Istituto Nazionale di Fisica Nucleare
Commissione Calcolo e Reti

Corso di formazione per neoassunti nelle attività di Computing

4–7 Mar 2024
LNF

Dal laptop al supercalcolo

Alessandro Costantini, Daniele Cesini - INFN-CNAF
alessandro.costantini <at> cnaf.infn.it

High Performance Computing
HPC

HTC and HPC - definition

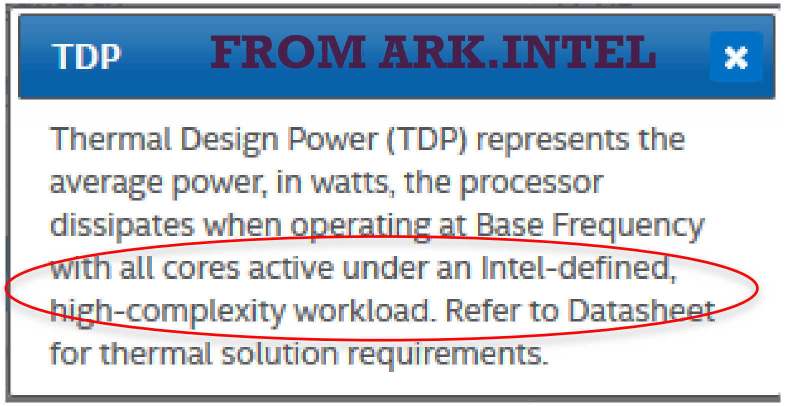
- High Throughput Computing (HTC)
  - The focus is on the execution of many copies of the *same program* at the *same time*
    - not in the speedup of individual jobs
  - Many copies of the same program run *in parallel* or *concurrently*
  - Maximize the **throughput**
- High Performance Computing (HPC)
  - speed up the individual job as much possible so that results are achieved more quickly
- HTC infrastructures tend to deliver large amounts of computational power over a long period of time.
  - In contrast, High Performance Computing (HPC) environments deliver a tremendous amount of compute power over a short period of time.
- The interest in HTC is in how many jobs complete over a long period of time instead of how fast an individual job can complete.

Glossary: GFLOPS and TDP

- GFLOPS: Billions of Floating Point Operations per second
  - Max GFLOPS of a system can be calculated using:

$$\text{GFLOPS} = \text{sockets} \times \frac{\text{cores}}{\text{socket}} \times \text{clock} \times \frac{\text{FLOPs}}{\text{cycle}} \qquad \text{(Clock in GHz)}$$

- TDP: **Thermal Design Power** is the maximum amount of heat generated by the CPU that the cooling system in a computer is required to dissipate in **typical** operation (*)



TDP FROM ARK.INTEL

Thermal Design Power (TDP) represents the average power, in watts, the processor dissipates when operating at Base Frequency with all cores active under an Intel-defined, high-complexity workload. Refer to Datasheet for thermal solution requirements.

(*) From wikipedia

Once upon a time….

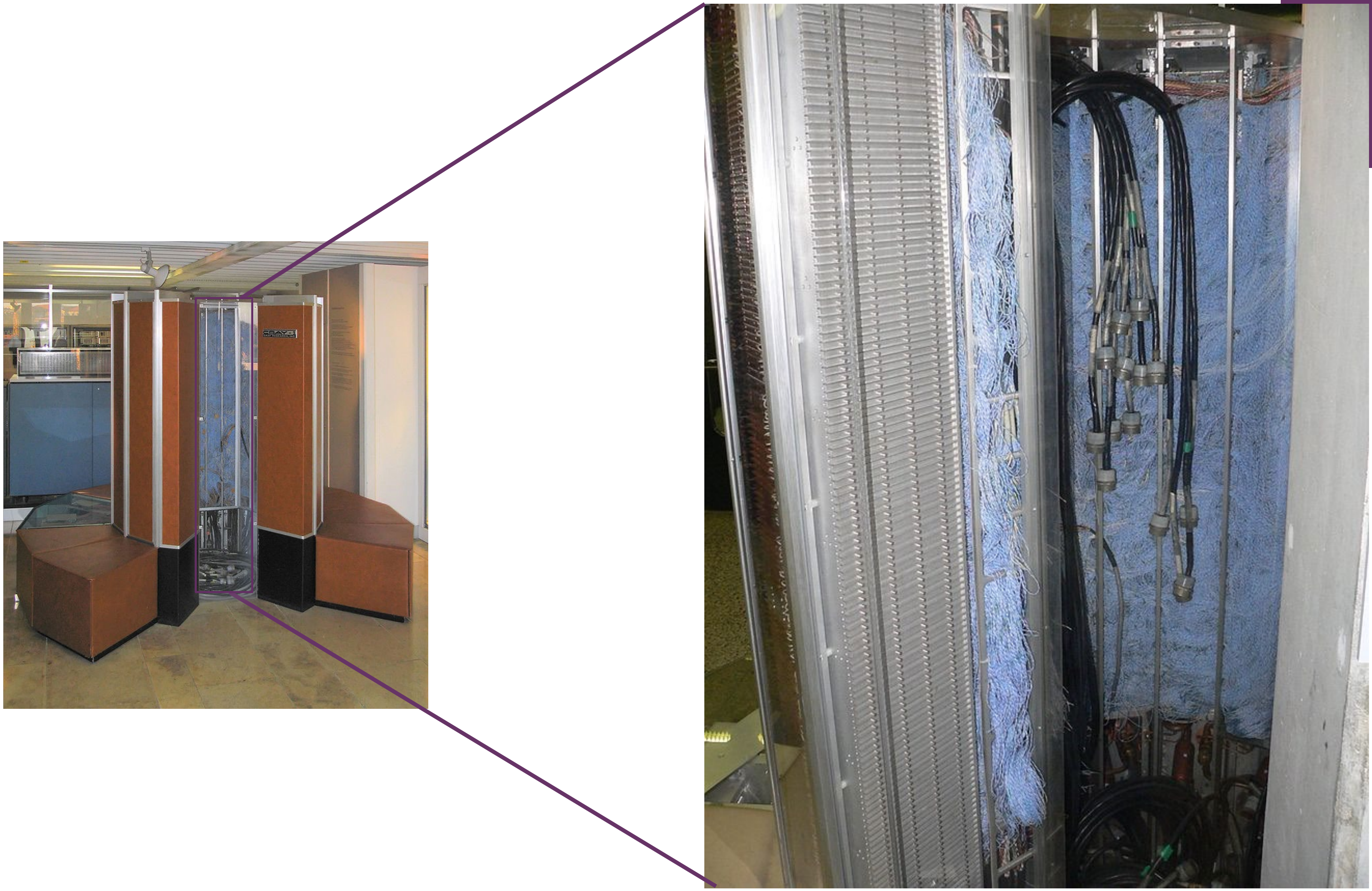
The vector machines

- Serial number 001 Cray-1 ™
  - Los Alamos National Laboratory in 1976
  - $8.8 million
  - 80 MFLOPS scalar, 160/250 MFLOPS vector
  - 1 Mword (64 bit) main memory
  - 8 vector registers
    - 64 elements 64bit each
  - Freon refrigerated
  - 5.5 tons including the Freon refrigeration
  - 115 kW of power
    - 330 kW with refrigeration

(*) Source: Wikipedia

- Serial number 003 was installed at the National Center for Atmospheric Research (NCAR) in 1977 and decommissioned in 1989

Wireless technology inside



+

CRAY-XMP…Vector MultiProcessor

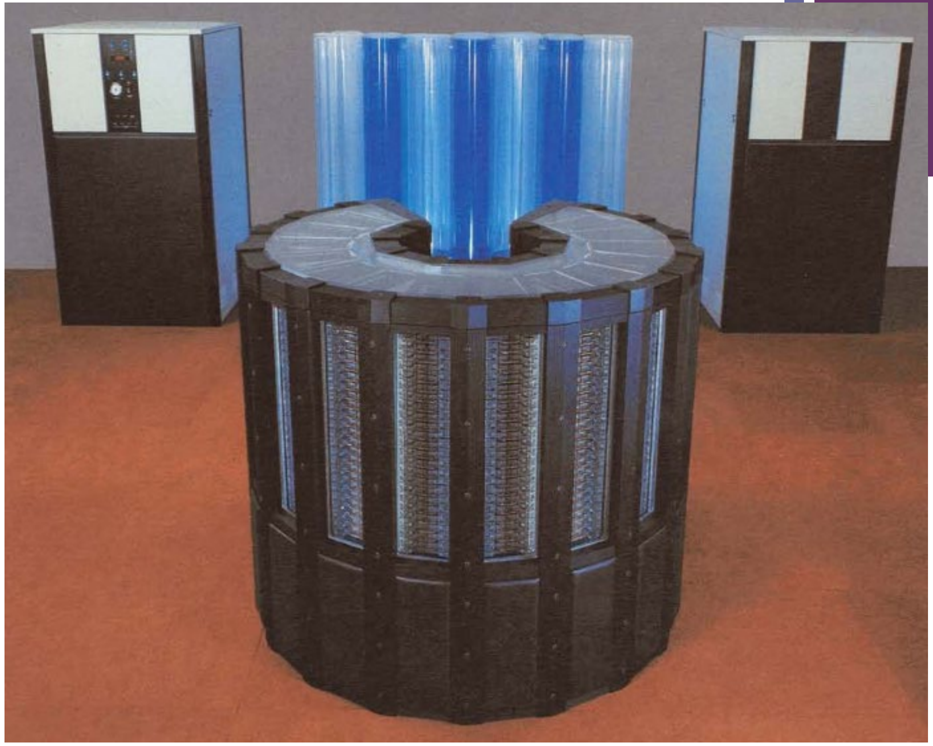
- 1982 CRAY X-MP 2 processors
  - 9.5 ns clock cycle (105 MHz)
  - 2x200MFLOPS
  - 2Mwords (64 bit) = 16MB
- 1984 CRAY X-MP four processors
  - 800 MFLOPS
  - 8Mwords
    - 64 MB main memory
  - about US$15 million
    - plus the cost of disks!!!

CRAY-XMP48 @ CERN in 1984
(*) source:
http://cerncourier.com/cws/article/cern/29150

+

The Cray-2

- The **Cray-2** released in 1985
  - 4 processors
  - 250MHz (4.1 ns)
  - 256 Mword (64bit) Main Memory
    - 2 GByte
  - 1.9 GFLOPS
  - 150 - 200 kW
  - Fluorinert cooling
  - 16 sq ft floor space
  - 5500 pounds
  - About $17 million



An inert fluorocarbon liquid circulates in the mainframe cabinet in direct contact with the integrated circuit packages. This liquid immersion cooling technology allows for the small size of the CRAY-2 mainframe and is thus largely responsible for the high computation rates.

