GP-GPU
and
High Performances Computing

Cours 1
Introduction

TOP 500

Rank	System	Cores	(PFlop/s)	(PFlop/s)	(kW)
1	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	8,699,904	1,194.00	1,679.82	22,703
2	**Supercomputer Fugaku** - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	7,630,848	442.01	537.21	29,899
3	**LUMI** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE EuroHPC/CSC Finland	2,220,288	309.10	428.70	6,016
4	**Leonardo** - BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64 GB, Quad-rail NVIDIA HDR100 Infiniband, Atos EuroHPC/CINECA Italy	1,824,768	238.70	304.47	7,404
5	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM DOE/SC/Oak Ridge National Laboratory United States	2,414,592	148.60	200.79	10,096

HPCG 500

Rank	TOP500 Rank	System	Cores	Rmax (PFlop/s)	HPCG (TFlop/s)
1	2	**Supercomputer Fugaku** - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	7,630,848	442.01	16004.50
2	1	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	8,699,904	1,194.00	14054.00
3	3	**LUMI** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE EuroHPC/CSC Finland	2,220,288	309.10	3408.47
4	4	**Leonardo** - BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64 GB, Quad-rail NVIDIA HDR100 Infiniband, Atos EuroHPC/CINECA Italy	1,824,768	238.70	3113.94
5	5	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM DOE/SC/Oak Ridge National Laboratory United States	2,414,592	148.60	2925.75

Green 500

Rank	TOP500 Rank	System	Cores	Rmax (PFlop/s)	Power (kW)	Energy Efficiency (GFlops/watts)
1	255	**Henri** - ThinkSystem SR670 V2, Intel Xeon Platinum 8362 32C 2.8GHz, NVIDIA H100 80GB PCIe, Infiniband HDR, Lenovo Flatiron Institute United States	8,288	2.88	44	65.396
2	34	**Frontier TDS** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	120,832	19.20	309	62.684
3	12	**Adastra** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE Grand Equipement National de Calcul Intensif - Centre Informatique National de l'Enseignement Suprieur (GENCI-CINES) France	319,072	46.10	921	58.021
4	17	**Setonix – GPU** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE Pawsey Supercomputing Centre, Kensington, Western Australia Australia	181,248	27.16	477	56.983

General information

General course information

Contact - Christophe Picard

- christophe.picard@univ-grenoble-alpes.fr
- Office 174 – IMAG Building – 1st floor
- Email headers: [CGPU]
- Course notes available on my website

Class organization

- Lectures and labs
- One written exam **and** a project

About the project

- ➔ It is **your** project. You **choose** the subject.
- ➔ It can be a group project (2 or 3).
- ➔ It should be related to your major (MMIS, MSIAM, MoSiG).
- ➔ You may reused project from other courses, but you should propose an improvement.
- ➔ It must involve some programming (Python, C, C++, Julia, Fortran).
- ➔ It must involve some design. You are **not allowed** to just use a library for parallelism.
- ➔ You must submit your proposal for project by **October 13th 2023.**

Grading the project

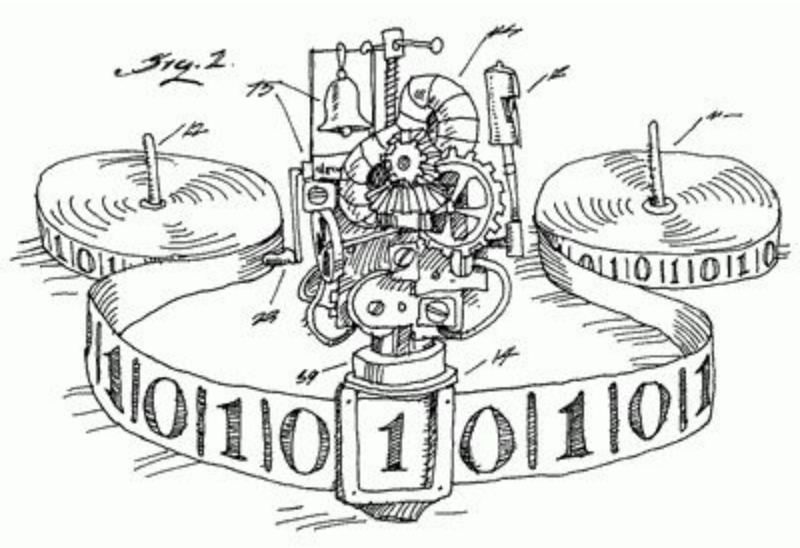
The grade will depend on :

