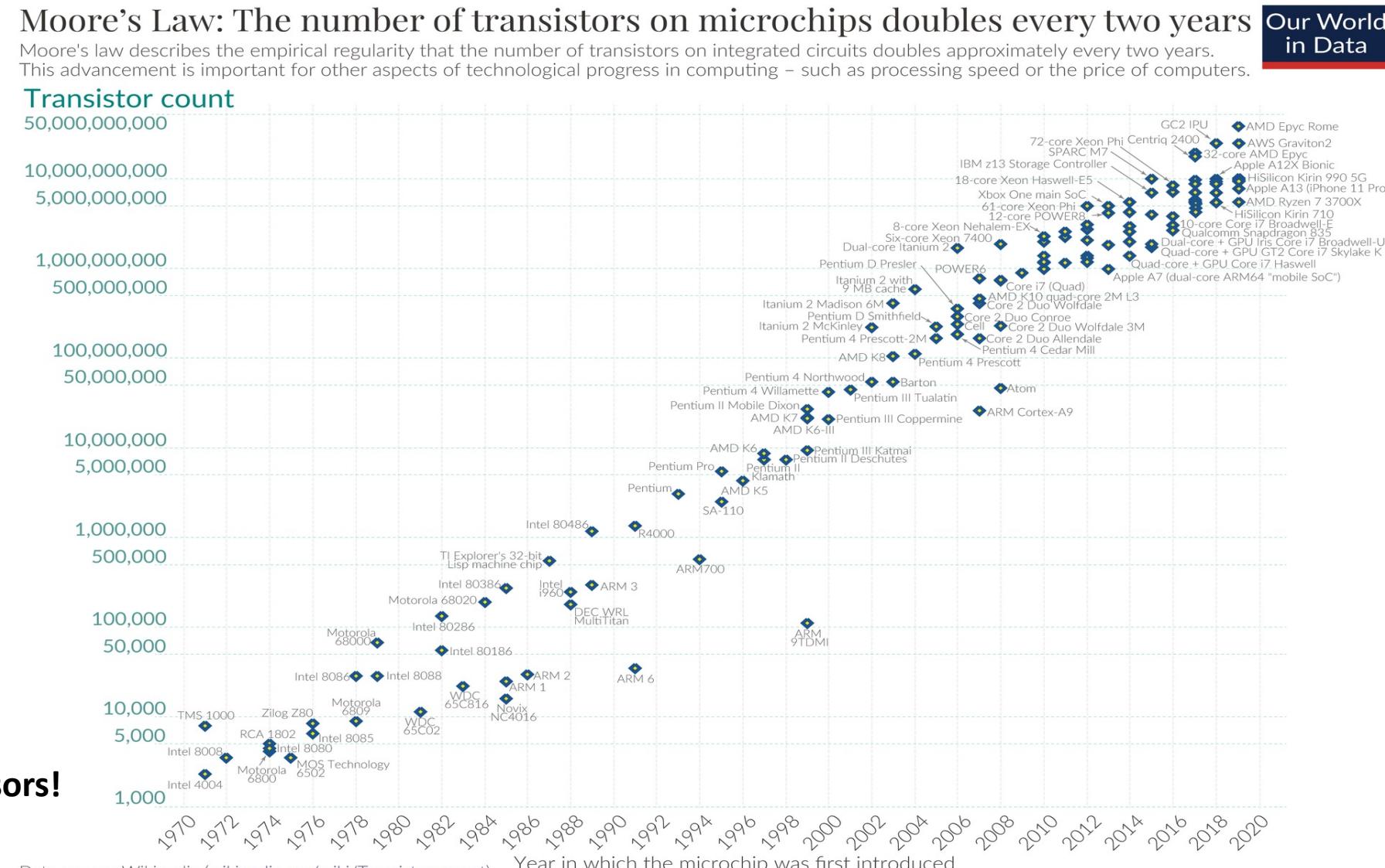
**~40 billion transistors**

1008 mm<sup>2</sup>



## 4-5 nm sized-transistor

**This is a big step....in modern processors!**

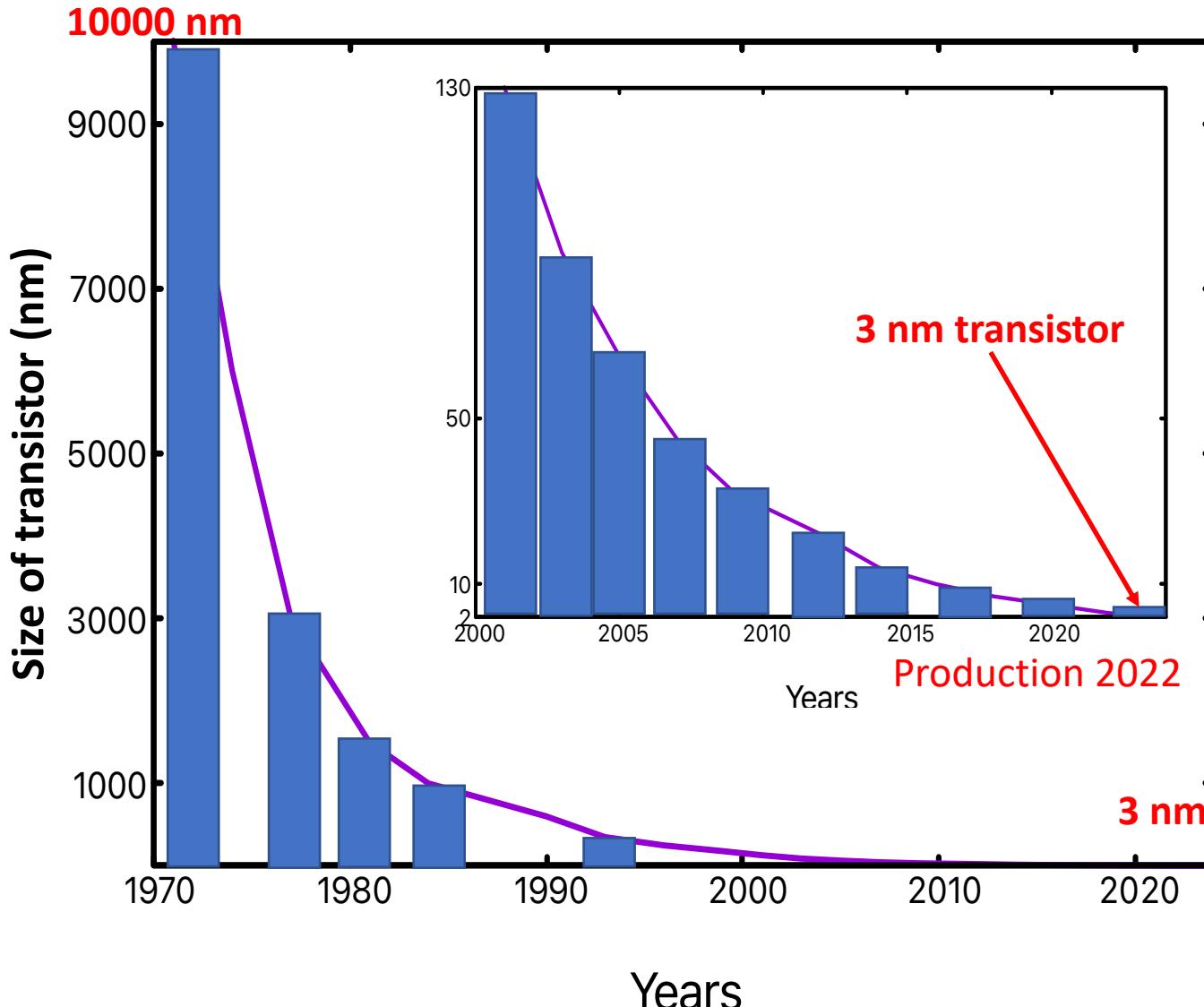


Year in which the microchip was first introduced

OurWorldinData.org - Research and data to make progress against the world's largest problems.

Licensed under CC-BY by the authors Hannah Ritchie and Max Rose

# Evolution of transistors: last 50 years



About 3334 times smaller

BUT is it possible to go below nm....??

Perspective:

## Towards PetaHz( $10^6$ GHz)-CPU with pico-m ( $10^{-3}$ nm) sized-transistors

### Atomic engineering

LETTERS

PUBLISHED ONLINE: 19 FEBRUARY 2012 | DOI: 10.1038/NNANO.2012.21

nature  
nanotechnology

### A single-atom transistor

Martin Ruechsl<sup>1</sup>, Jill A. Miwa<sup>1</sup>, Sudhasatta Mahapatra<sup>1</sup>, Hoon Ryu<sup>2</sup>, Sunhee Lee<sup>3</sup>,  
Oliver Warschkow<sup>4</sup>, Lloyd C. L. Hollenberg<sup>5</sup>, Gerhard Klimeck<sup>3</sup> and Michelle Y. Simmons<sup>1\*</sup>

**Fabrication of working devices such as transistors with extremely short gate lengths requires the ability to position individual atoms in materials with atomic precision.**

Ultimately to make atomic-scale logic circuits that operate at the picometer ( $10^{-3}$  nm)-length scale.

This is a breakthrough in physical sciences!

# Perspective:

## Towards PetaHz( $10^6$ GHz)-CPU with pico-m ( $10^{-3}$ nm) sized-transistors

LETTER

RESEARCH LETTER

doi:10.1038/nature19821

### Multi-petahertz electronic metrology

M. Garg<sup>1</sup>, M. Zhan<sup>1</sup>, T. T. Luu<sup>1</sup>, H. Lakhotia<sup>1</sup>, T. Klostermann<sup>1</sup>, A. Guggenmos<sup>1</sup> & E. Goulielmakis<sup>1</sup>

The speed limit of electronics (signal processing) is determined by the frequency of electric current. The use of light to drive electrons promises to access to vastly higher frequencies. (increase the bandwidth of electronics).

Improving the performance by 6 order of magnitude. From GHz (clock rate) towards PHz-based processors.

This is a breakthrough in physical sciences!

### Atomic engineering

LETTERS

PUBLISHED ONLINE: 19 FEBRUARY 2012 | DOI: 10.1038/NNANO.2012.21

nature  
nanotechnology

### A single-atom transistor

Martin Rueckle<sup>1</sup>, Jill A. Miwa<sup>1</sup>, Sudhasatta Mahapatra<sup>1</sup>, Hoon Ryu<sup>2</sup>, Sunhee Lee<sup>3</sup>, Oliver Warschka<sup>4</sup>, Lloyd C. L. Hollenberg<sup>5</sup>, Gerhard Klimeck<sup>3</sup> and Michelle Y. Simmons<sup>1\*</sup>

Fabrication of working devices such as transistors with extremely short gate lengths requires the ability to position individual atoms in materials with atomic precision. Ultimately to make atomic-scale logic circuits that operate at the picometer ( $10^{-3}$  nm)-length scale.

# Performance of a computer

Supercomputer



Cluster



Computer  
(or node/server)



# Performance of a computer

Supercomputer

- The performance of a processor is measured by the quantity:

FLOPS (Floating-Point Operations Per Second).

- It is a measure of the speed of a computer to perform arithmetic operations.

- For a single processor:

$$\text{FLOPS} = (\text{Clock speed}) \times (\text{cores}) \times (\text{FLOPs/cycle}) = \text{Peak performance}$$

FLOP is a way of encoding real numbers (i.e. DB 64bit or SP 32bit...)