(*) http://archive.computerhistory.org/resources/text/Cray/Cray.Cray2.1985.102646185.pdf

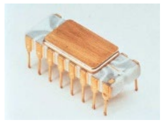
Microprocessors

The Intel 80486
1989

The Intel 80286
1982

The Intel 8086
1978

The Intel 8080
1974

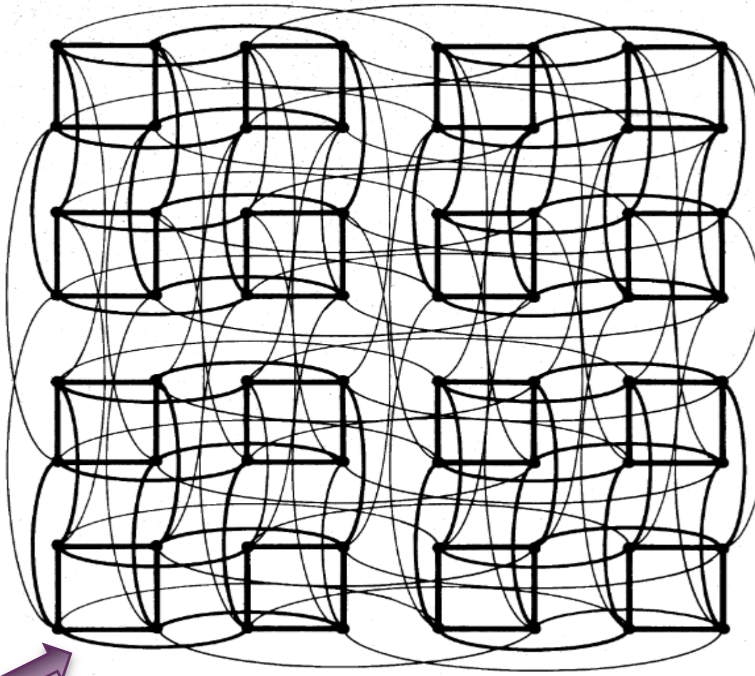


The Intel 4004
1971

In the 80s started the replacement of vector supercomputers with Massively Parallel Processors (MPP) and Clusters assembled from a larger number of lower performing microprocessors

The attack of the Killer Micros

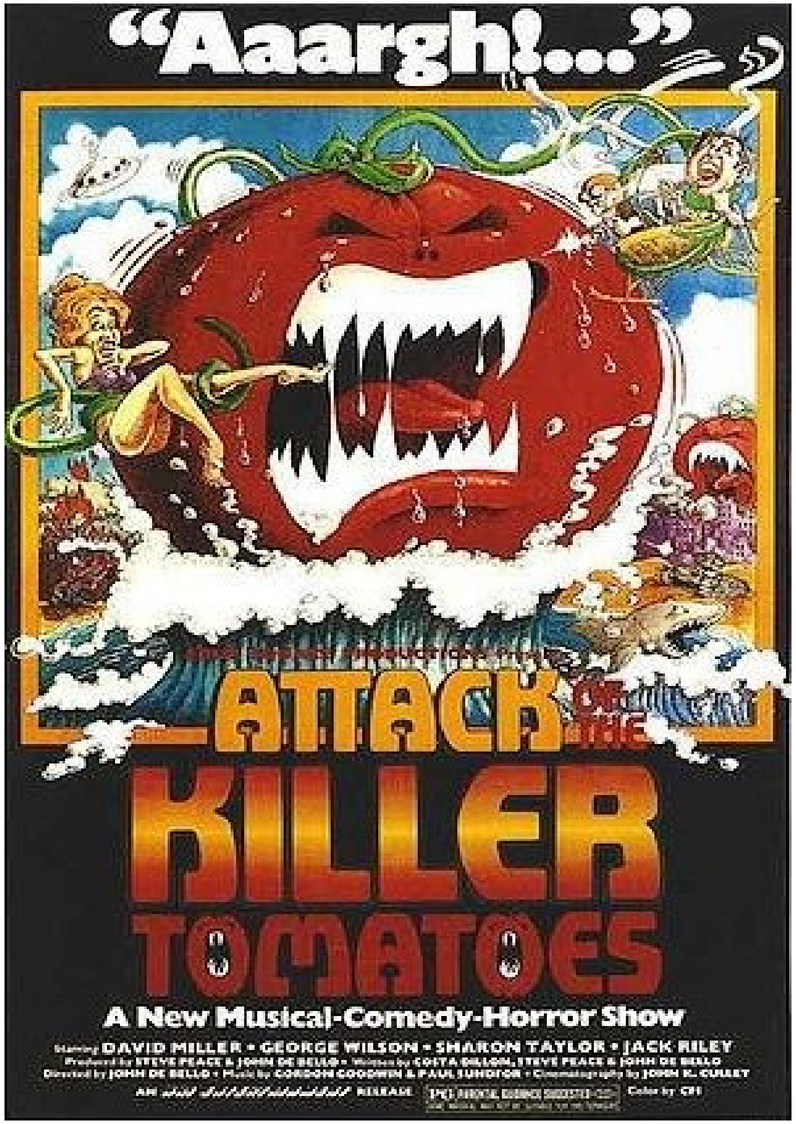
Taken from the title of Eugene Brooks' talk "**Attack of the Killer Micros**" at Supercomputing 1990



- Caltech Cosmic Cube
  - By Charles Seitz and Geoffrey Fox in1981
  - 64 Intel 8086/8087 processors
  - 128 kB per processor
  - 6 dimensions hypercube

(*)http://calteches.library.caltech.edu/3419/1/Cubism.pdf

The Killer Tomatoes


"Aaargh!..."
ATTACK OF THE KILLER TOMATOES
A New Musical-Comedy-Horror Show
DAVID MILLER • GEORGE WILSON • SHARON TAYLOR • JACK RILEY

Attack of the Killer Tomatoes is a 1978 comedy horror film directed, produced, edited, scored and co-written by John DeBello

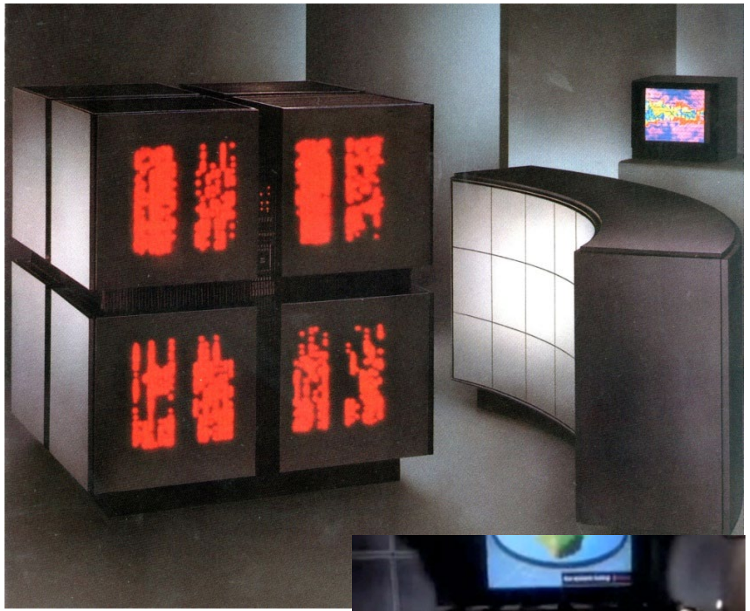
Had three sequels!!!

Massively Parallel Processor (MPP)

- A single computer with many networked processors
  - Specialized interconnect networks
  - Low latency interconnection
- Up to thousands of processors
- Some examples
  - Connection Machines (CM-1/2/200/5)
  - Intel Paragon
  - ASCI series
  - IBM SP
  - IBM BlueGene

Thinking Machines

- 1985: Thinking Machines introduces the Connection Machine CM-1
- Connection Machine CM-200
  - maximum configuration of 65536 1-bit CPUs(!)
  - floating-point unit for every 32 1-bit CPUs
  - A cube composed of 8 cubes
    - each cube contains up to 8096 processors
  - (The curved structure is a Data Vault - a disk array)
  - 40 GFLOPS peak
- 1991: CM-5
  - **Featured in "Jurassic Park"**



(*) Sources:
http://www.corestore.org/cm200.htm
http://www.new-npac.org/projects/cdroms/cewes-1999-06-vol1/nhse/hpccsurvey/orgs/tmc/tmc.html
http://en.wikipedia.org/wiki/Thinking_Machines_Corporation

Intel Paragon MPP

- Launched in 1993
- Up to 2048 (later 4000) Intel i860 RISC microprocessors
  - Connected in a 2D grid
  - Processors @ 50 MHz
- World most powerful supercomputer in 1994
  - Paragon XP/S140
  - 3680 processors
  - 184 GFLOPS peak

(*) Source: Wikipedia

ASCI Red MPP

- 1996 At Sandia Laboratories
- Based on the Paragon architecture
- Fastest supercomputer from 1997 to 2000
  - 1.4 TFLOPS (peak) in 1997
    - 9152 cores
  - 3,2 TFLOPS (peak) in 1999
    - 9632 cores
- 1st supercomputer above 1 TFLOPS

TERA SCALE

(*) Source: Wikipedia

IBM BlueGene/Q MPP

- **Trading the speed of processors for lower power consumption**
- System-on-a-chip design. All node components were embedded on one chip
- A large number of nodes
- 5D xTorus interconnect
- Compute chip is an 18 core chip (2012)
  - The 64-bit PowerPC A2
  - 4-way simultaneously multithreaded per core
  - 1.6 GHz
  - a 17th core for operating system functions
  - chip manufactured on IBM's copper SOI process at 45 nm.
  - 204.8 GFLOPS and 55 watts per processor
- Up to 20 PFLOPS (peak)
  - 16384 cores