- ➔ the quality of the code.
- ➔ the quality of the report.
- ➔ the design of the parallelism.
- ➔ the performances.
- ➔ the number of students in the project.
- ➔ the difficulty of the subject.

Introduction

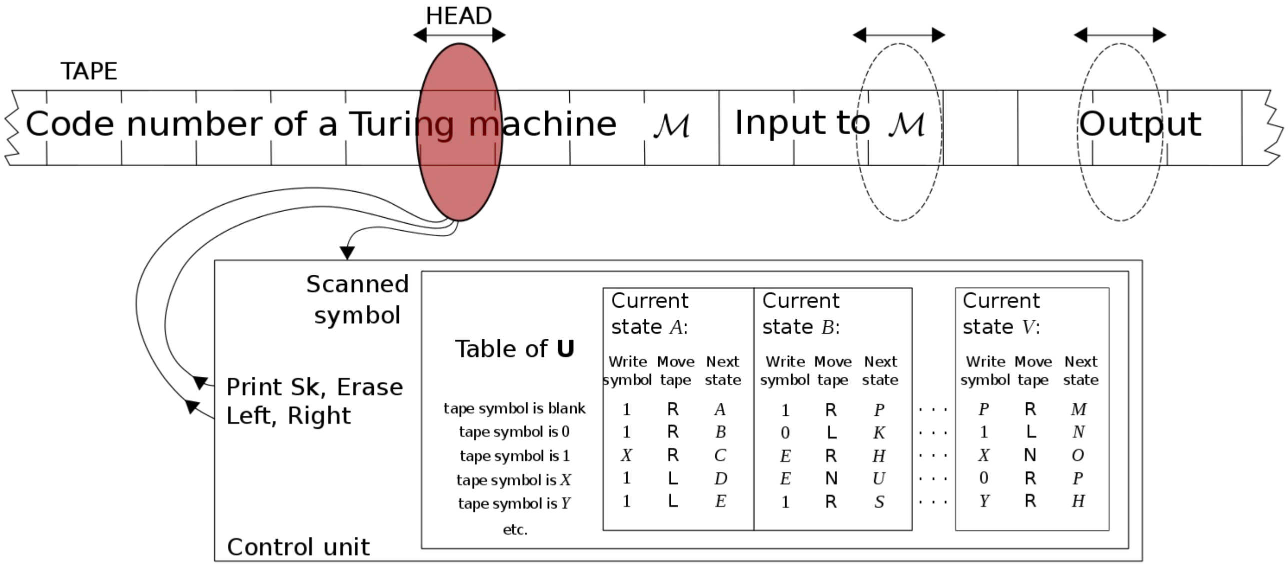
What's a computer look like?