- 1 GigaFLOPS: processor can handle **billion floating-point (64 bit) operations every second**.
- For matching: 1GigaFLOPS ~performing one calculation every second for 31.69 years.
- 1 TeraFLOPS =  $10^{12}$  calculations per second.
- 1 PetaFLOPS =  $10^{15}$  calculations per second.



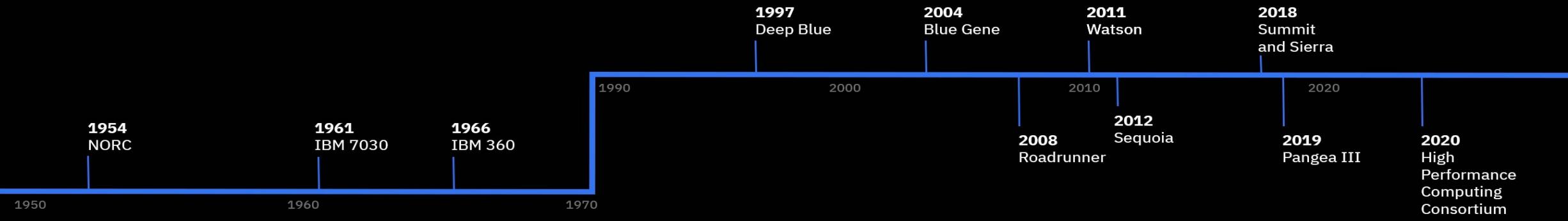
Cluster



Computer  
(or node/server)



# IBM Supercomputing timeline



**1954**  
The Naval Ordnance Research Calculator helped forecast weather and performed other complex calculations.



**1961**  
The IBM 7030 was capable of 2 million operations per second.



**1966**  
The IBM 360 and its successors helped power NASA's Apollo program.



**1997**  
Deep Blue wins its match with chess grandmaster Garry Kasparov.



**2004**  
Blue Gene ushers in a new era of high-performance computing as it helps biologists explore gene development.



**2008**  
Built for Los Alamos National Laboratory, Roadrunner is the first supercomputer in the world to reach petaflop speed.



**2011**  
Watson beats human competitors on Jeopardy!, earning a million-dollar jackpot for charity.



**2012**  
Sequoia, the third-generation Blue Gene system, reaches speeds of 16.32 petaflops.



**2018**  
Summit begins work at Oak Ridge National Laboratory; a sister machine, Sierra, launches at Lawrence Livermore National Laboratory.



**2019**  
IBM builds Pangea III, the world's most powerful commercial supercomputer, for Total to accurately locate new energy resources.



**2020**  
IBM helps launch the COVID-19 High Performance Computing Consortium to research the COVID-19 virus and its potential cures.

**MegaFLOPS**  
Million operation/s  
**1<sup>st</sup> supercomputer**

**A new era of HPC**

**PetaFLOPS**  
Quadrillion

**1<sup>st</sup> P-EFLOPS supercomputer**

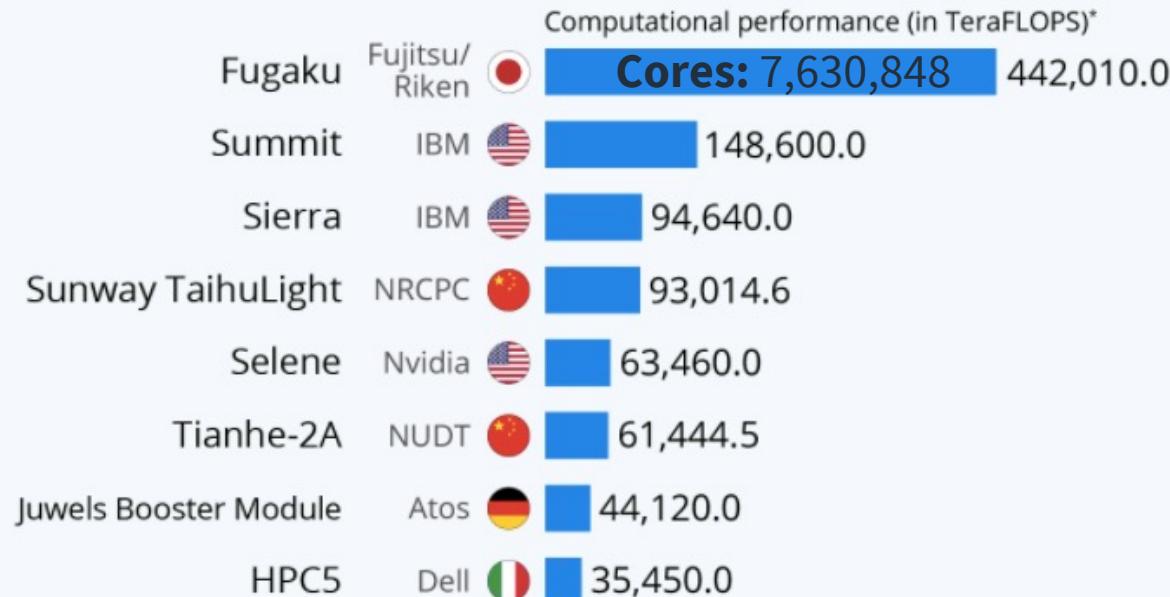
**16 PetaFLOPS**

**~500 PetaFLOPS**



# The World's Top Supercomputers

Computational performance of the most powerful supercomputers (as of November 2020)



\* FLOPS=floating point operations per second, i.e. the number of basic mathematical operations a computer can perform in a second

Source: Top500.org



statista

<https://www.fujitsu.com/>



The Fugaku (**158,976 nodes**) compute system was designed and built by Fujitsu and RIKEN. They take the first place worldwide in TOP500.

**Peak performance  
537 PetaFLOPS**

=537 (quadrillion)  $10^{15}$  (64 bit) operations/s  
Exceeding Summit by more 3x.

# LUMI is one of the fastest supercomputers in the world



Taken from <https://www.lumi-supercomputer.eu/>

## Peak Performance

550 Petaflop/s

~550 quadrillion  
calculations per second

Computing power equivalent to

1 500 000



Modern laptop computers

- **LUMI** (Large Unified Modern Infrastructure):
- **LUMI** is located in a data center in Kajaani, **Finland**.
- Funded by the **EuroHPC JU** (50%) and a **consortium of 10 countries**.
- **LUMI consortium**: Finland, Belgium, The Czech republic, Denmark, Estonia, **Norway**, Poland, Sweden, Switzerland and Iceland.

About the size of a tennis court



Weight around 150 000 Kg

## Phase I: CPU Partition (LUMI-C)

- **1536 compute nodes** with 2x AMD EPYC 7763 (Milan)
- 128 cores per node, **196 608 cores** total
- 256 GiB of memory per node
- Some nodes with 512 GiB and 1 TiB



## Phase 2: GPU Partition (LUMI-G)

- Next generation AMD Instinct GPUs (**2560 nodes** each with **8 GPUs (MI250X)**)
- 550 Pflops peak performance
- The fastest supercomputer in Europe
- Available summer (August-September) 2022

Norway's share (2 %: ~ 4000 cores on LUMI-C, ~ 200 GPUs on LUMI-G)

EuroHPC JU's share (50 %: ~ 100k cores on LUMI-C, ~ 5000 GPUs on LUMI-G)

# What Supercomputers can be used for ?

To solve major challenges in the world.

Supercomputer

platform

High-Performance Computing (HPC)



Artificial Intelligence (AI)



To solve complicated problems in physical sciences engineering such that:

- Exploring the **boundaries of quantum chemistry** (first project LUMI).
- Predicting the structure of proteins using data-driven methodologies (ML, DL).
- **Understanding the functionality of COVID-19 virus & potential cures.**
- Designing new molecules with unique functionality for modern technology.
- **Delivering reliable weather and climate predictions.**

# Conclusion

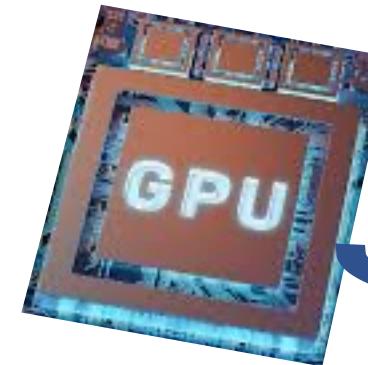
- The **lifetime** of high-performance **hardware** is less than **five years**, while **software can be used for decades**.
- **Software investments** should therefore provide more **flexibility to new and fast evolving technology**.