PETA SCALE

(*) Source: Wikipedia

+

Clusters

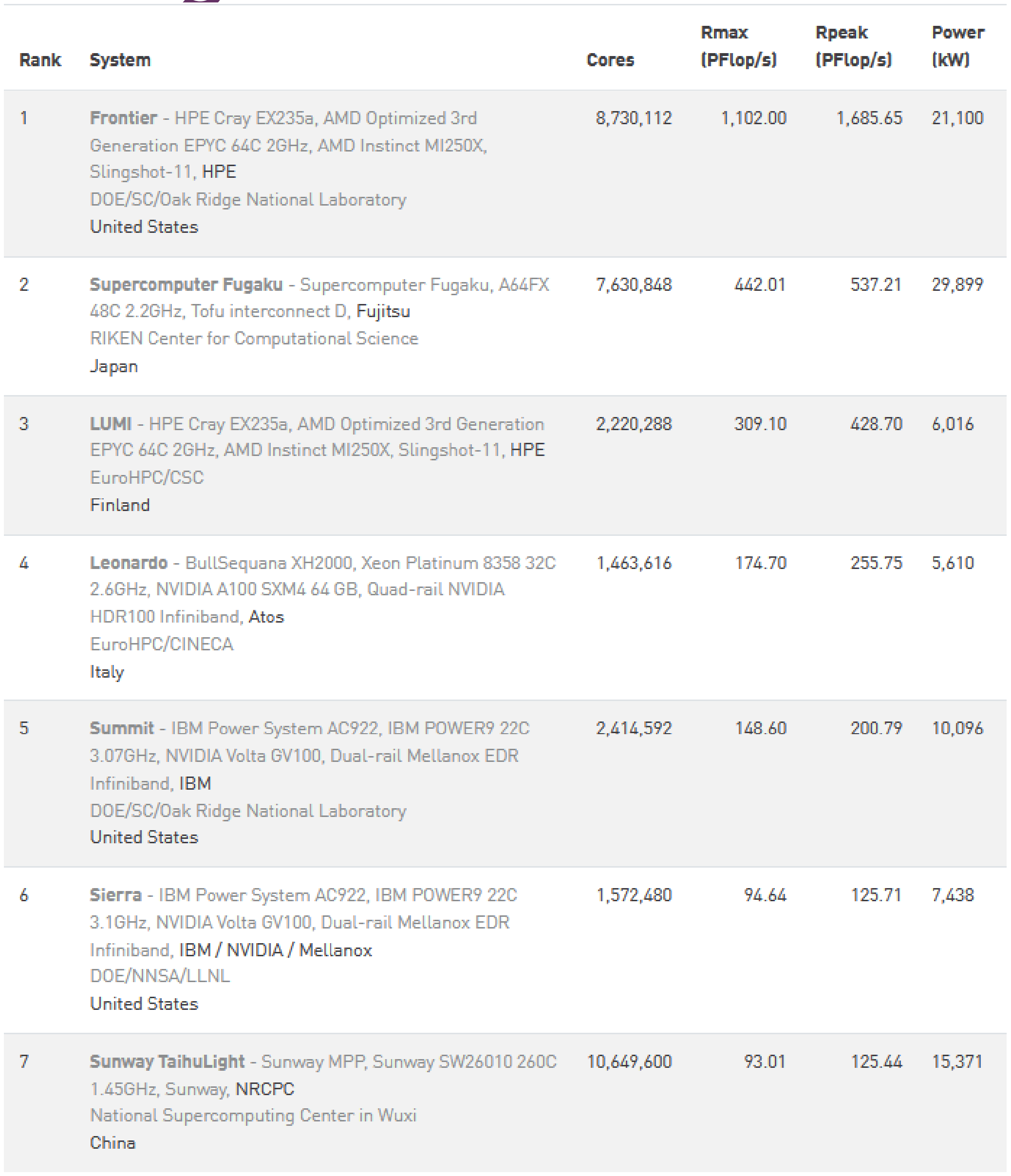
[a cluster is a] parallel computer system comprising an integrated collection of independent nodes, each of which is a system in its own right, capable of independent operation and derived from products developed and marketed for other stand-alone purposes

Dongarra et al. : "High-performance computing: clusters, constellations, MPPs, and future directions", Computing in Science & Engineering (Volume:7 , Issue: 2)

(*) Picture from: http://en.wikipedia.org/wiki/Computer_cluster

Top500.org

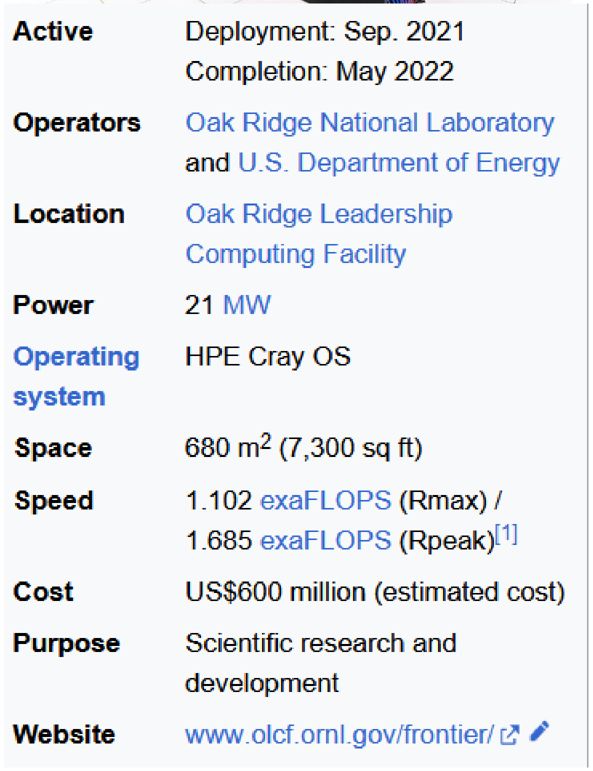
Nov 2022 List



Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (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	8,730,112	1,102.00	1,685.65	21,100
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	1,463,616	174.70	255.75	5,610
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
6	**Sierra** - IBM Power System AC922, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM / NVIDIA / Mellanox DOE/NNSA/LLNL United States	1,572,480	94.64	125.71	7,438
7	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway, NRCPC National Supercomputing Center in Wuxi China	10,649,600	93.01	125.44	15,371

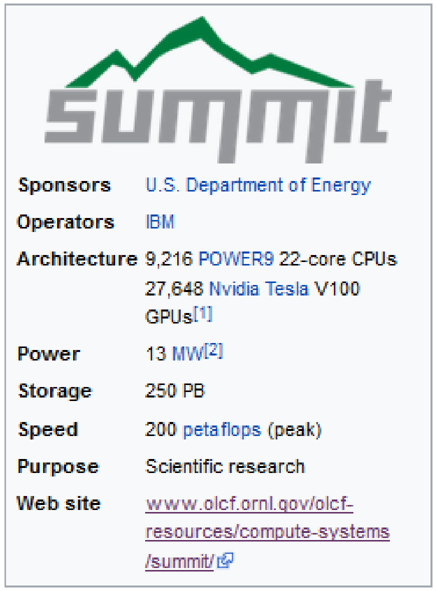
+
Frontier

- Frontier, or OLCF-5, is the world's first and fastest exascale supercomputer, hosted at the Oak Ridge Leadership Computing Facility (OLCF) in Tennessee, United States
- It is based on the Cray EX and is the successor to Summit (OLCF-4).
- As of March 2023, Frontier is the world's fastest supercomputer.
- Frontier achieved an Rmax of 1.102 exaFLOPS
- Frontier uses 9,472 AMD Epyc 7453s "Trento" 64 core 2 GHz CPUs (606,208 cores) and 37,888 Radeon Instinct MI250X GPUs (8,335,360 cores).



Active	Deployment: Sep. 2021 Completion: May 2022
Operators	Oak Ridge National Laboratory and U.S. Department of Energy
Location	Oak Ridge Leadership Computing Facility
Power	21 MW
Operating system	HPE Cray OS
Space	680 m^2 (7,300 sq ft)
Speed	1.102 exaFLOPS (Rmax) / 1.685 exaFLOPS (Rpeak)[1]
Cost	US$600 million (estimated cost)
Purpose	Scientific research and development
Website	www.olcf.ornl.gov/frontier/

Summit OLCF-4 supercomputer



Sponsors	U.S. Department of Energy
Operators	IBM
Architecture	9,216 POWER9 22-core CPUs 27,648 Nvidia Tesla V100 GPUs[1]
Power	13 MW[2]
Storage	250 PB
Speed	200 petaflops (peak)
Purpose	Scientific research
Web site	www.olcf.ornl.gov/olcf-resources/compute-systems/summit/

600 GB of coherent memory addressable by all CPUs and GPUs

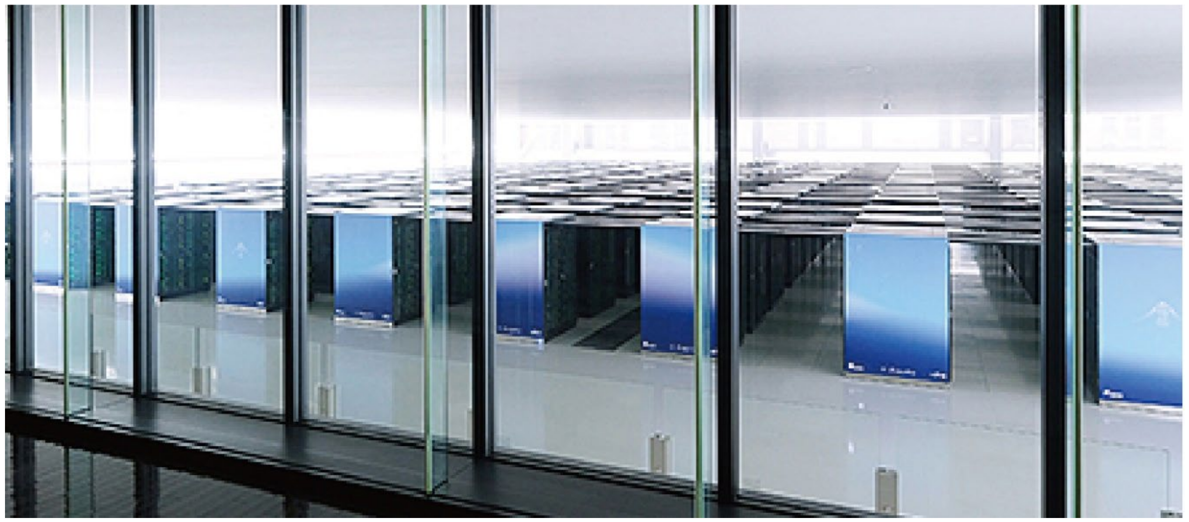
800 GB of non-volatile RAM that can be used as a burst buffer or as extended memory.

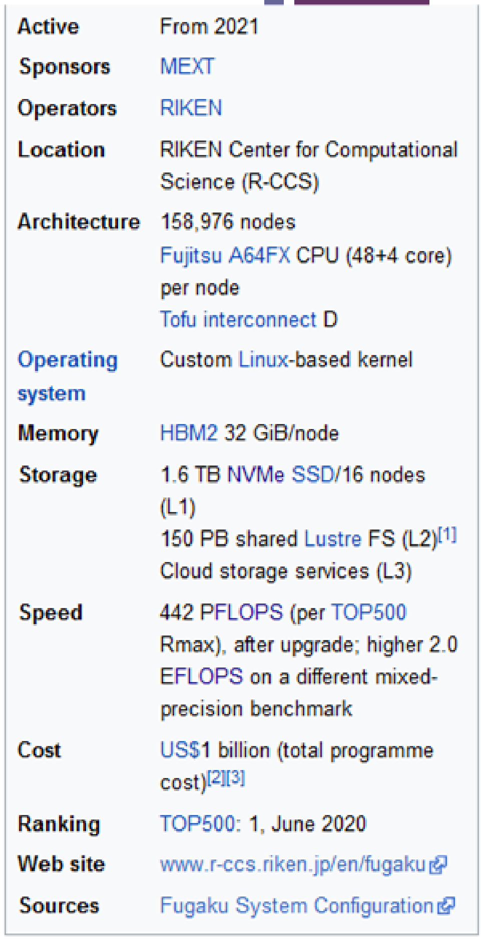
Non-blocking fat-tree topology using a dual-rail Mellanox EDR InfiniBand interconnect for both storage and inter-process communications traffic

United States Department of Energy awarded a $325 million contract in November, 2014 to IBM, Nvidia and Mellanox for the construction of Summit and Sierra

- Summit is tasked with civilian scientific research and is located at the Oak Ridge National Laboratory in Tennessee.
- Sierra is designed for nuclear weapons simulations and is located at the Lawrence Livermore National Laboratory in California.
- Summit is estimated to cover the space of two basketball courts and require 136 miles of cabling

Fugaku Supercomputer





Active	From 2021
Sponsors	MEXT
Operators	RIKEN
Location	RIKEN Center for Computational Science (R-CCS)
Architecture	158,976 nodes Fujitsu A64FX CPU (48+4 core) per node Tofu interconnect D
Operating system	Custom Linux-based kernel
Memory	HBM2 32 GiB/node
Storage	1.6 TB NVMe SSD/16 nodes (L1) 150 PB shared Lustre FS (L2)[1] Cloud storage services (L3)
Speed	442 PFLOPS (per TOP500 Rmax), after upgrade; higher 2.0 EFLOPS on a different mixed-precision benchmark
Cost	US$1 billion (total programme cost)[2][3]
Ranking	TOP500: 1, June 2020
Web site	www.r-ccs.riken.jp/en/fugaku
Sources	Fugaku System Configuration

- The supercomputer is built with the Fujitsu A64FX microprocessor.
  - Based on the ARM version 8.2A processor architecture
  - Fugaku was aimed to be about 100 times more powerful than the K computer
    - i.e. a performance target of 1 exaFLOPS

- The initial (June 2020) configuration of Fugaku used 158,976 A64FX CPUs joined together using Fujitsu's proprietary torus fusion interconnect.