Turing machine



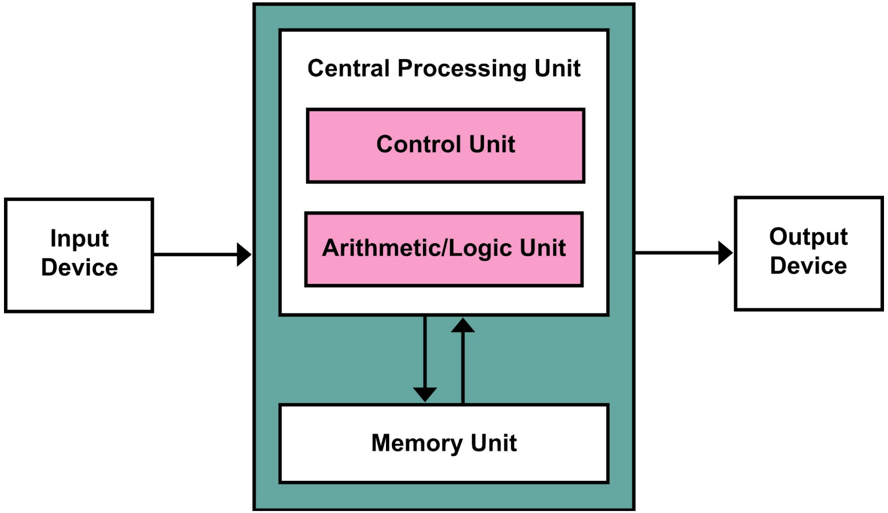
What's a computer look like?

A universal Turing Machine



What's a computer look like?

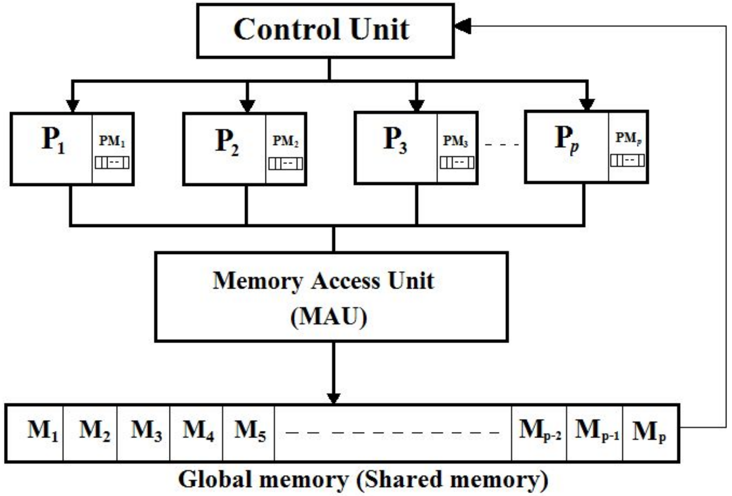
Von Neumann Architecture



Properties of “computers”

- Sequential processing
  - Control or logical flow
- Algorithm costs measured in this model
  - Big-O notation counts number of sequential steps/bits for storage
- This is the basis for the CS curriculum
  - And it’s just wrong
  - Computers are not sequential and performance is more nuanced than counting the number of steps
- We look at computers as parallel entities
  - Do many tasks concurrently
  - Tasks interfere with each other
  - More accurately reflects hardware and bottlenecks
- What about parallel computation models?
  - Exist but not useful, because reality collides with the abstraction