- CPU & GPU –architectures
- Programming models
- Parallelism in OpenACC

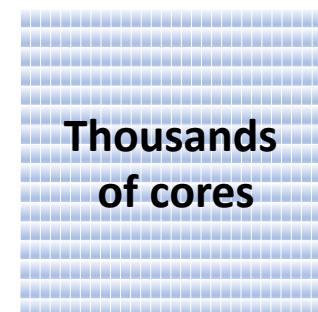
# Outline

- CPU & GPU –architectures
- Programming models
- Parallelism in OpenACC

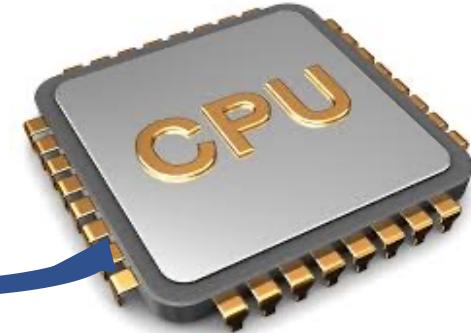
## Heterogenous computing



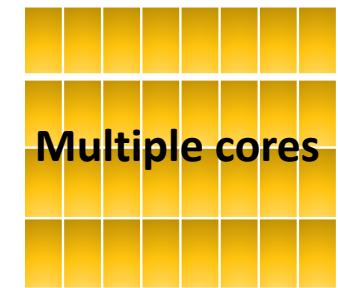
GPU-Device



GPU: Graphics Processing Unit  
CPU: Central Processing Unit



CPU-Host

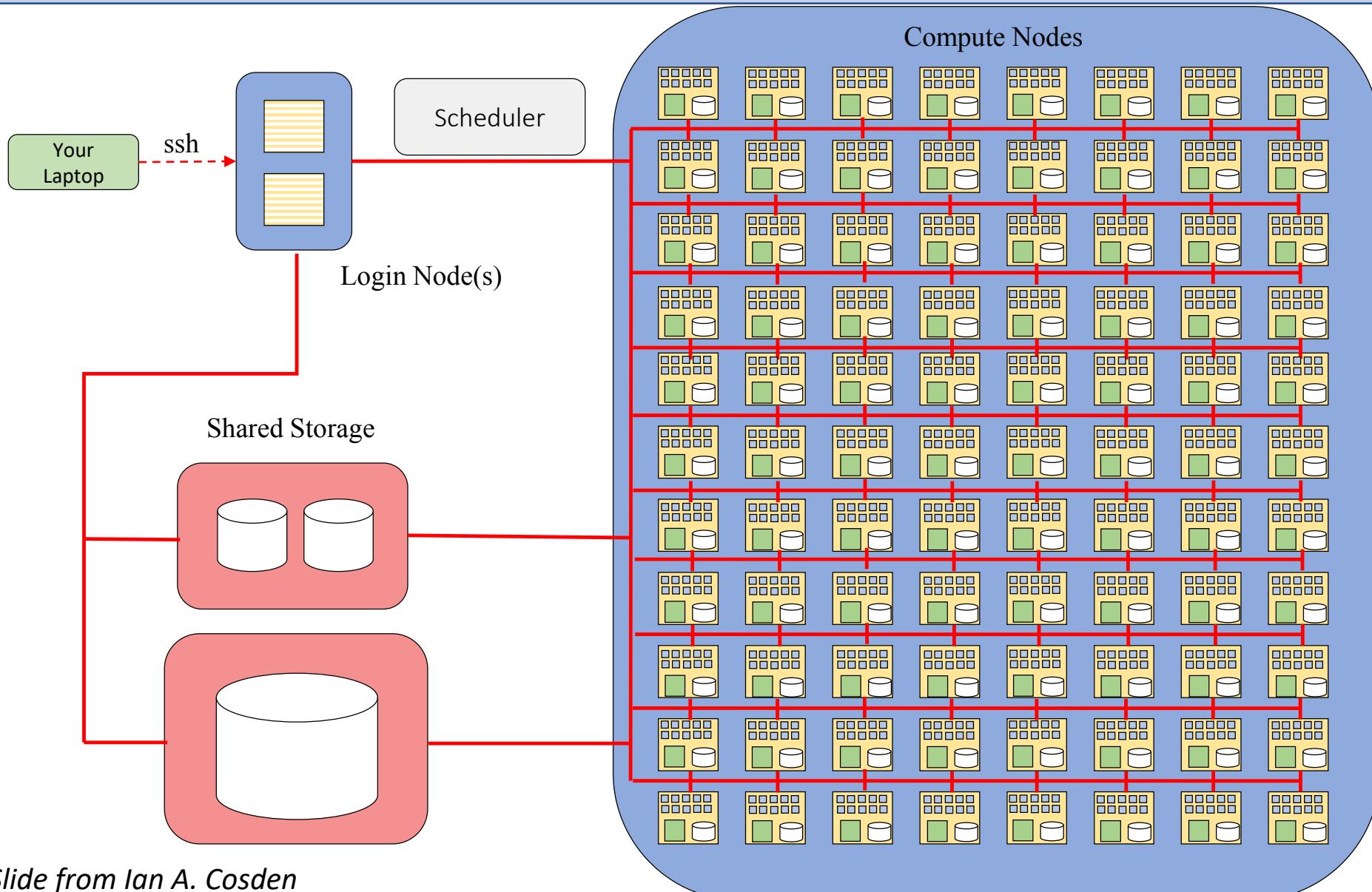


Programming model  
To offload a code region  
to a GPU-device

```
do i=1,n  
do j=1,m  
A(i,j) = B(i,j) + C(i,j)  
enddo  
enddo
```

# **CPU-architecture**

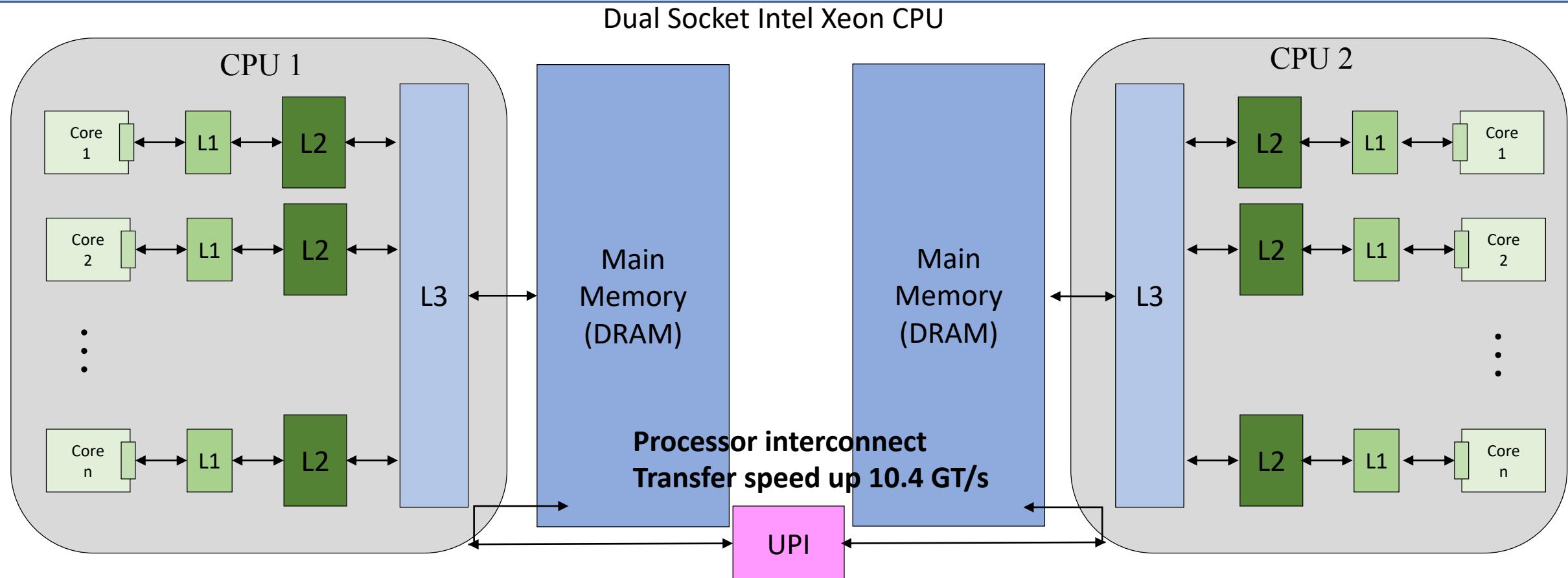
# Architecture of Cluster



Slide from Ian A. Cosden

[https://indico.cern.ch/event/814979/contributions/3401193/attachments/1831477/3105158/comp\\_arch\\_codas\\_2019.pdf](https://indico.cern.ch/event/814979/contributions/3401193/attachments/1831477/3105158/comp_arch_codas_2019.pdf)

# CPU-Architecture: ex. Betzy 128 cores



Access L1  
Memory is  
Faster than L2

	Registers	L1 Cache	L2 Cache	L3 Cache	DRAM	Disk
Speed	1 cycle	~4 cycles	~10 cycles	~30 cycles	~200 cycles	10ms
Size	< KB per core	~32 KB per core	~256 KB per core	~35 MB per socket	~100 GB per socket	TB

# **GPU-architecture**

# Architecture of NVIDIA GPU devices



Figure 1. NVIDIA Tesla V100 SXM2 Module with Volta GV100 GPU



Figure 2. Volta GV100 full GPU with 84 SM Units

# Architecture of NVIDIA GPU devices



Figure 1. NVIDIA Tesla V100 SXM2 Module with Volta GV100 GPU

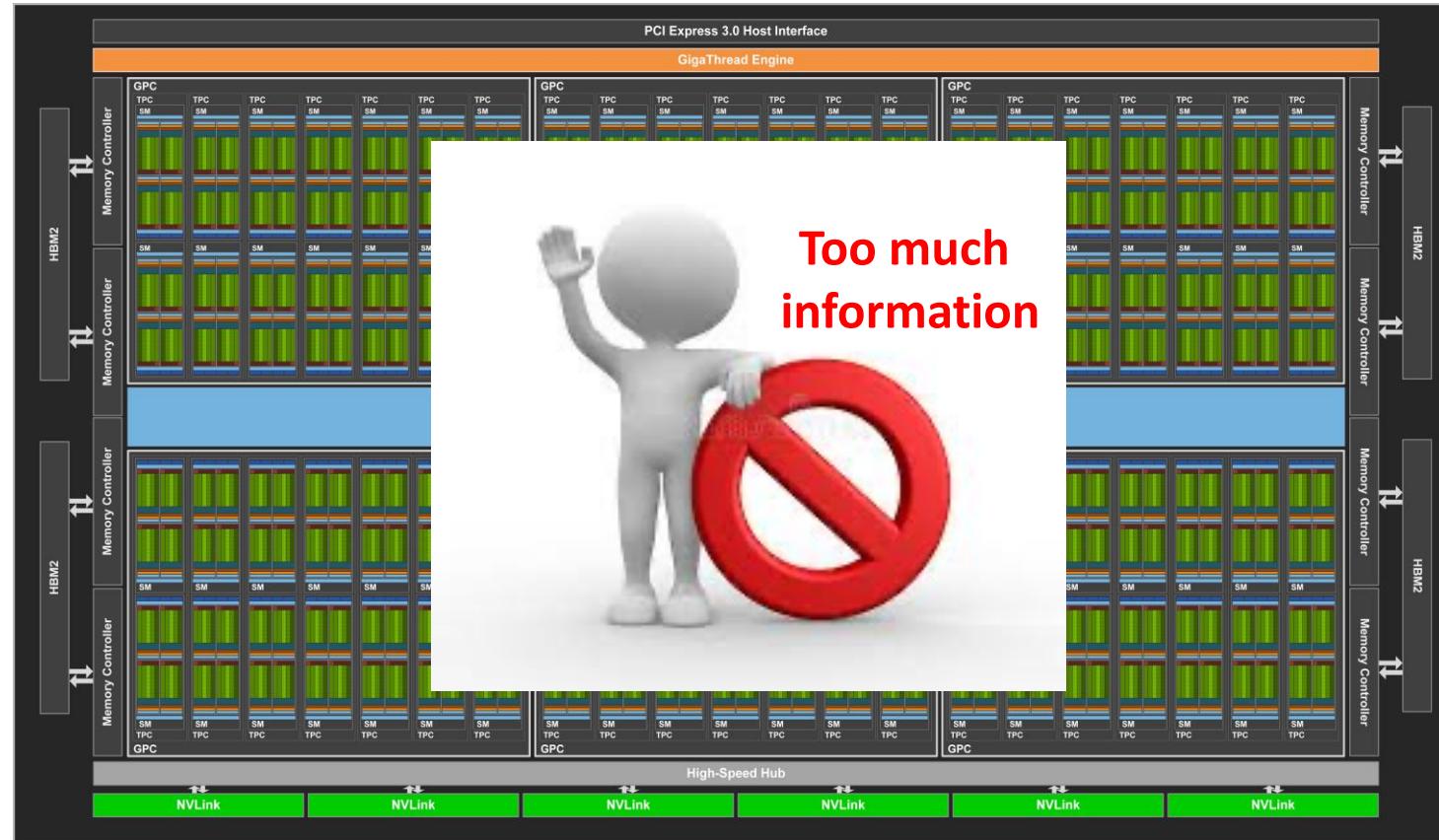
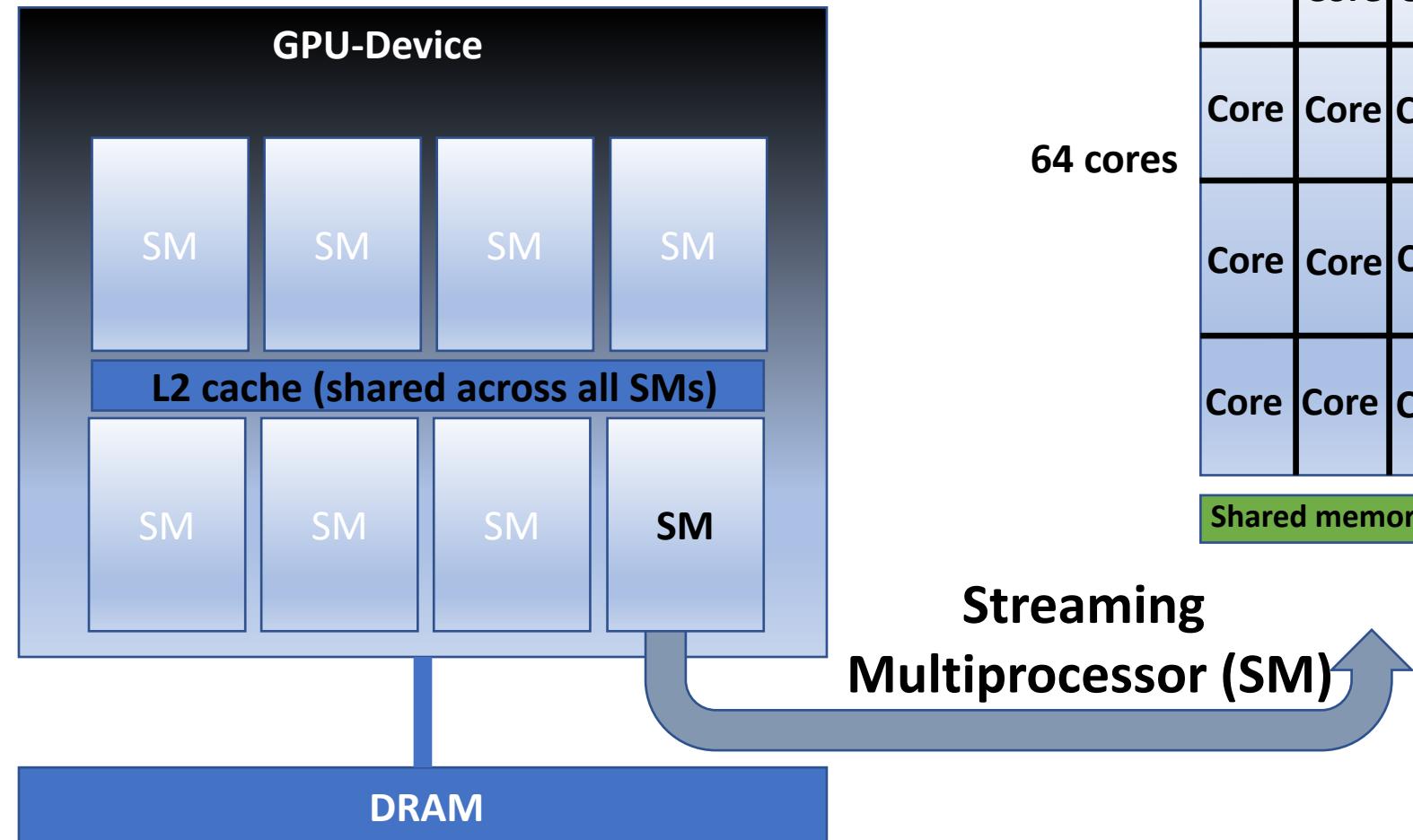