- An upgrade in November 2020 increased the number of processors
  - **To reach 442 petaFLOPS**

- **1 Billion $ total cost**

Sunway TaihuLight

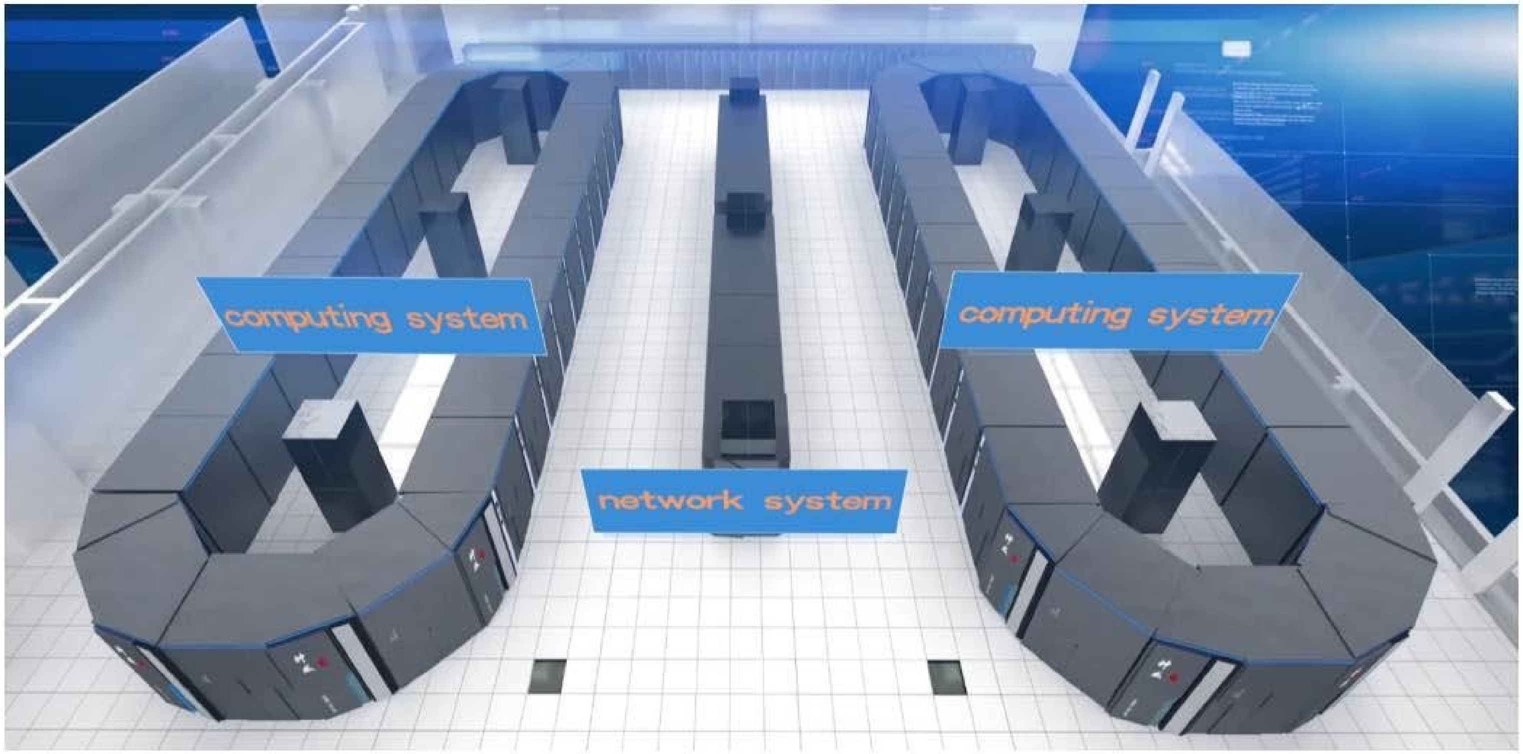
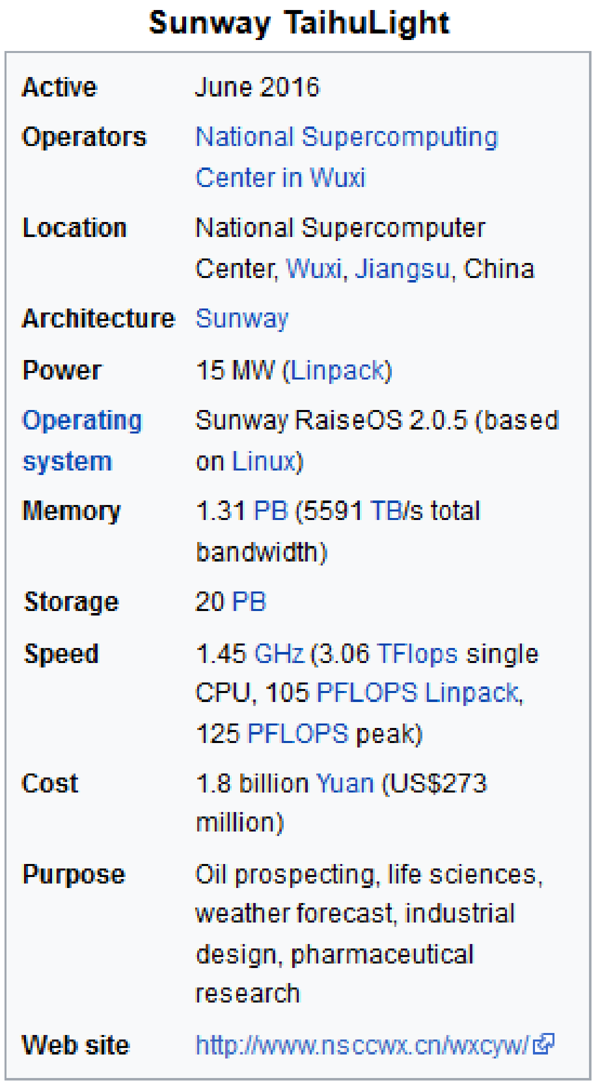


Figure 4: Overview of the Sunway TaihuLight System

+

Sunway TaihuLight



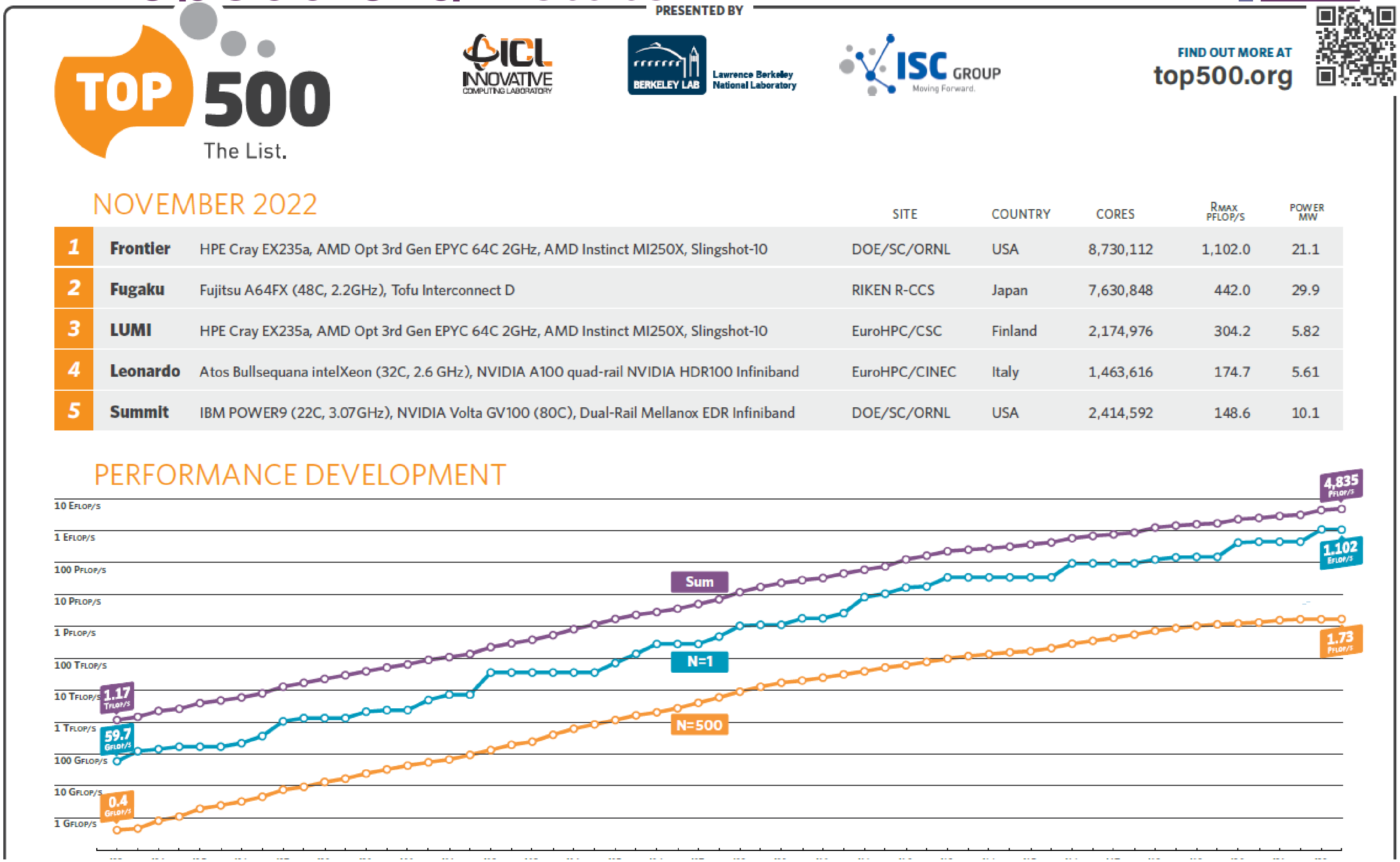
Sunway TaihuLight

Active	June 2016
Operators	National Supercomputing Center in Wuxi
Location	National Supercomputer Center, Wuxi, Jiangsu, China
Architecture	Sunway
Power	15 MW (Linpack)
Operating system	Sunway RaiseOS 2.0.5 (based on Linux)
Memory	1.31 PB (5591 TB/s total bandwidth)
Storage	20 PB
Speed	1.45 GHz (3.06 TFlops single CPU, 105 PFLOPS Linpack, 125 PFLOPS peak)
Cost	1.8 billion Yuan (US$273 million)
Purpose	Oil prospecting, life sciences, weather forecast, industrial design, pharmaceutical research
Web site	http://www.nsccwx.cn/wxcyw/

Sunway TaihuLight

- Architecture
- The Sunway TaihuLight uses a total of 40,960 Chinese-designed SW26010 manycore 64-bit RISC processors based on the Sunway architecture.
- Each processor chip contains 256 processing cores, and an additional four auxiliary cores for system management (also RISC cores, just more fully featured) for a total of 10,649,600 CPU cores across the entire system.
- The processing cores feature 64 KB of scratchpad memory for data (and 16 KB for instructions) and communicate via a network on a chip, instead of having a traditional cache hierarchy.
- http://www.netlib.org/utk/people/JackDongarra/PAPERS/sunway-report-2016.pdf

Top500.org - stats



PRESENTED BY

TOP 500 The List.