PRAM Model

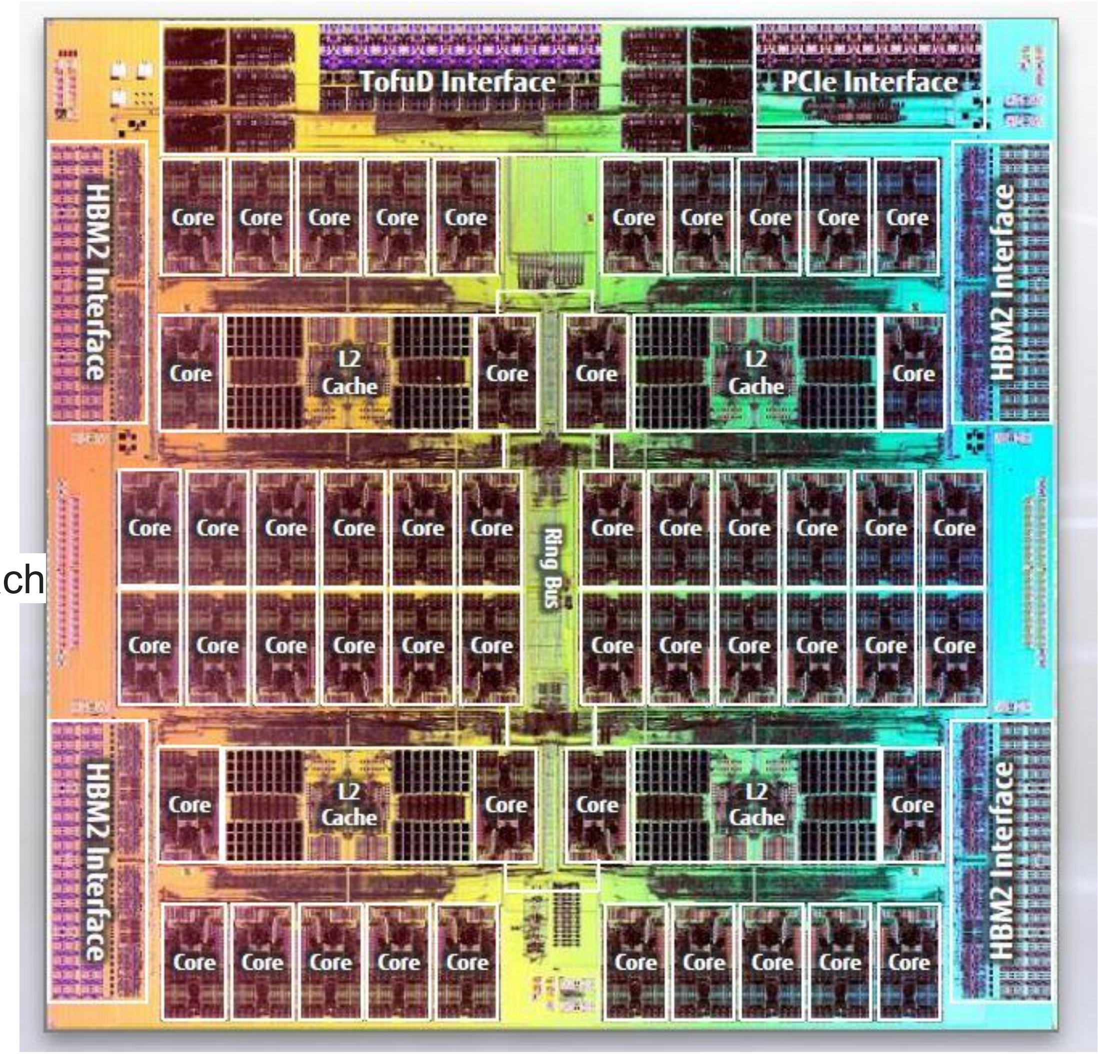


Control Unit
P1
PM1
P2
PM2
P3
PM3
Pp
PMp
Memory Access Unit
(MAU)
M1
M2
M3
M4
M5
Mp-2
Mp-1
Mp
Global memory (Shared memory)