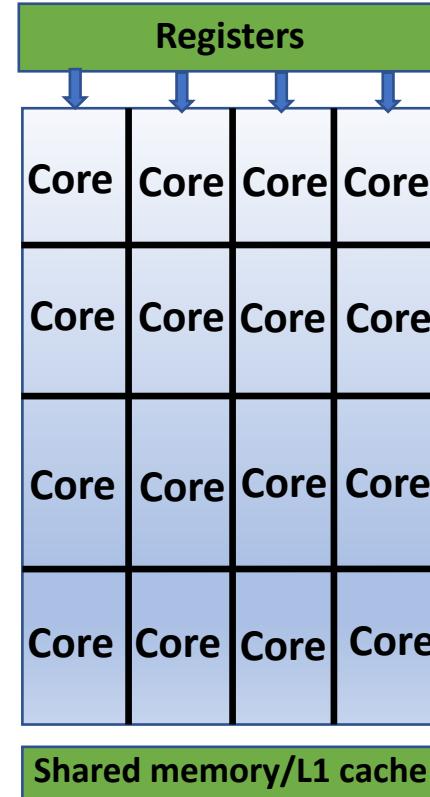
Figure 2. Volta GV100 full GPU with 84 SM Units

# Architecture of NVIDIA-GPU devices (simplified version)

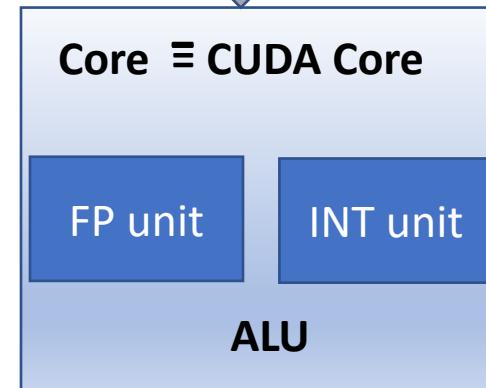


64 cores

Streaming  
Multiprocessor (SM)



Data are stored temporarily. Registers are private to each thread.



Core ≡ CUDA Core

FP unit

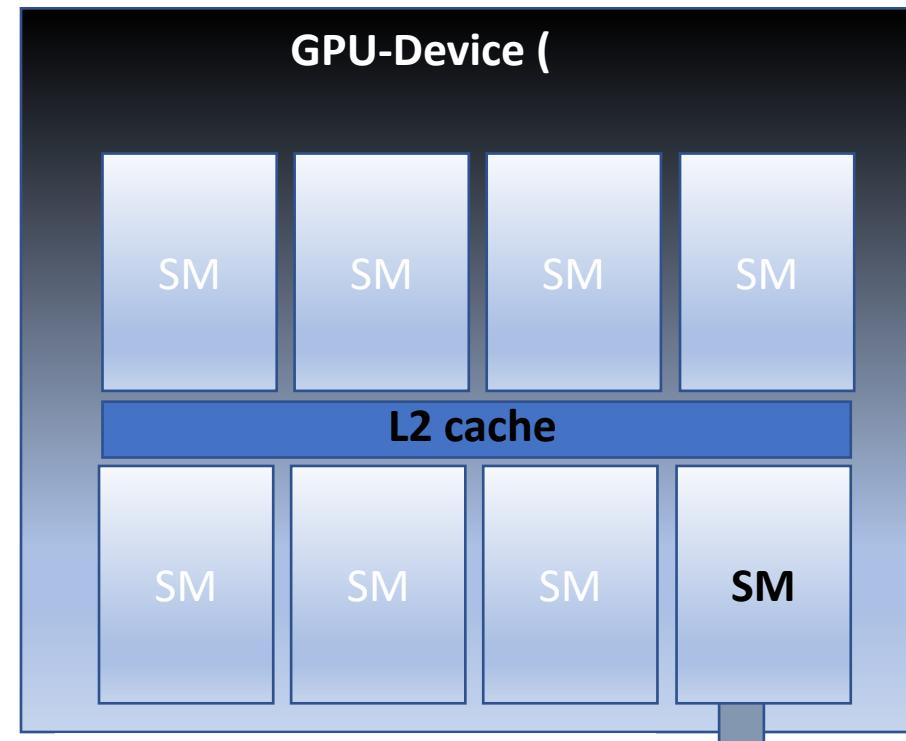
INT unit

ALU

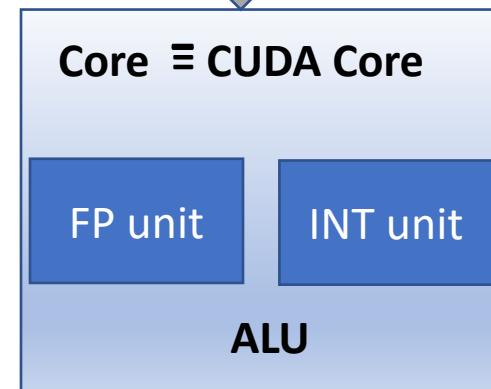
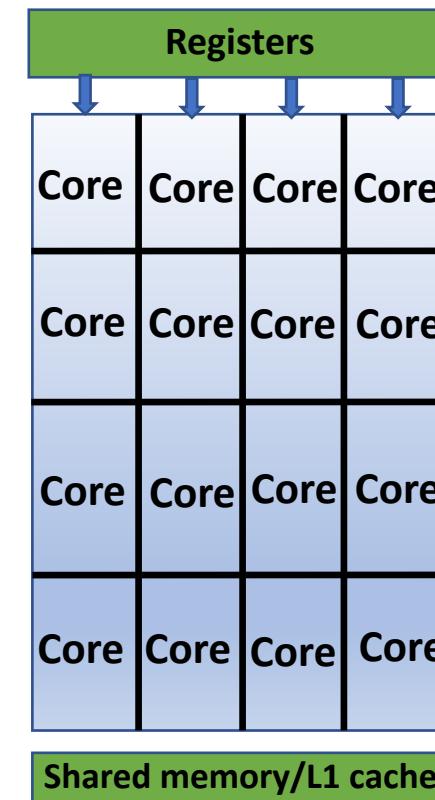
All threads share L1 cache.

- Core includes ALU. Various logical operations for computations (FP32, FP64), e.g. addition, multiplication and ...
- ALU executes SIMD instructions. (processing multiple data with the same operation concurrently (in parallel))

# Architecture of NVIDIA-GPU devices (simplified version)



Streaming  
Multiprocessor (SM)



For Tesla A100 GPU: 108 SMs, each SM has:

64 FP32 cores ===== A total of 6912 FP32 CUDA cores

64 INT32 cores ===== A total of 6912 INT32 CUDA cores.



NVIDIA Tesla P100



NVIDIA Tesla V100



NVIDIA Tesla A100

Architecture	Tesla P100 (Pascal)	Tesla GV100 (Volta)	Tesla A100
SMs	56	84	108
NVIDIA CUDA cores	3584	5376	6912
Tensor cores/GPU	NA	672	432
Peak performance	9.3 TFLOPS	15.7 TFLOPS	39 TFLOPS
Transistors	15.3 billion	21.1 billion	54.2 billion
GPU die size	610 mm <sup>2</sup>	815 mm <sup>2</sup>	826 mm <sup>2</sup>