ICL INNOVATIVE COMPUTING LABORATORY

BERKELEY LAB Lawrence Berkeley National Laboratory

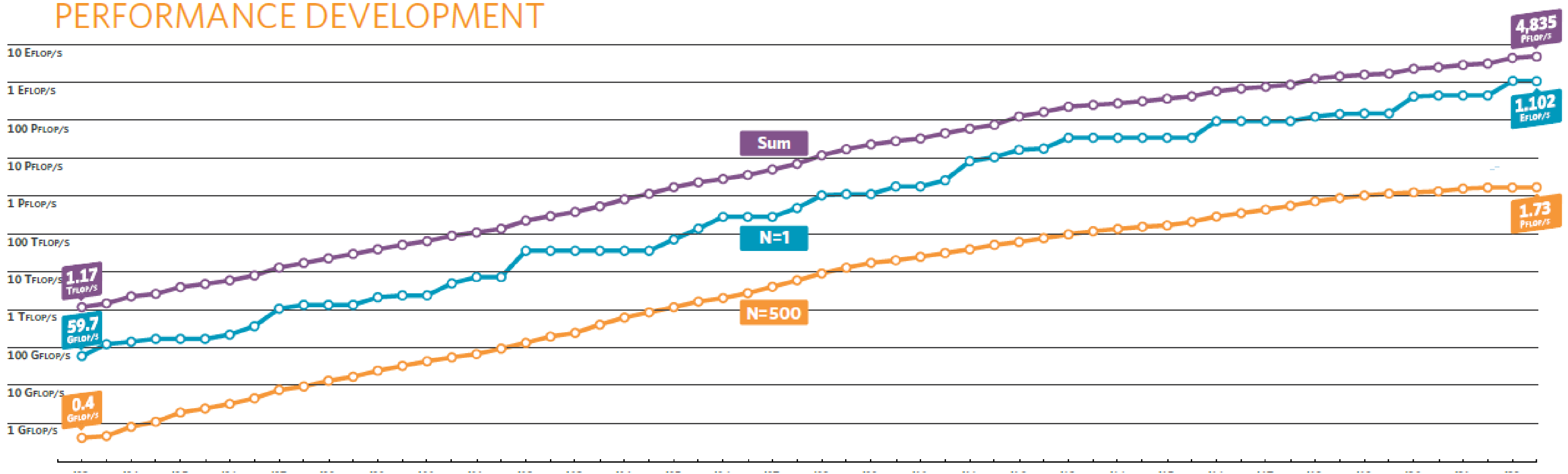
ISC GROUP Moving Forward.

FIND OUT MORE AT top500.org

NOVEMBER 2022

			SITE	COUNTRY	CORES	RMAX PFLOP/S	POWER MW
1	**Frontier**	HPE Cray EX235a, AMD Opt 3rd Gen EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-10	DOE/SC/ORNL	USA	8,730,112	1,102.0	21.1
2	**Fugaku**	Fujitsu A64FX (48C, 2.2GHz), Tofu Interconnect D	RIKEN R-CCS	Japan	7,630,848	442.0	29.9
3	**LUMI**	HPE Cray EX235a, AMD Opt 3rd Gen EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-10	EuroHPC/CSC	Finland	2,174,976	304.2	5.82
4	**Leonardo**	Atos Bullsequana intelXeon (32C, 2.6 GHz), NVIDIA A100 quad-rail NVIDIA HDR100 Infiniband	EuroHPC/CINEC	Italy	1,463,616	174.7	5.61
5	**Summit**	IBM POWER9 (22C, 3.07GHz), NVIDIA Volta GV100 (80C), Dual-Rail Mellanox EDR Infiniband	DOE/SC/ORNL	USA	2,414,592	148.6	10.1