A64FX
CPU
4 NUMA nodes
12 compute cores each

Memory : 32 GB

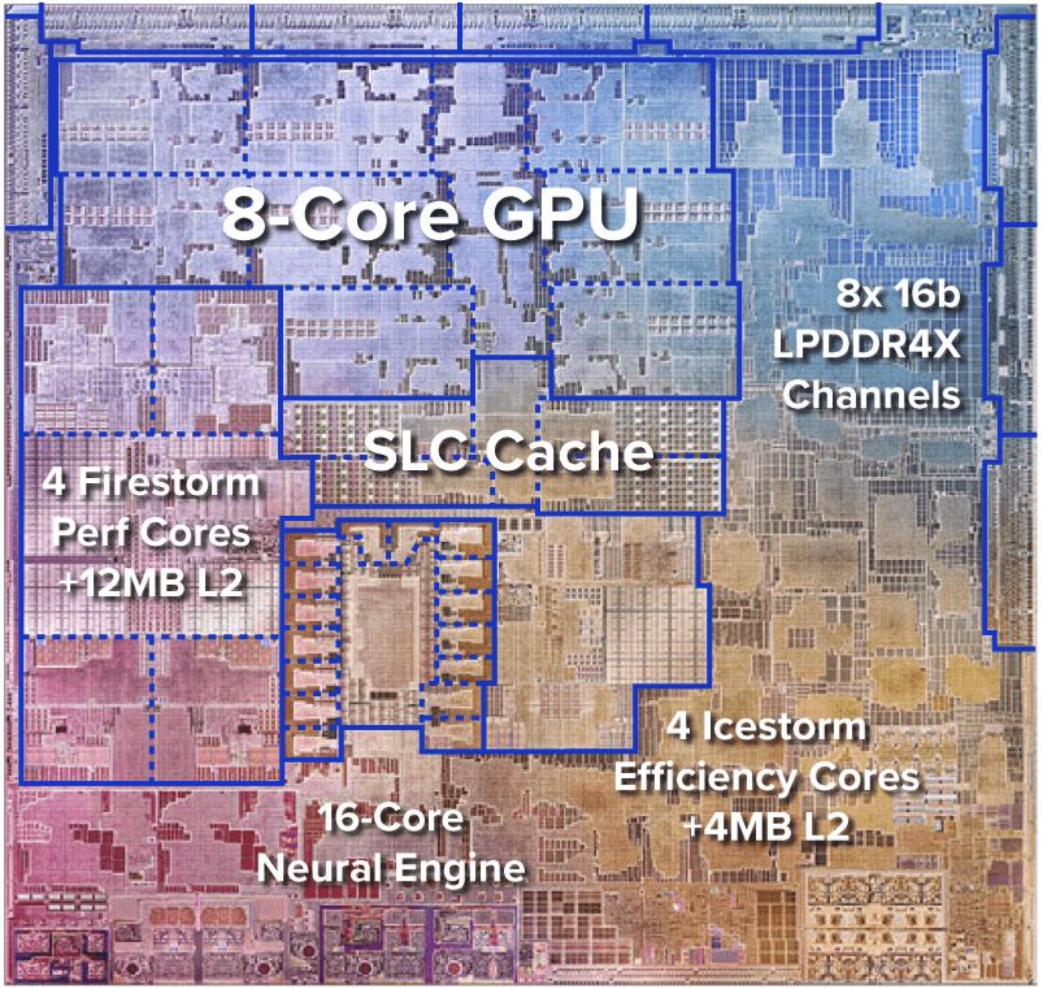
FLOPS : 13.5 10^12



Apple M1
CPU
8 compute cores

GPU
Up to 128 execute unit
Up to 8 threads

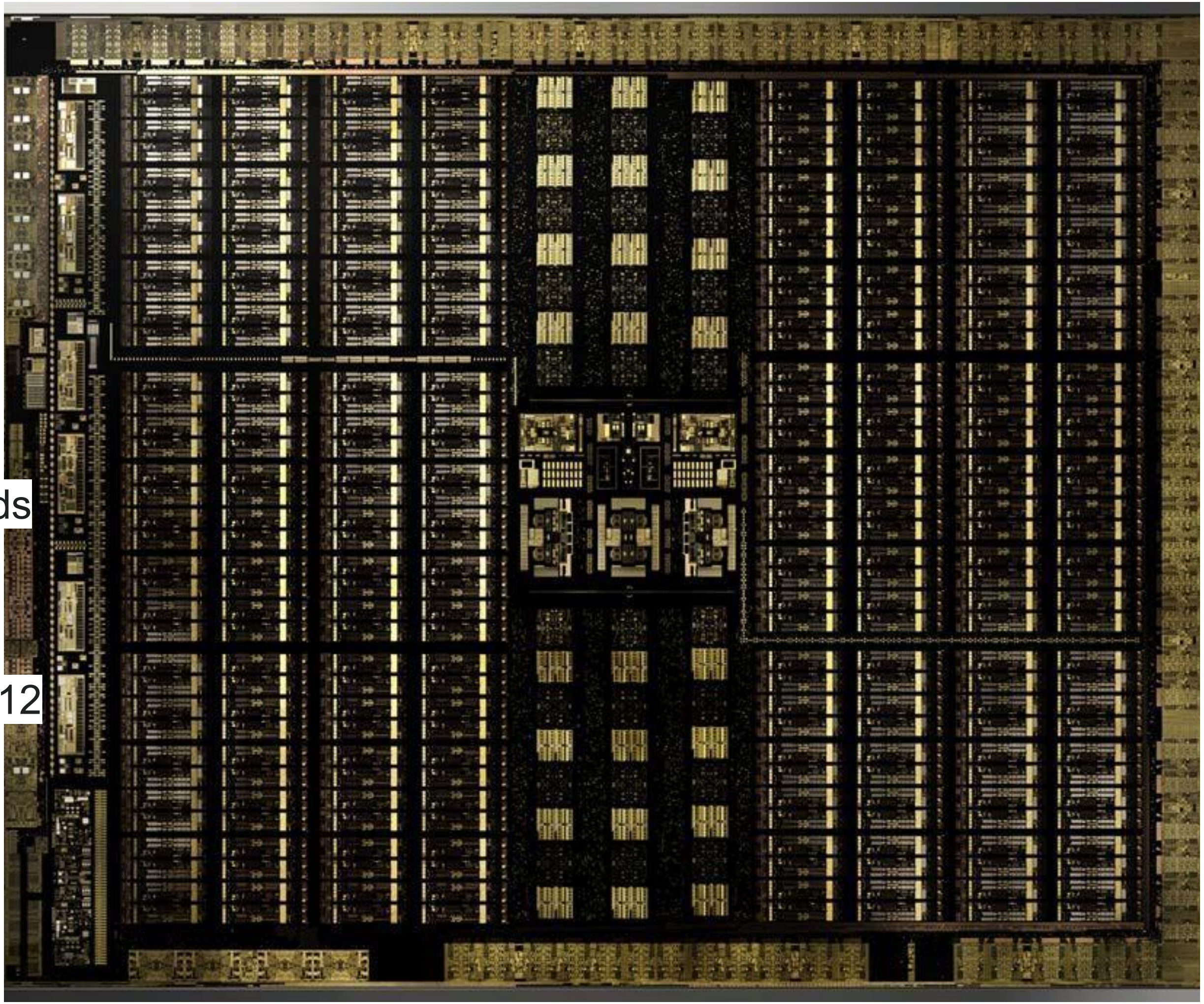
FLOPS : 2.6 10^12



NVIDIA RTX
GPU
Up to 4,608 threads

Memory : 24Gb

FLOPS : 16.3 10^12



Serial computing should be re-invented

- Realities of computing
  - There are tons of wasted cycles
  - CPU utilization typically <10% (of useful work)
- Many other things limit performance
  - Pipeline stalls
  - Lock interference
  - Waiting for I/O and network
  - Data dependencies
- Writing serial programs is broken
  - Parallelism is everywhere
  - Must exploit it to realize time efficiency, power savings

About parallelism

- Why do **I want** to write parallel programs?
  - to solve problems faster (strong scaling)
  - to solve bigger problems (weak scaling and memory)

- Why **I do not want** to write parallel programs?
  - tools are more difficult to use: expect 10x programming effort
  - for many problem performance does not improve

This course will help to decide when to develop a parallel approach and how to write it.