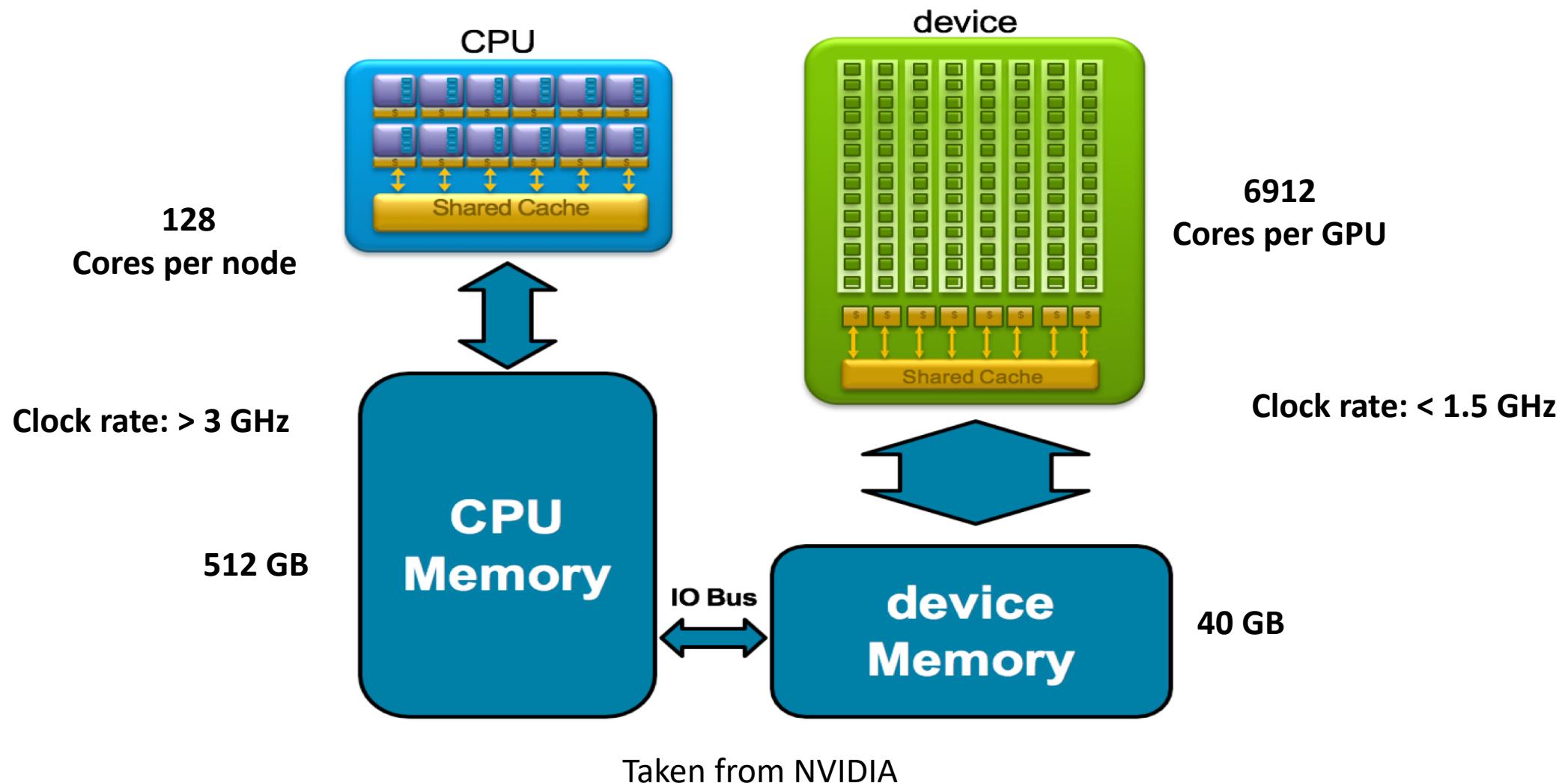


32X Faster Training Throughput  
than a CPU<sup>1</sup>



GPU-Dimension (cm)  
26.7x11x4

# Conclusion: CPU vs GPU



# **Heterogenous programming models**

# Heterogenous programming models

Programming models can be hardware-dependent or hardware-independent

Directive based models:

High-level models

OpenACC

OpenMP

Compiler supports

NVHPC

GCC

Cray (only  
acc .2.0 F)

Clang

Cray

GCC

Icx (Intel)

Hardware

NVIDIA

NVIDIA

NVIDIA

AMD

AMD

AMD

Intel

Low level offload models:

CUDA

Only on NVIDIA

OpenCL

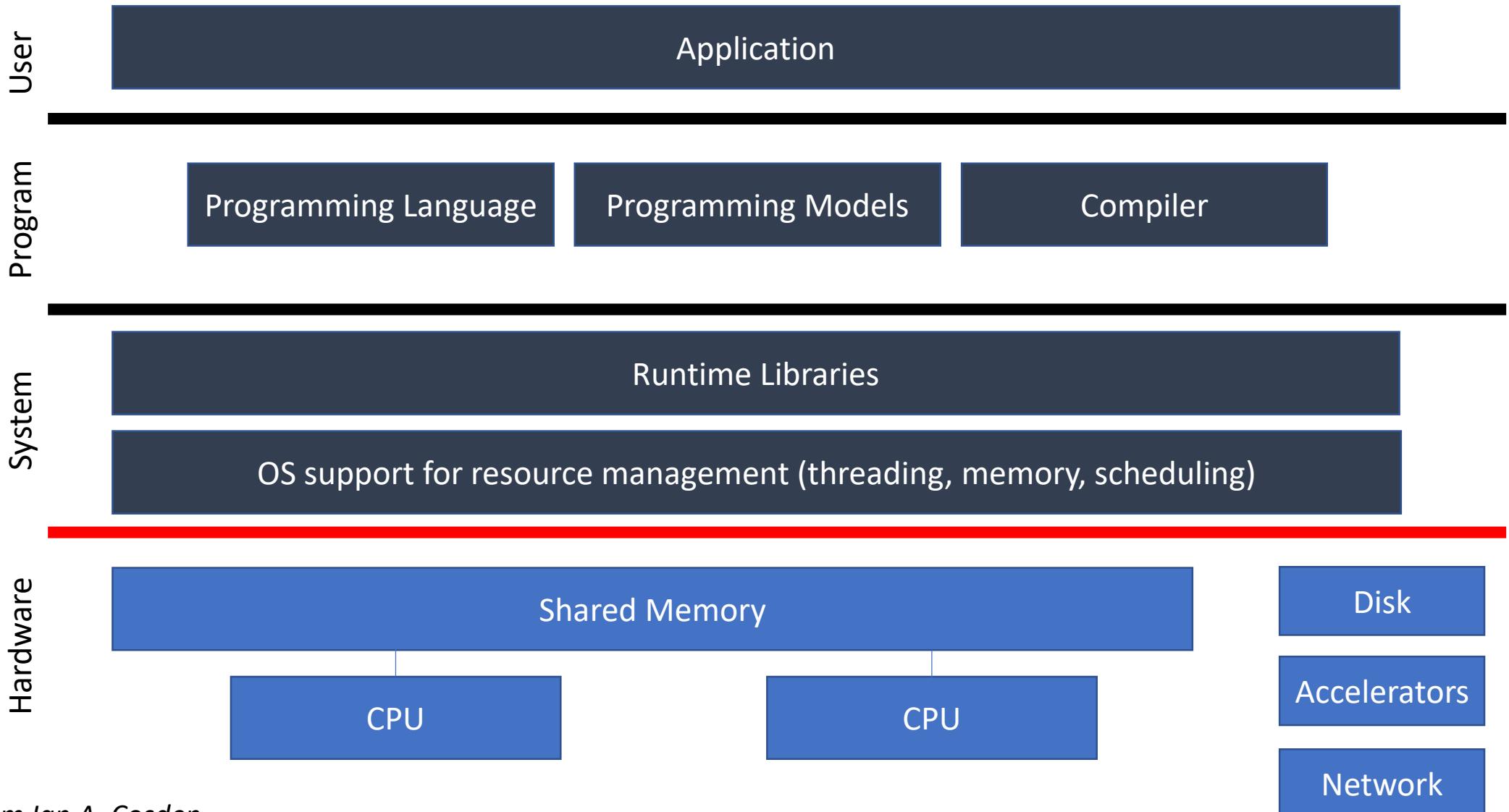
HIP

Only on AMD

Intel GPU: OpenCL  
can be migrated via  
SYCL API migrating  
from CUDA to  
DPC++. DPC++ tool  
is part of the Intel  
OneAPI Toolkit.

Hybrid multi-GPU programming: Combining MPI/OpenMP threading & OpenACC/OpenMP offloading

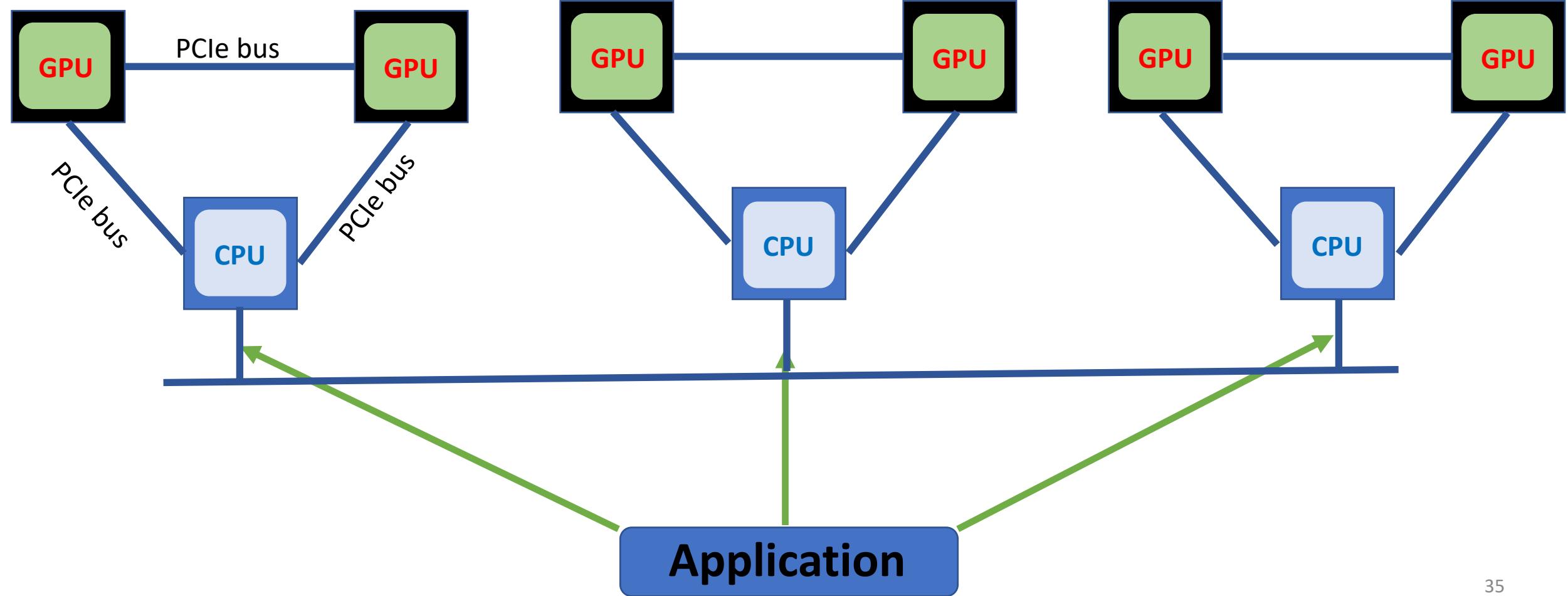
# Hardware-architecture



# Multi-GPU programming

MPI+GPU programming model

GPU-to-GPU communication



# **Parallelism in OpenACC/CUDA**

CUDA= Compute Unified Device Architecture

## **What is OpenACC?**

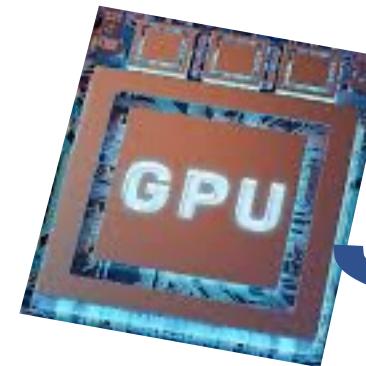
OpenACC is a directive-based approach similar to OpenMP.

It is designed to simplify parallel programming between heterogenous systems CPU/GPU.

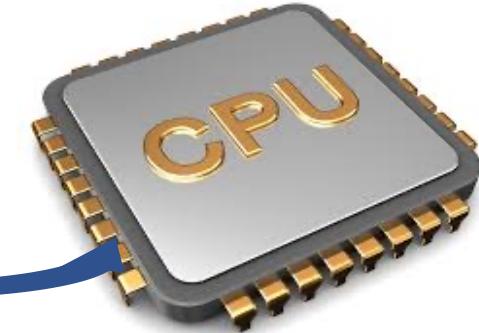
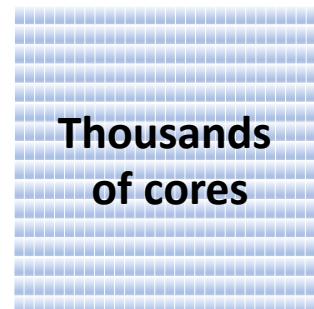
It is API (Application Programming Interface) to communicate between CPU and GPU.