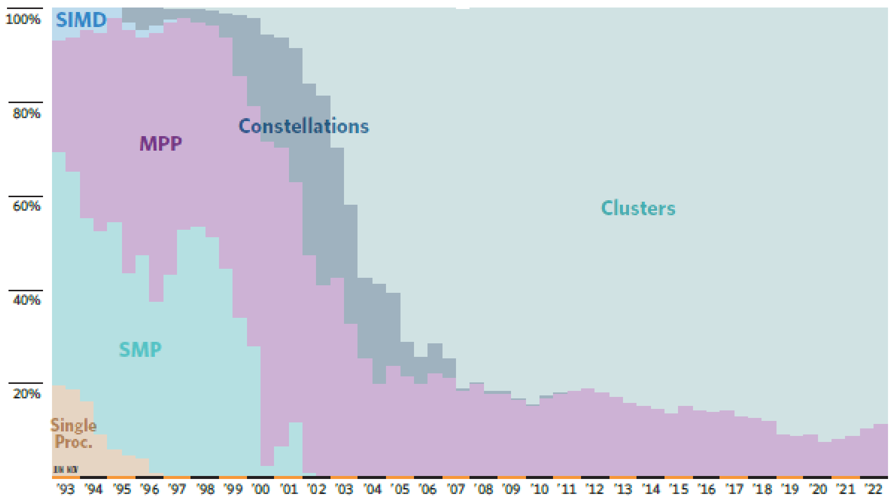
PERFORMANCE DEVELOPMENT



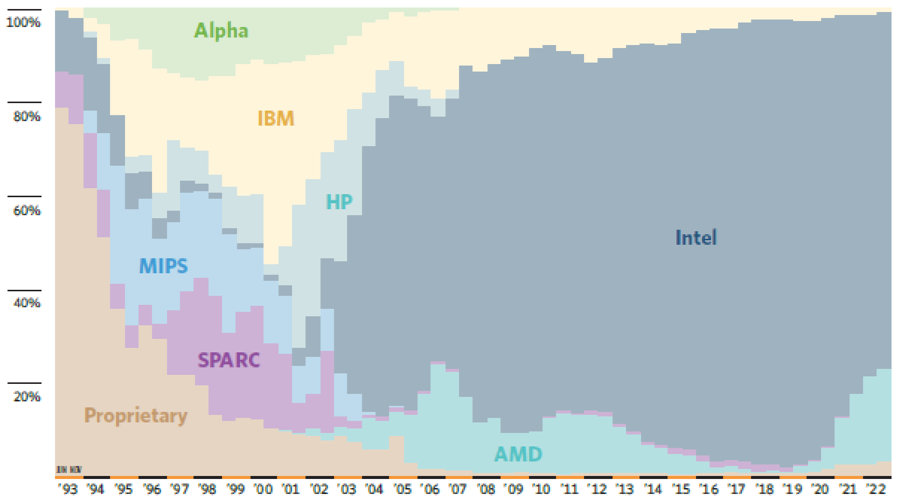
+

Top500.org – stats

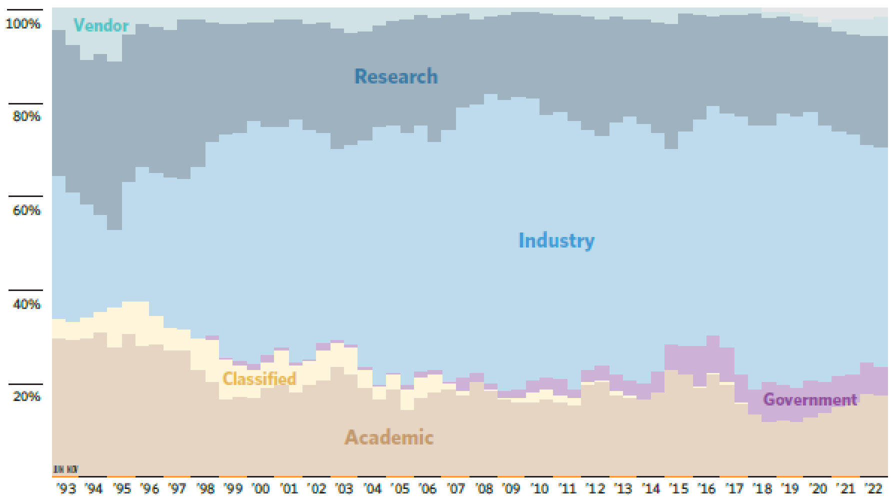
ARCHITECTURES



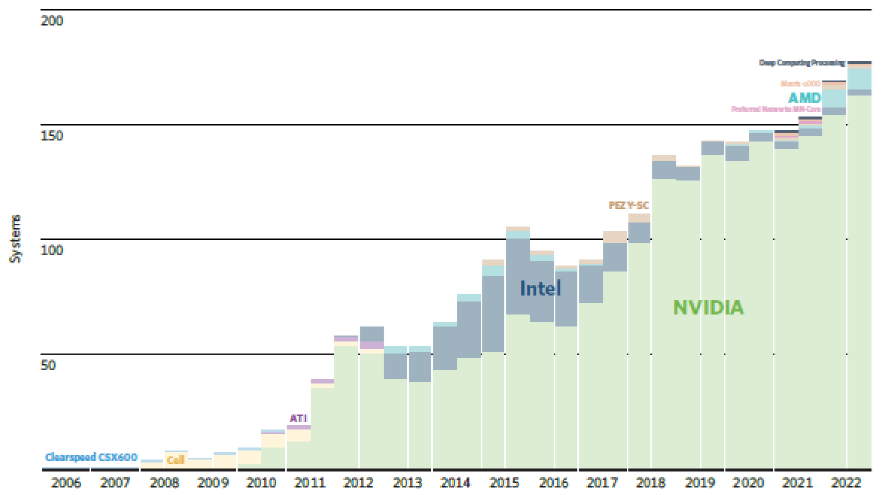
CHIP TECHNOLOGY



INSTALLATION TYPE

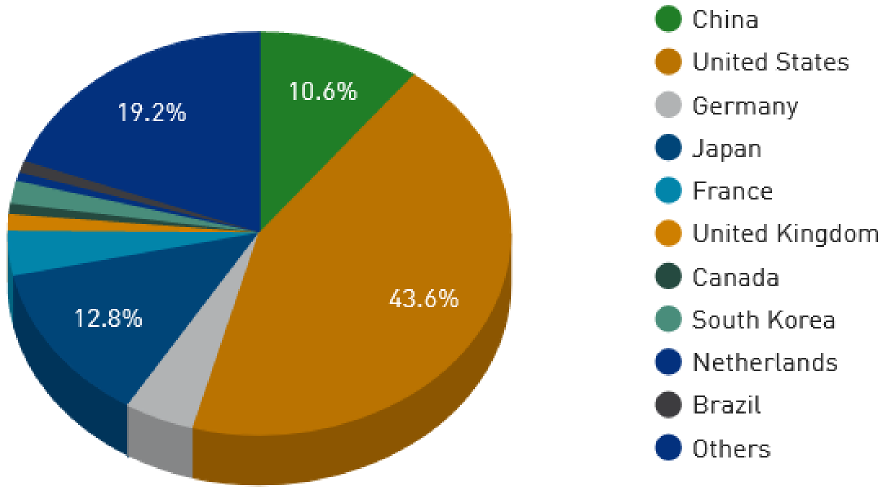


ACCELERATORS/CO-PROCESSORS

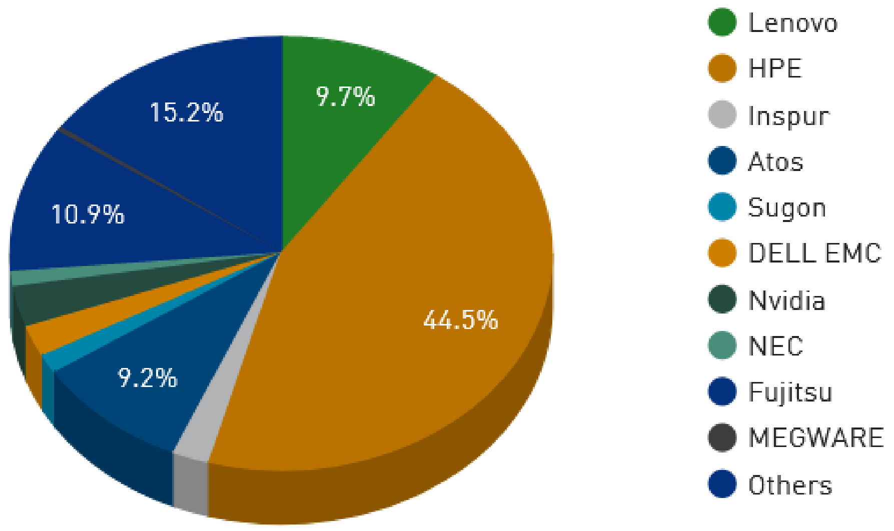


+ Top500.org

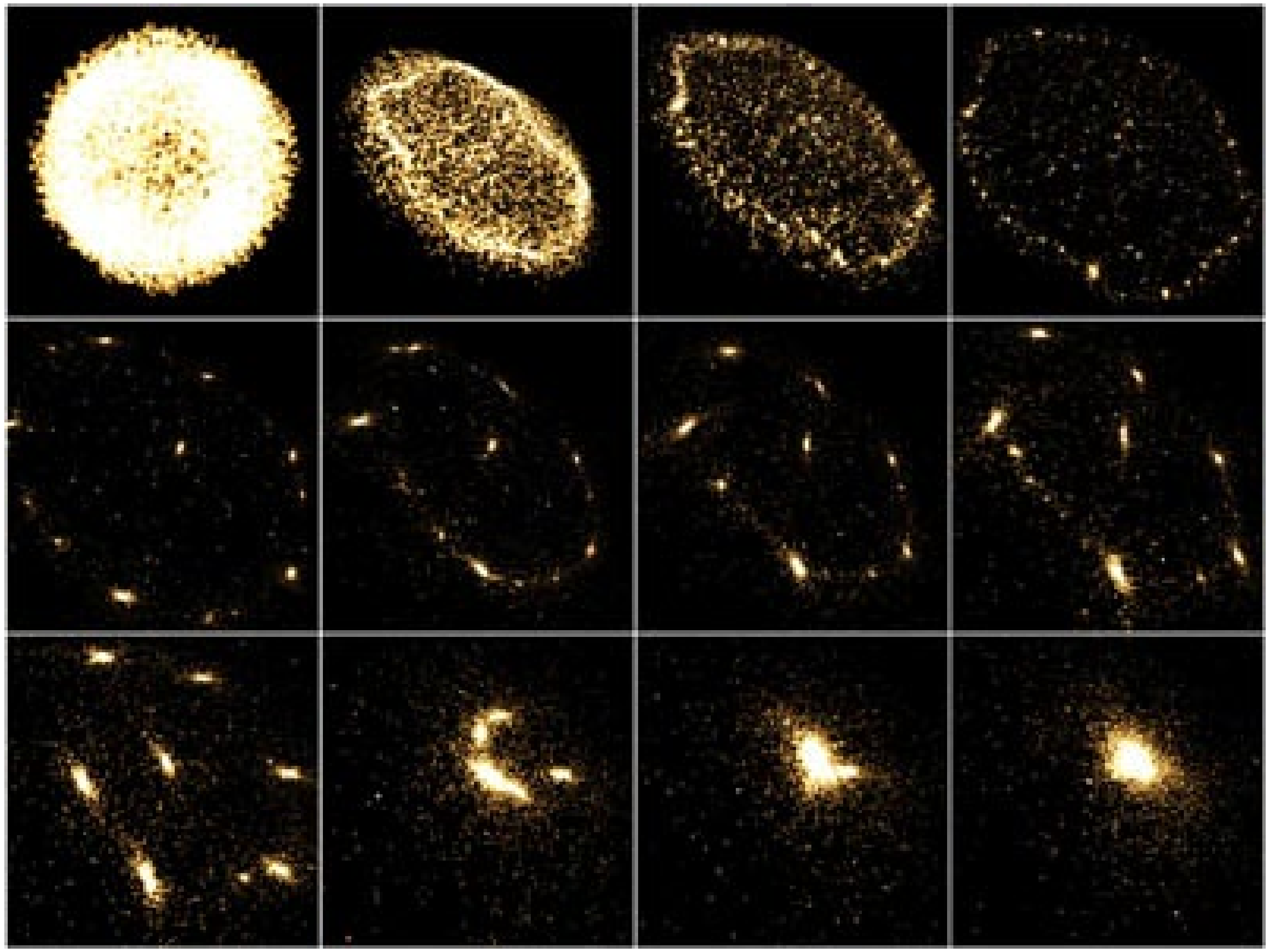