Context

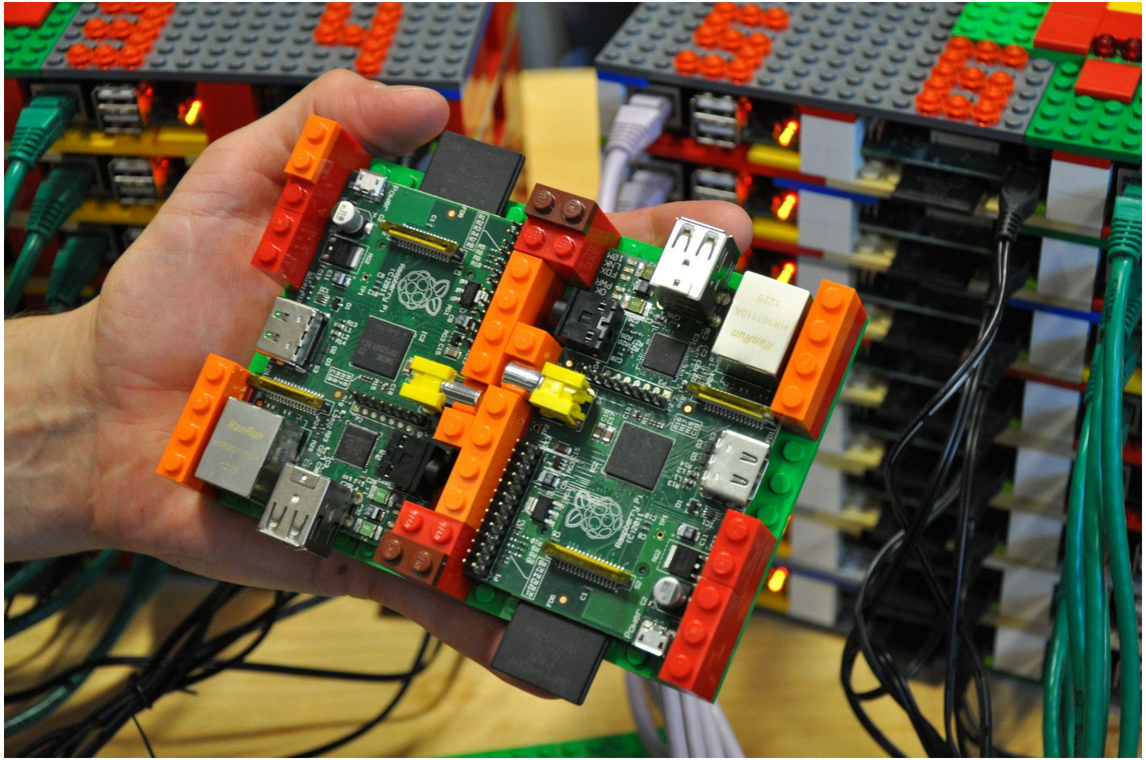
- ➔ Decrease time to solution.
- ➔ Solve larger problem.
- ➔ Combine resources of several processing units: gain access to more memory and more processing power.
- ➔ Harness the processing power of modern architectures.
- ➔ Use idle computer to perform embarrassing parallelism computation (SETI@home).
- ➔ Improve the precision of computations in a limited time (weather forecast).

How to achieve efficient parallelism?

- ➔ Processors: multicore, memory, network, accelerators, instructions.
- ➔ Compilers: dedicated library, automatic parallelism.
- ➔ Algorithms: tailored algorithms.
- ➔ Mathematics: adapted numerical methods, evolutionary methods.

Few words on hardware

A student HPC system



A “real” HPC system

What you will be working with!

- ➔ 33 RTX 6000 are available.
- ➔ Ressources are on virtual machine (up to 3 students on each card)
- ➔ Technical details :
  - ◆ CUDA Threads 4,608
  - ◆ NVIDIA Tensor threads 576
  - ◆ NVIDIA RT 72
  - ◆ GPU 24 GB GDDR6
  - ◆ Performances FP32 - 16,3 TFLOPS


PCI Express 3.0 Host Interface
GigaThread Engine
GPC
Raster Engine
TPC
SM
RT CORE
Memory Controller
L2 Cache
High-Speed Hub
NVLink – Two x8 Links

Conclusions

Every computer is a parallel computer

- ➔ Parallel computers need independent work to run their many cores (or other resources) efficiently

Themes of this class

- ➔ Identify available parallelism in application
- ➔ Design parallel approaches
- ➔ Understand parallel hardware and how to optimize parallel performance