# Parallelism in OpenACC

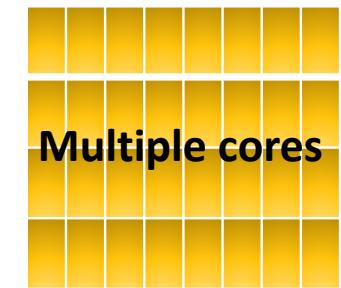
## Heterogenous computing



GPU-Device



CPU-Host



**Programming model**  
To offload a code region  
to a GPU-device

```
do i=1,n  
  do j=1,m  
    A(i,j) = B(i,j) + C(i,j)  
  enddo  
enddo
```

# Parallelism in OpenACC

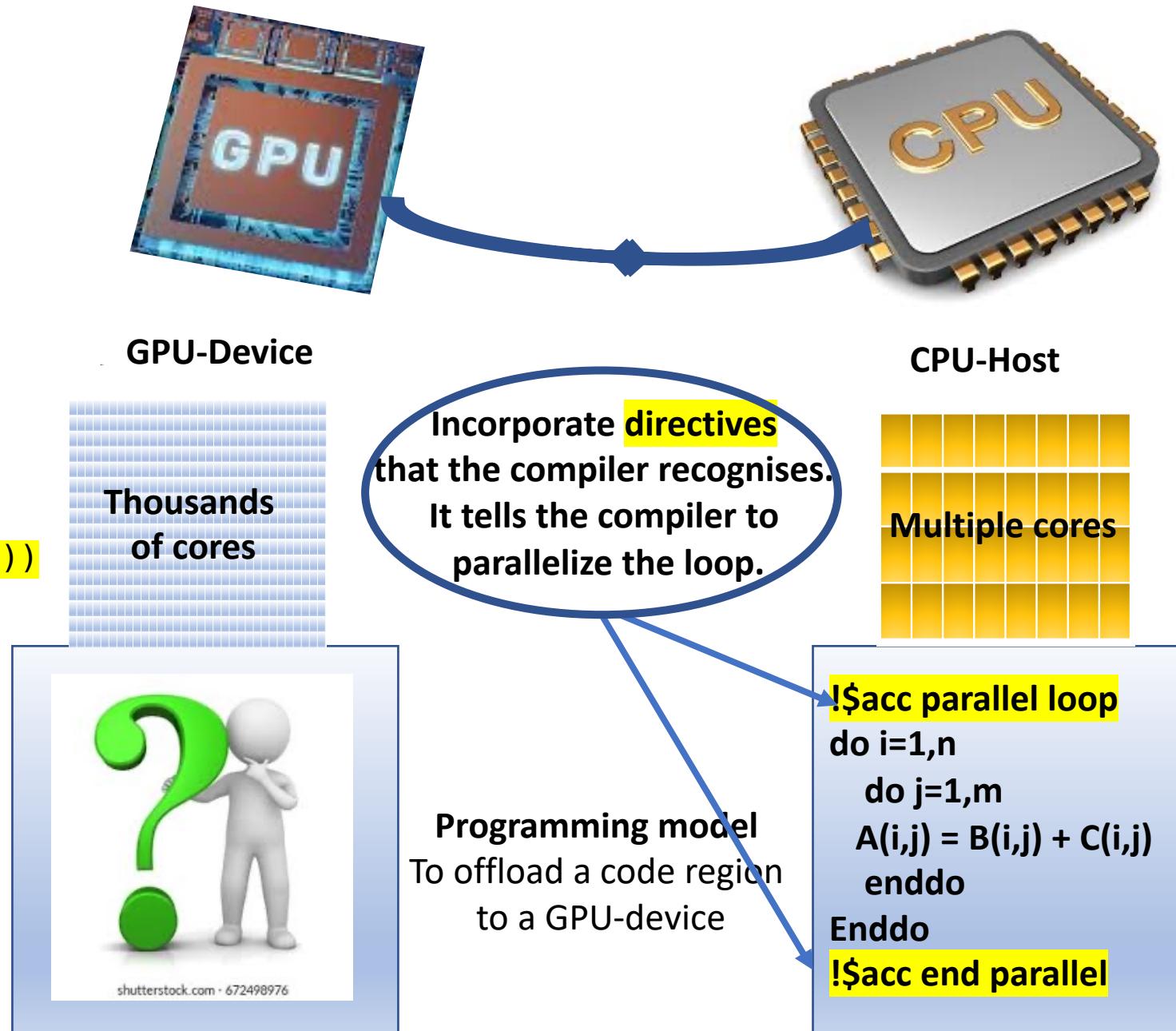
It uses CUDA to exploit parallelism.

## Heterogenous computing

Focus on how parallelism is done.

### Message from the compiler

```
10, Generating Tesla code
11, !$acc loop gang, vector(32), worker(4)
! blockidx%x threadidx%x threadidx%y
12, !$acc loop seq
10, Generating implicit copyin(B(:,:,),C(:,:,))
[if not already present]
Generating implicit copyout(A(:,:,))
[if not already present]
12, Loop is parallelizable
```