Countries Performance Share



Vendors Performance Share

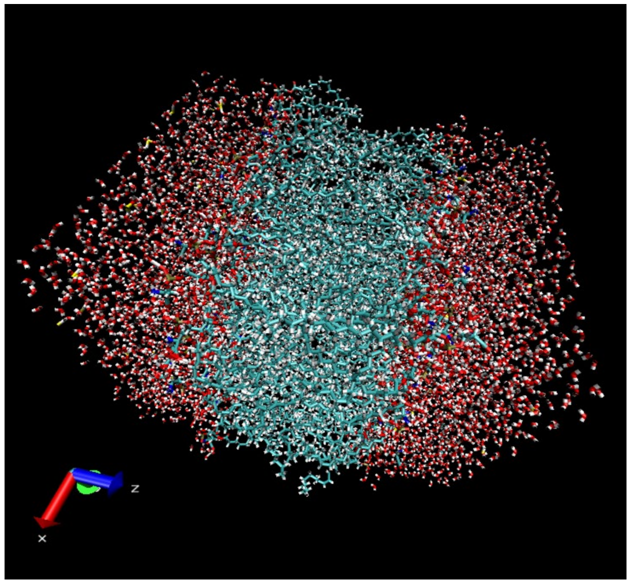


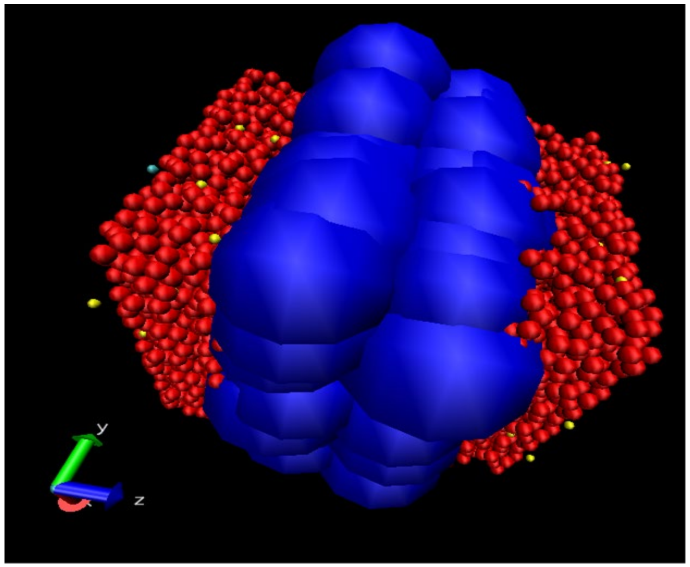
Applications



HPC - Applications

- Molecular Dynamics

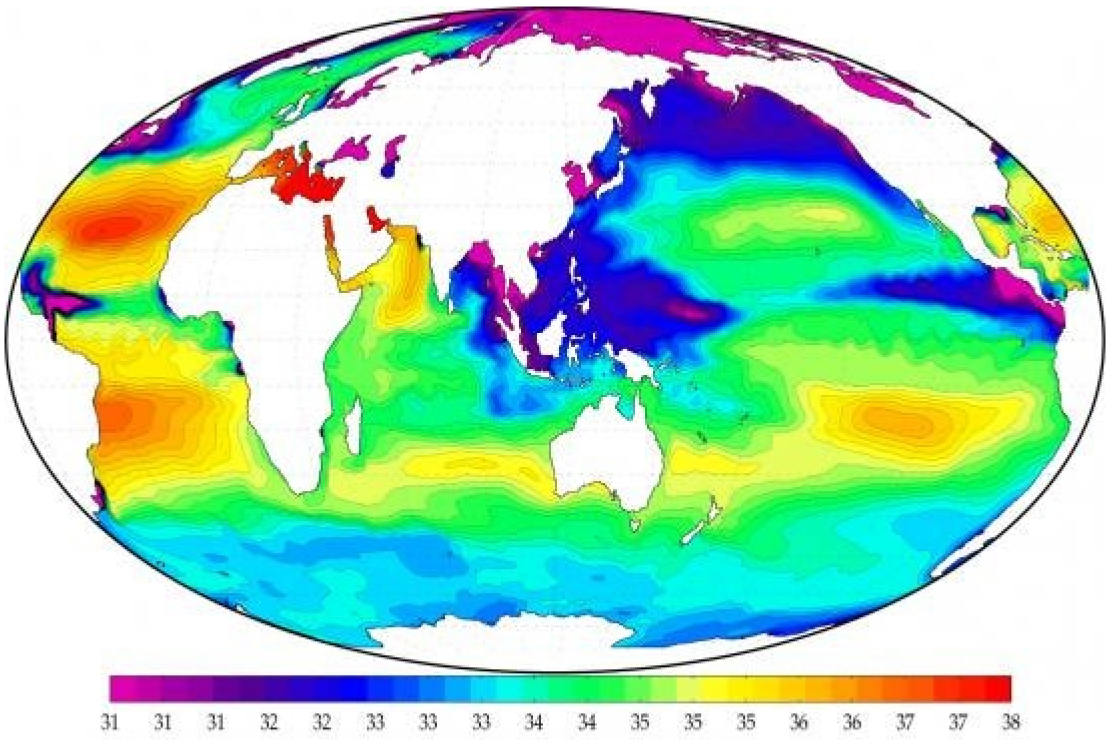


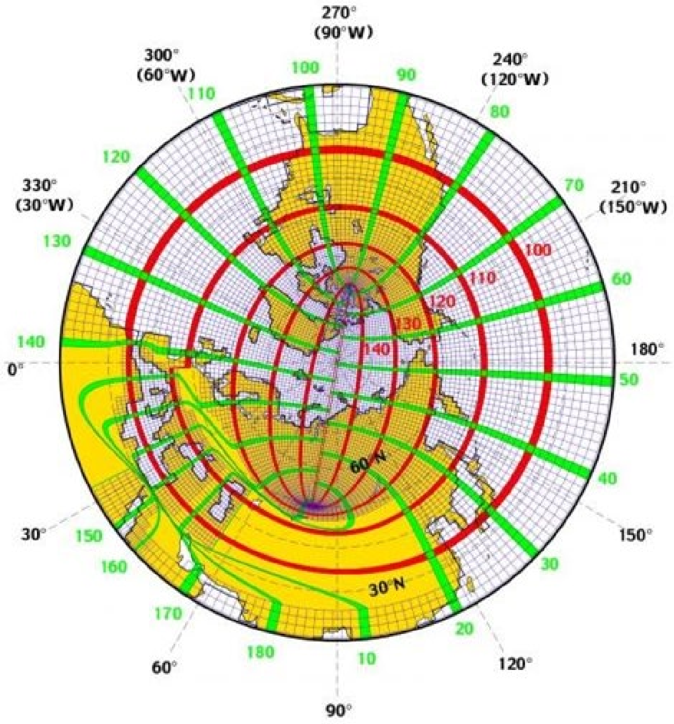


NAMD, Quantum Espresso, Gromacs, Gaussian, etc..

HPC - Applications

- Earth simulation

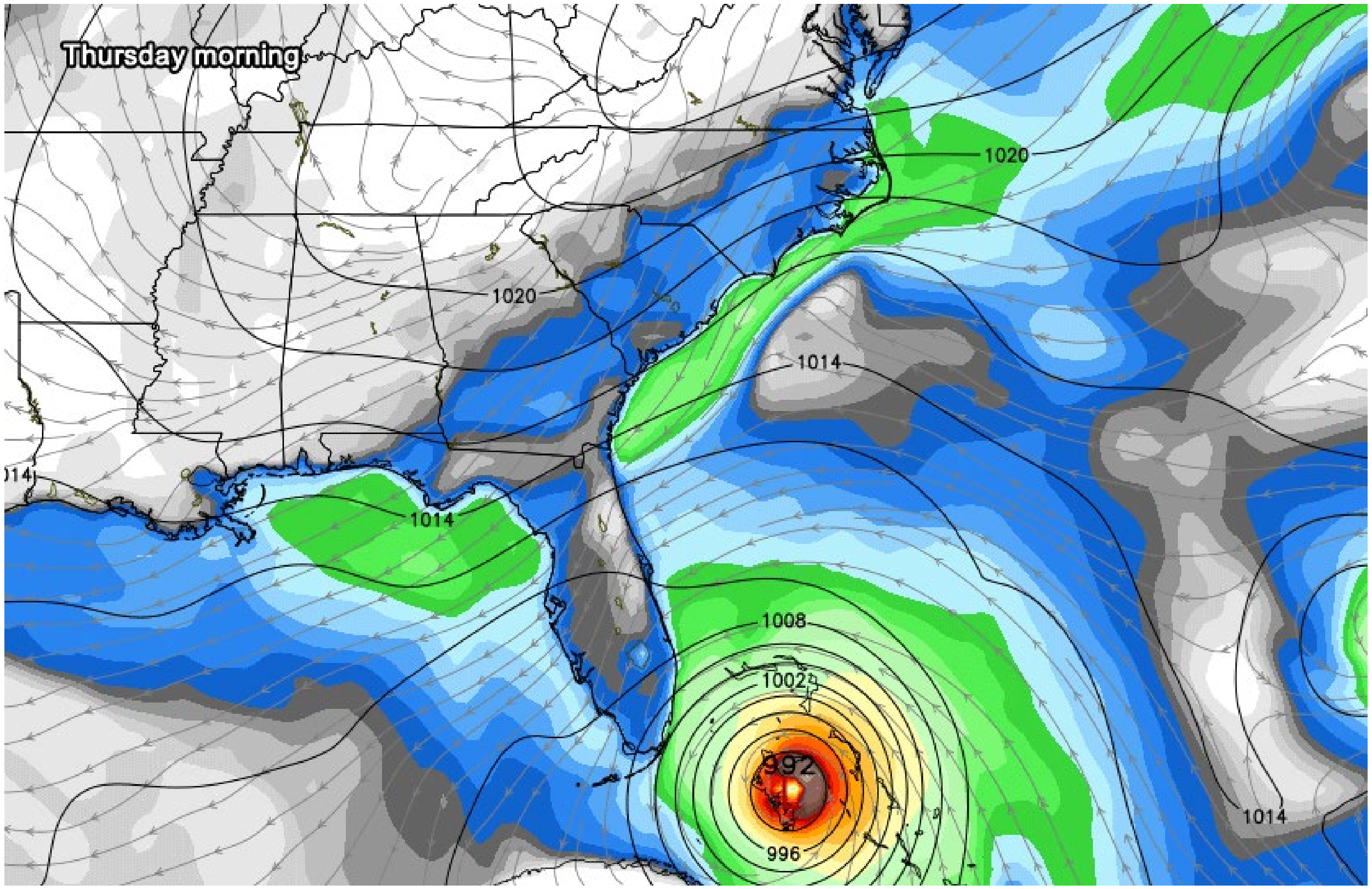




WRF, MM5, GLOBO, NEMO, etc..