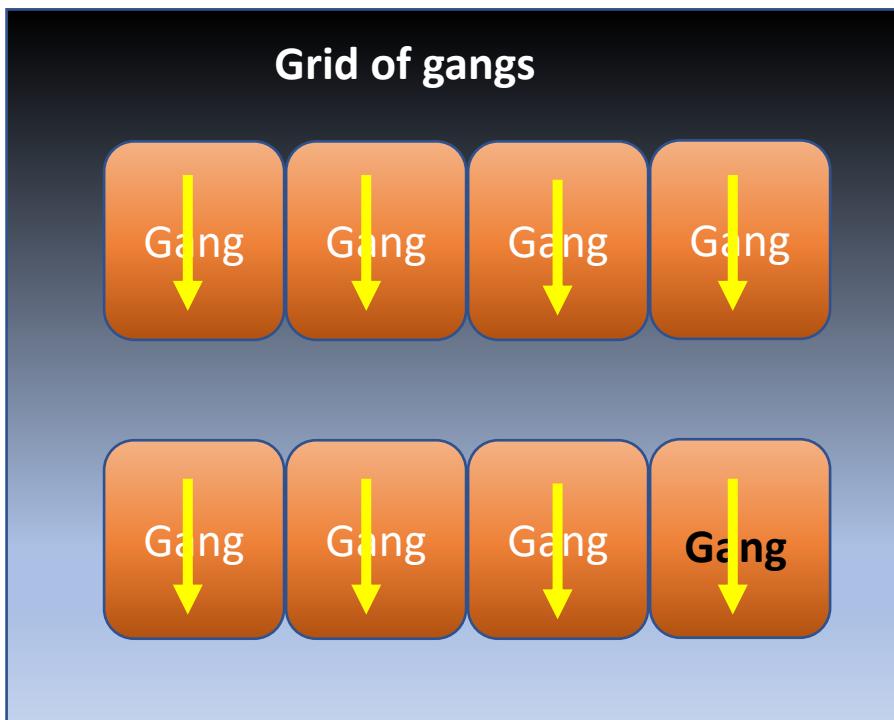


# Parallelism in OpenACC

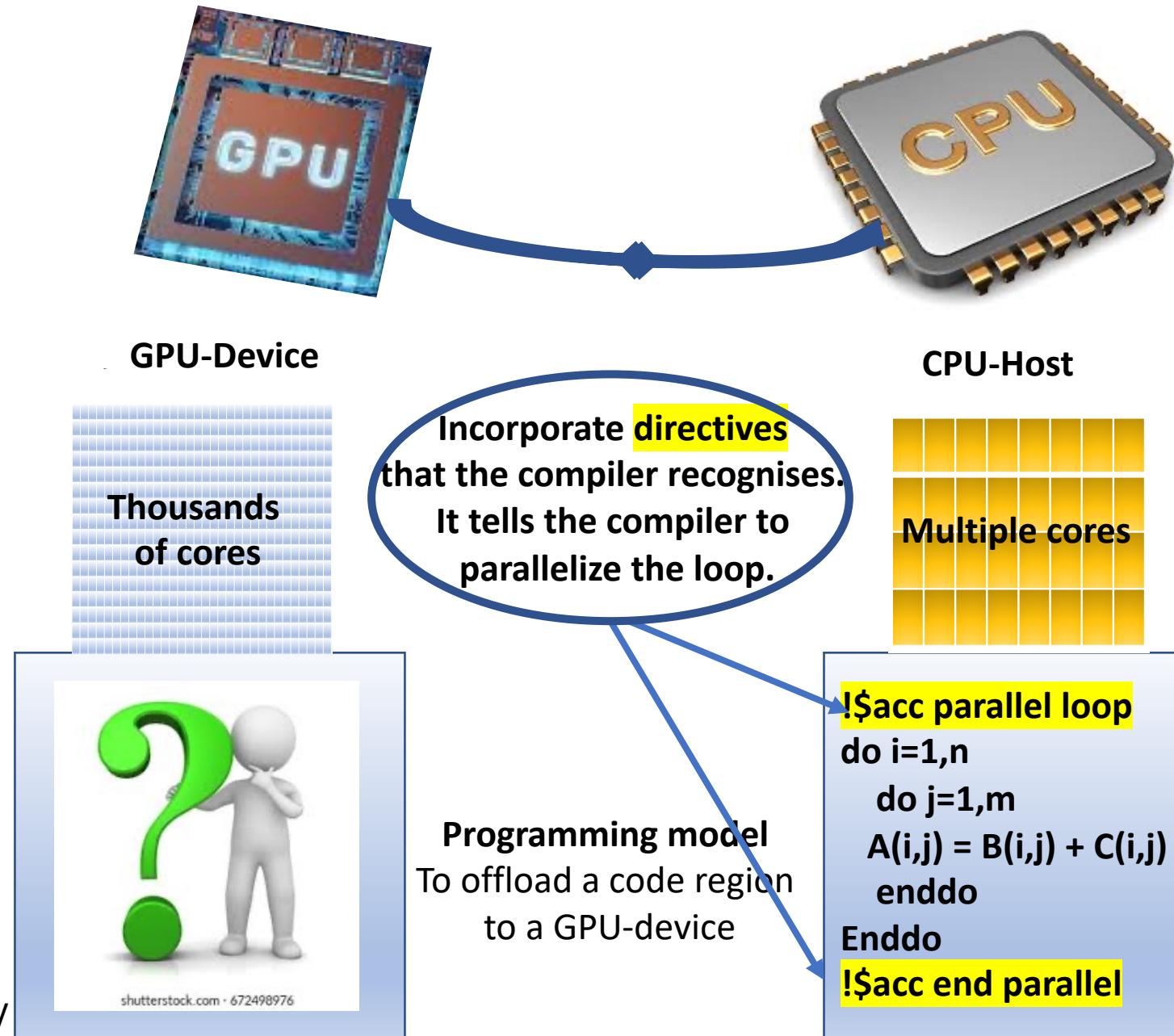
## Heterogenous computing

### Software scheme

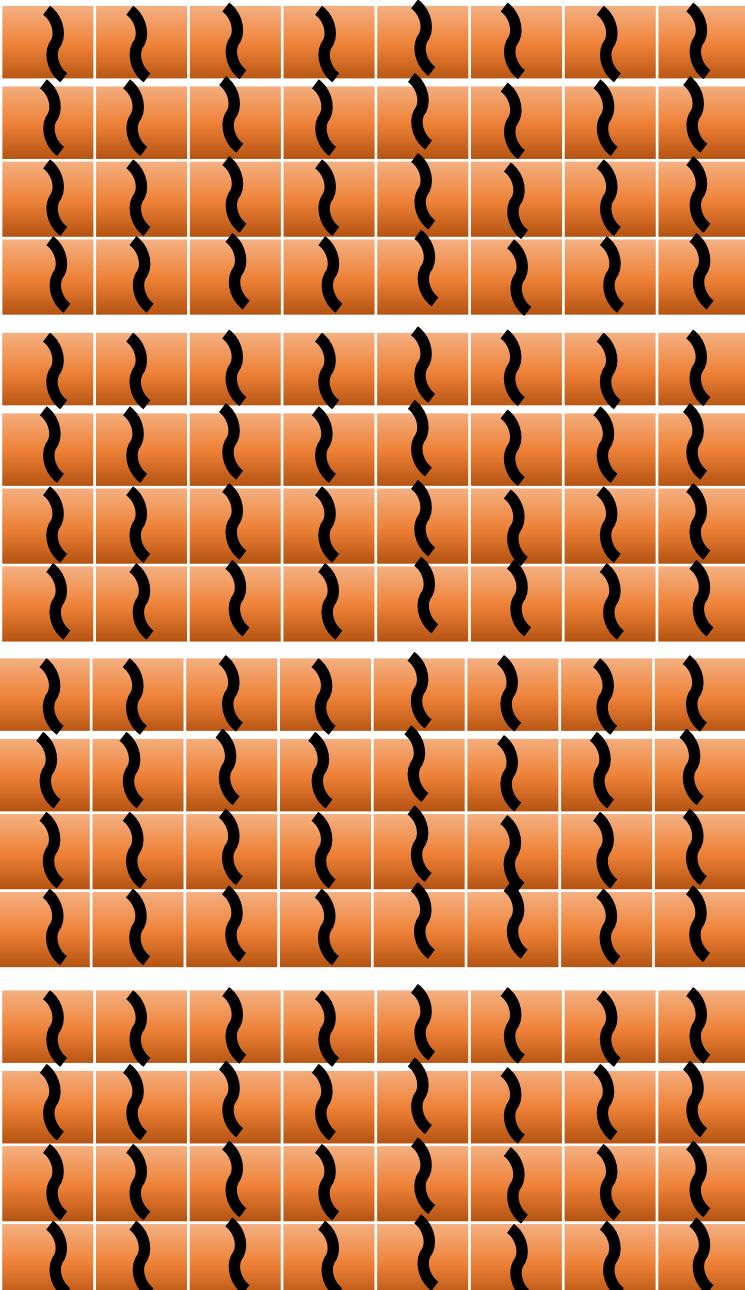
Only the “parallel” construct



**Step 1:** The compiler generates one or more gangs.  
**Step 2:** The offloaded loop gets executed redundantly on each gang (get duplicated).

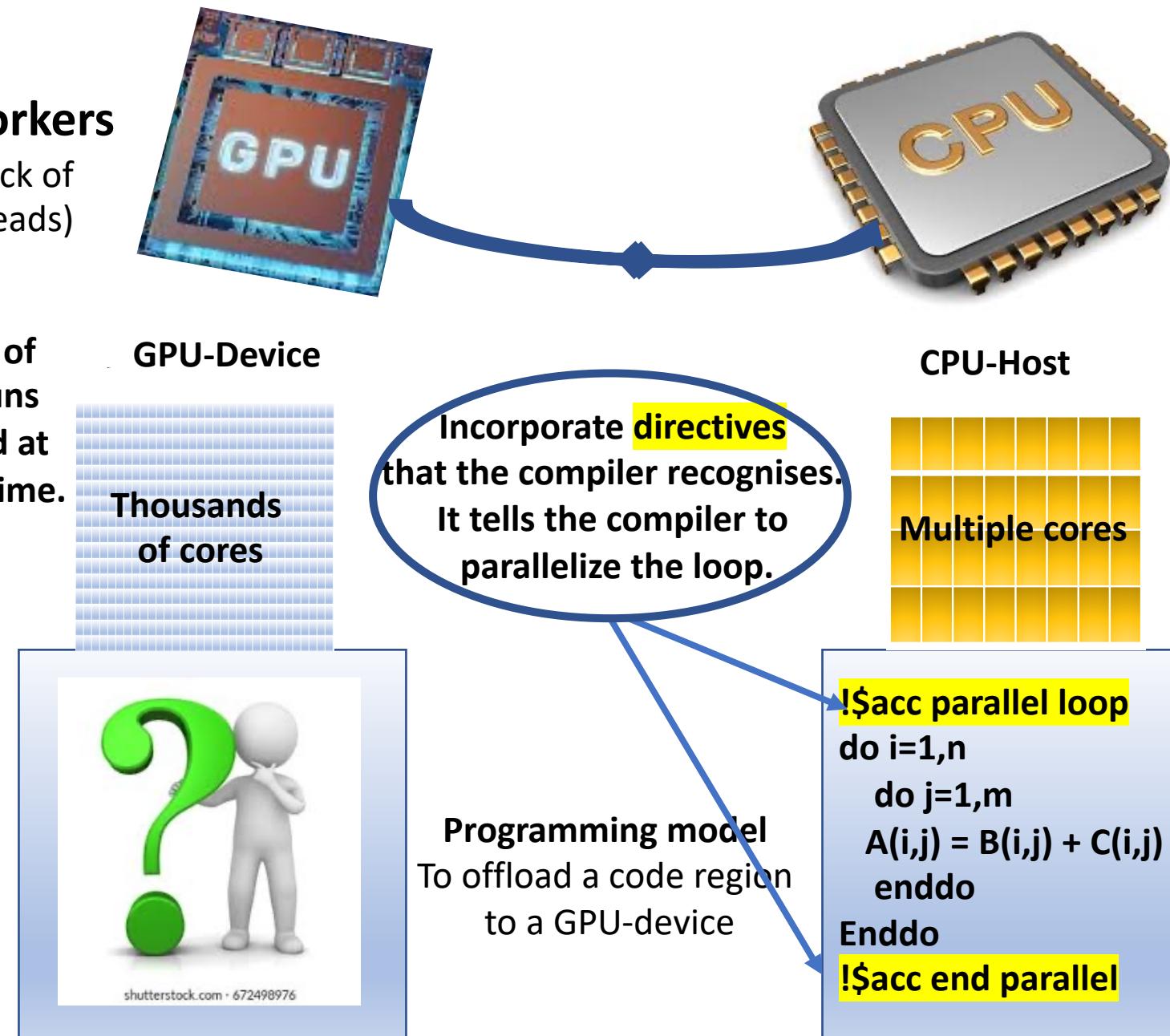


# Parallelism in OpenACC



Each piece of the loop runs by a thread at the same time.

# Heterogenous computing



# Parallelism in OpenACC (it uses CUDA to exploit the parallelism)

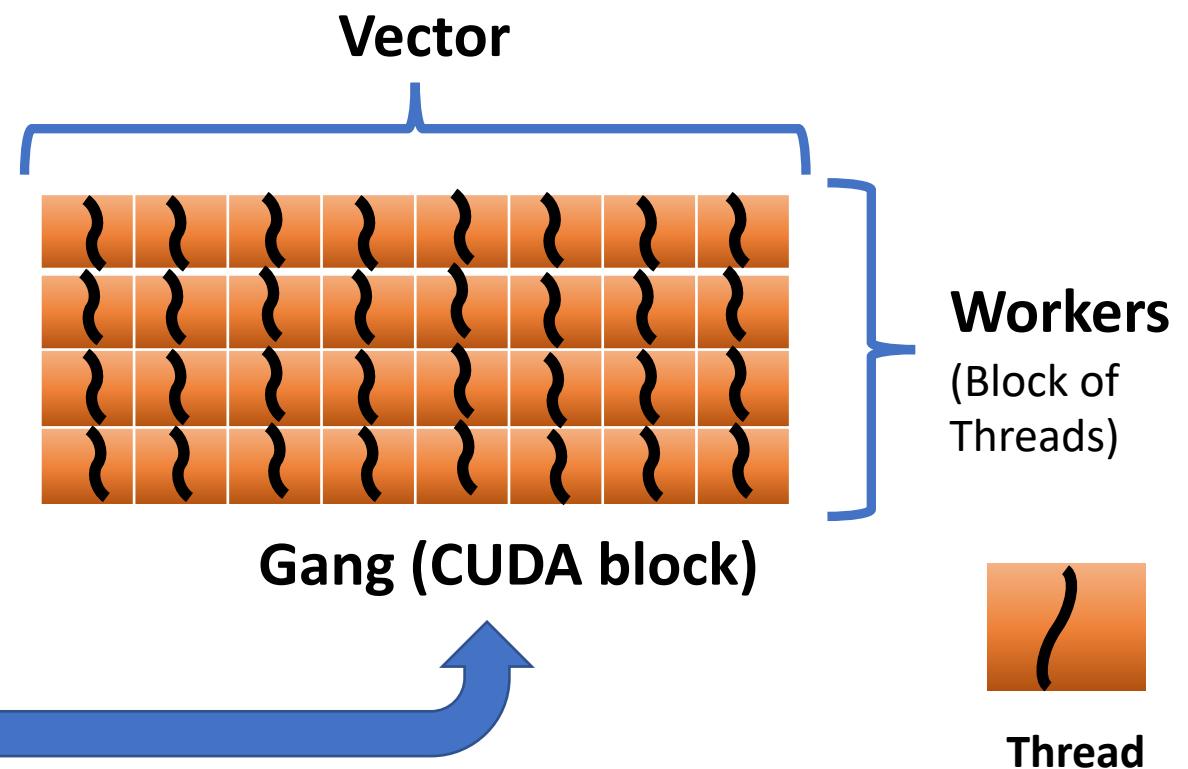
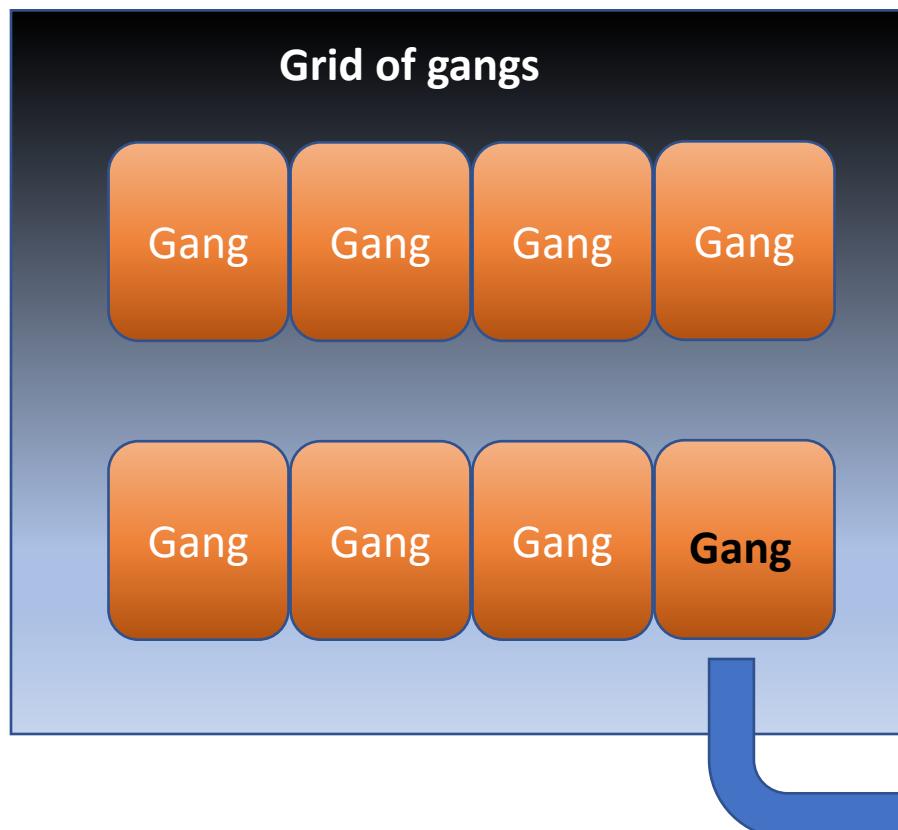
The parallelism in OpenACC is based on three concepts:

- **Gang**: Consists of a set of workers (block of thread)
- **Worker**: threads.
- **Vector**: It tells the OpenACC that it can exploit *vector parallelism*

Each thread has an index: it is used for calculating memory address locations.

vectorization naturally occurs in hardware across a **group of threads** referred to as a **warp (32 threads)** at the **same time**.

## Software scheme



## Mapping a matrix into a device

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Each thread executes a different piece  
of data item (data parallelism).

Each CUDA block has 1024 threads  
(32 warps. A warp=collection of 32 threads  
are executed simultaneously by a SM)

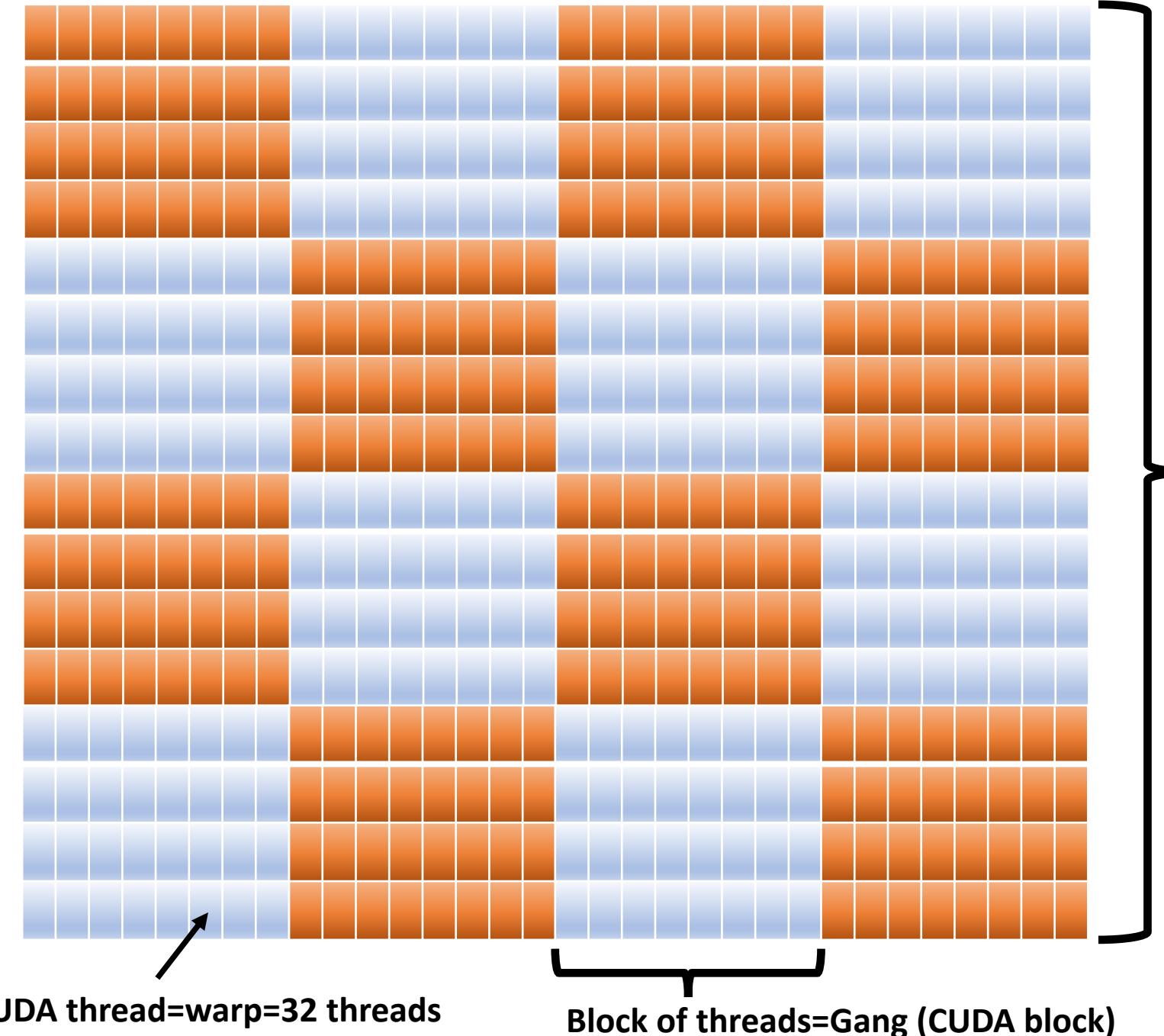
Execution on hardware

CUDA thread

CUDA block

CUDA grid

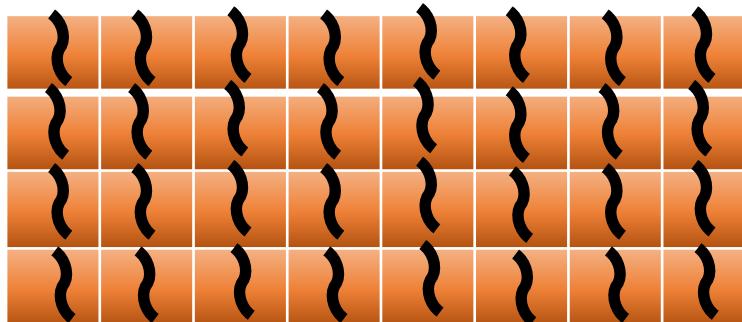
Grid of threads



# Execution on GPU

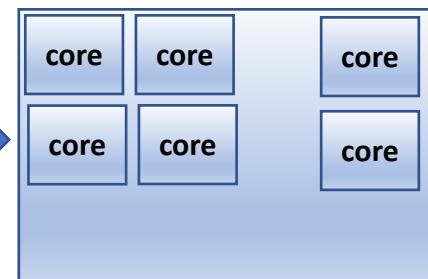
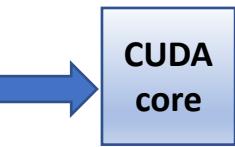
## Software scheme

## Hardware scheme



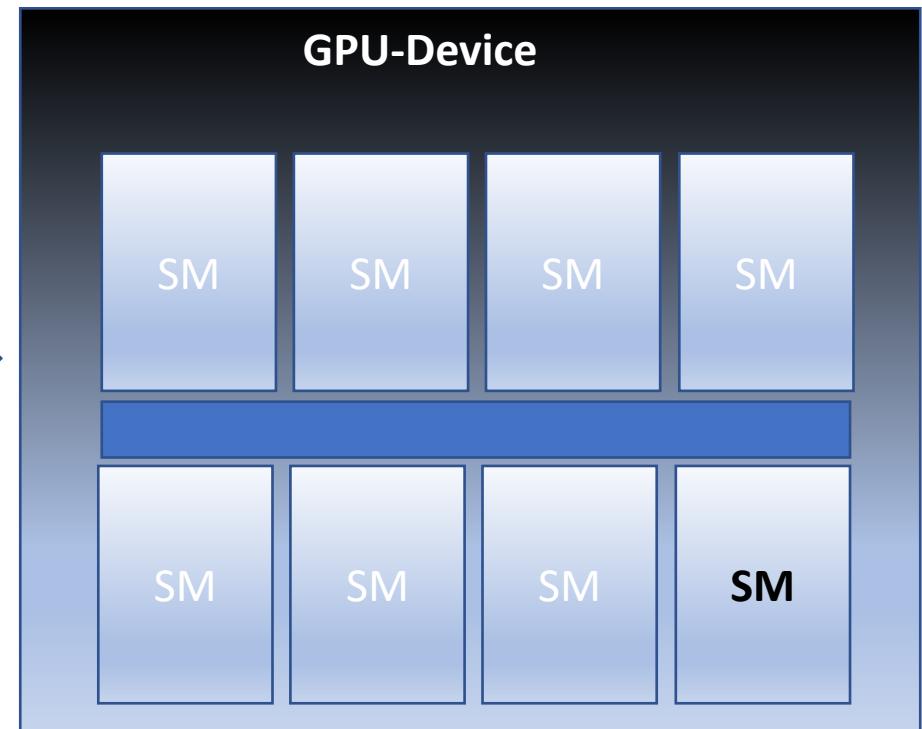
CUDA thread block

**is executed on**



Streaming  
Multiprocessor  
(SM)

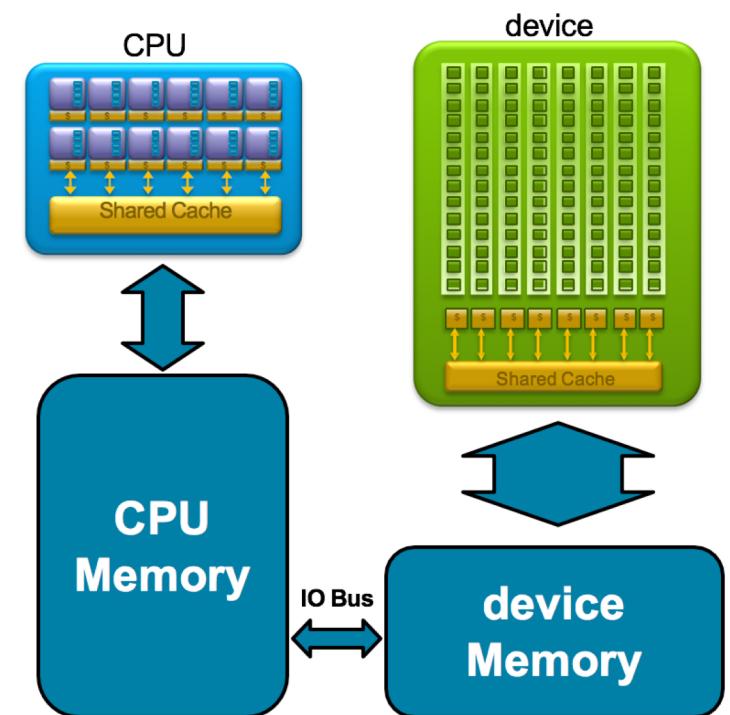
Grid of gangs  
(CUDA kernel grid)



# Conclusion

The parallelism in OpenACC is based on introducing three basic steps:

- Step 1: The compiler generates one or more gangs.
- Step 2: The offloaded loop gets executed redundantly on each gang (get duplicated).
- Step 3: A block of thread will execute different pieces of the offloaded loop.
- ***vector parallelism***: to perform multiple data-parallel operations at the same time.
- vectorization naturally occurs in hardware across a group of threads referred to as a warp at the same time.
- Data transfer between CPU and GPU is time consuming.
- The performance of GPU can be slower than CPU in some cases.
- Bottleneck in GPU-programming: Portability and hardware limitation.
- Converting codes affects the performance.
- Knowledge of GPU-architecture is important for GPU-programming.



Taken from NVIDIA