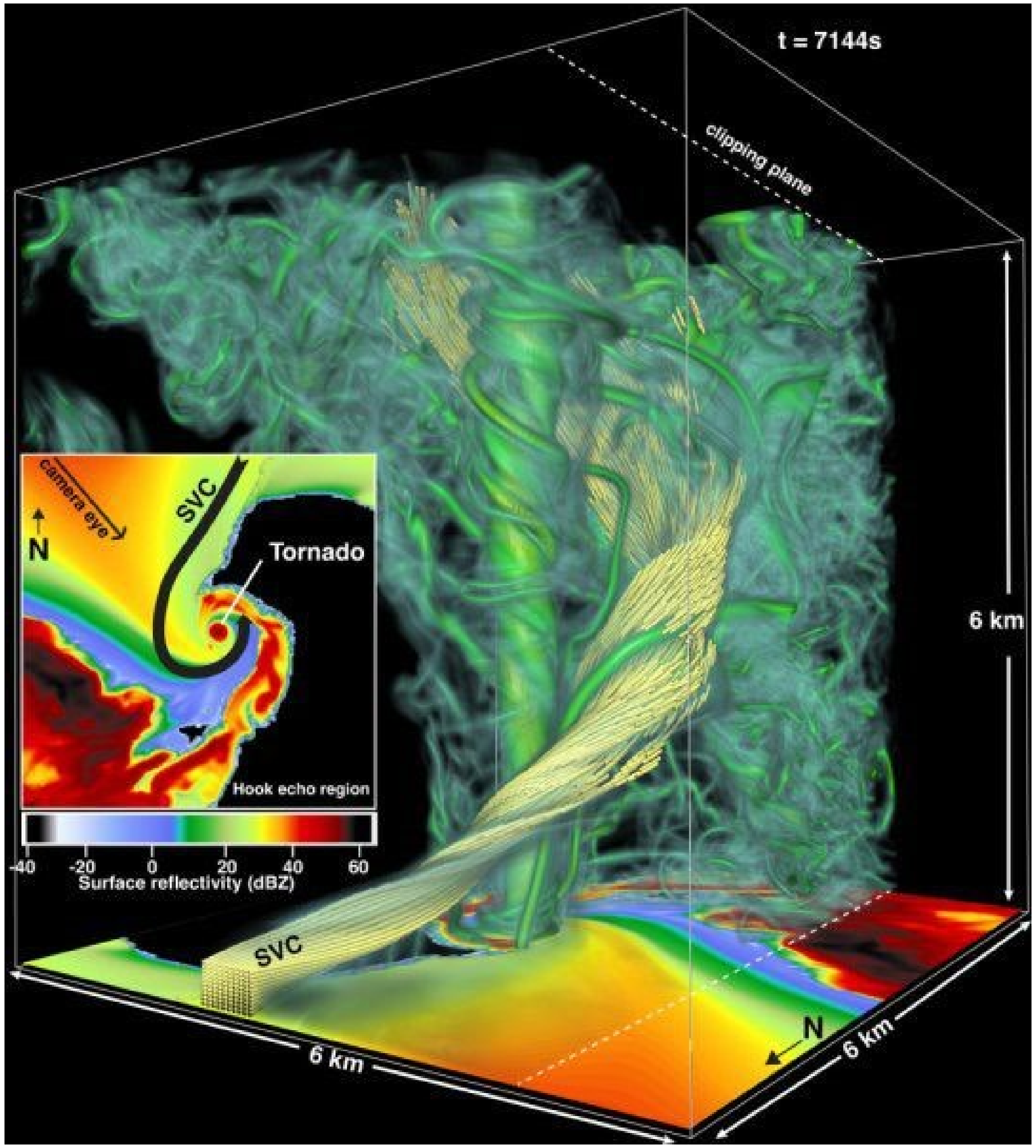
+

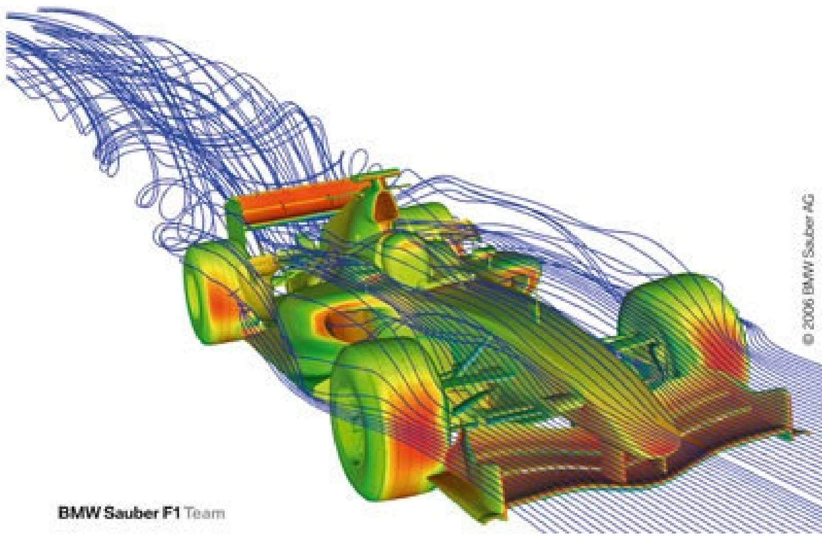
HPC- Applications



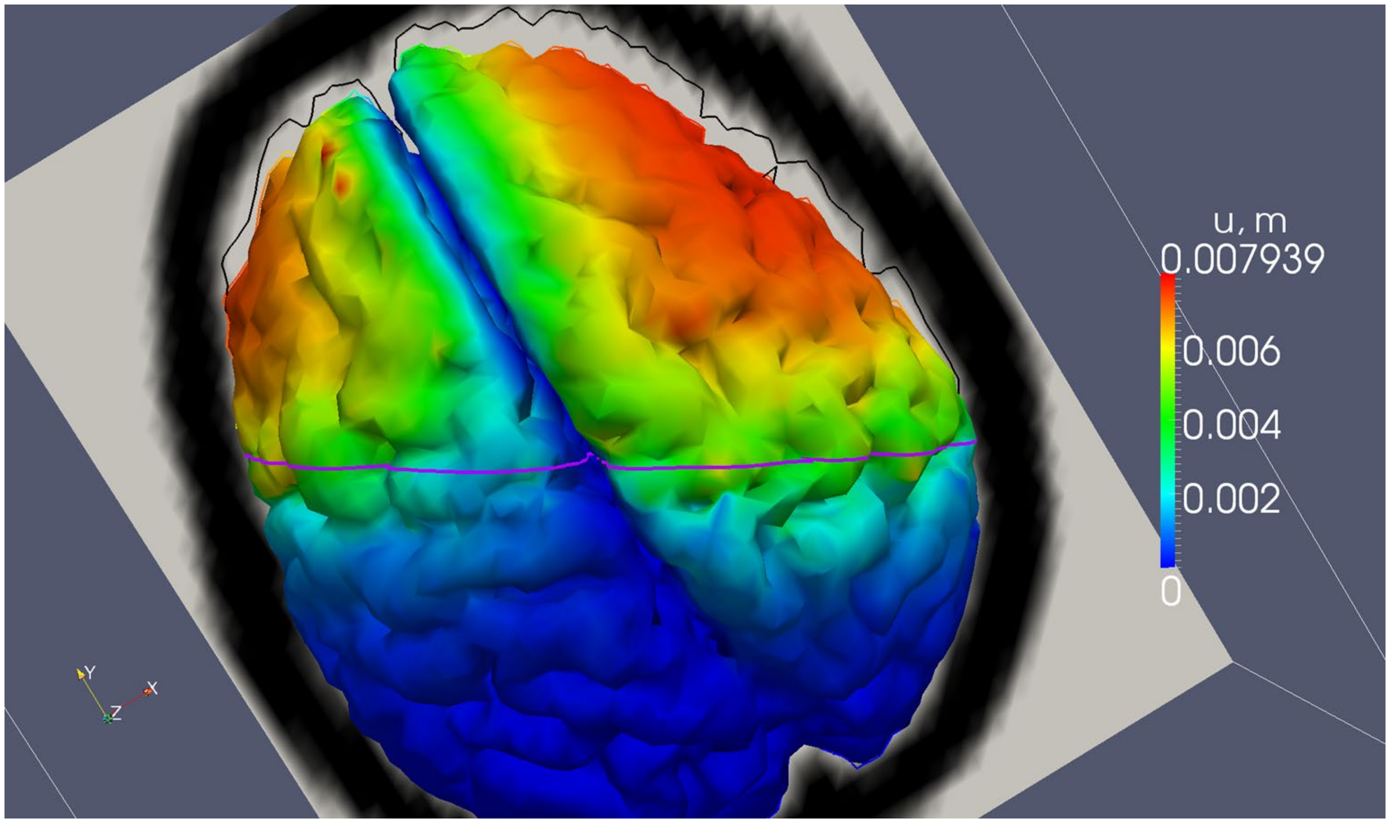
HPC - Applications

- Fluid Dynamics





HPC - Applications



- Brain Simulation

HPC - Applications

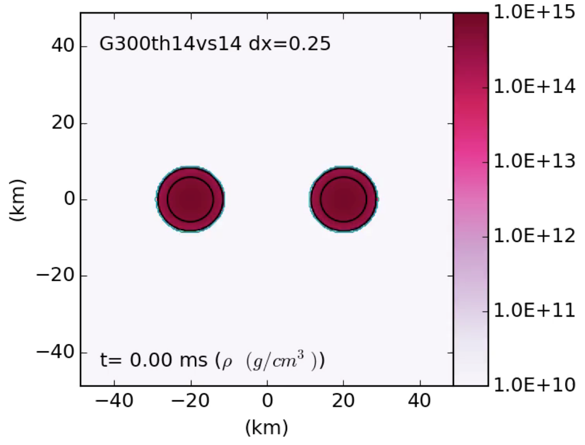
- General relativity

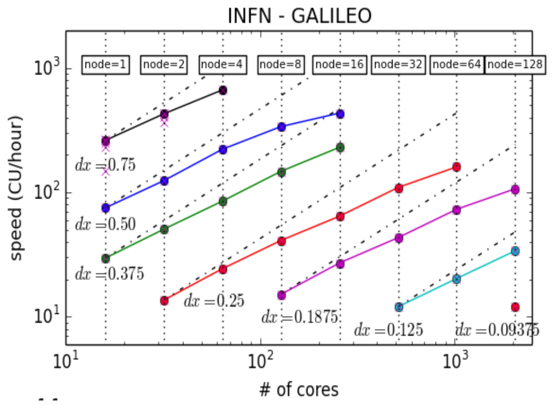
- The scientific case: high resolution simulation of inspiral and merger phase of binary neutron stars system
  - one of source of the gravitational waves that are the observational target of the LIGO/VIRGO experiment

- **Computation performed using The Einstein ToolKit**

- **Result obtained on Galileo at CINECA**

© Roberto De Pietri , Roberto Alfieri - INFN Parma and Parma University, 2012





Speedup of an application

- Speedup: measures the increased performance in running in parallel on P processors

$$S(P) = \frac{T_{Seq}(1)}{T_{Par}(P)}$$

- Perfect Linear Speedup: no overhead due to parallelism. Speedup equals the number of processors

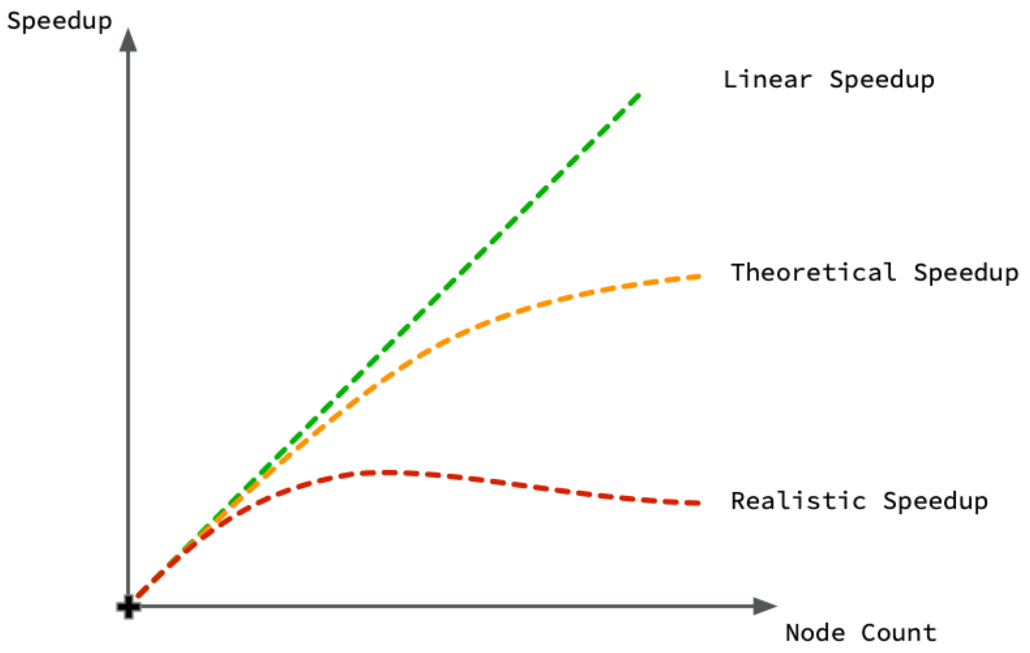
$$S(P) = P$$

Parallel computation efficiency

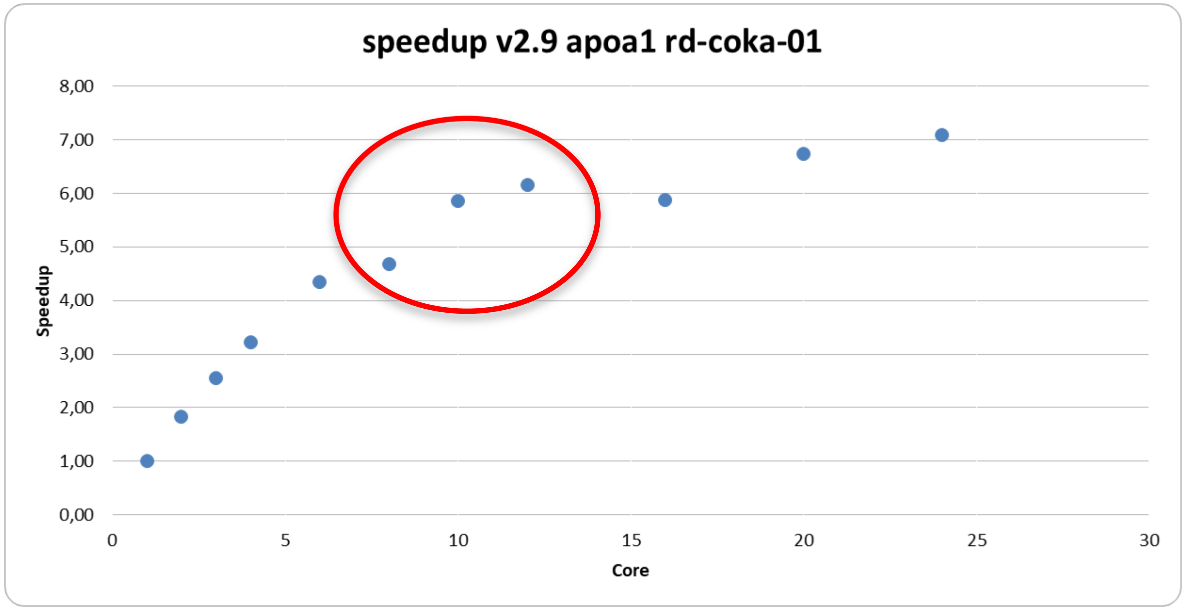
- Efficiency: measures how well the hardware resources (processors) are utilized

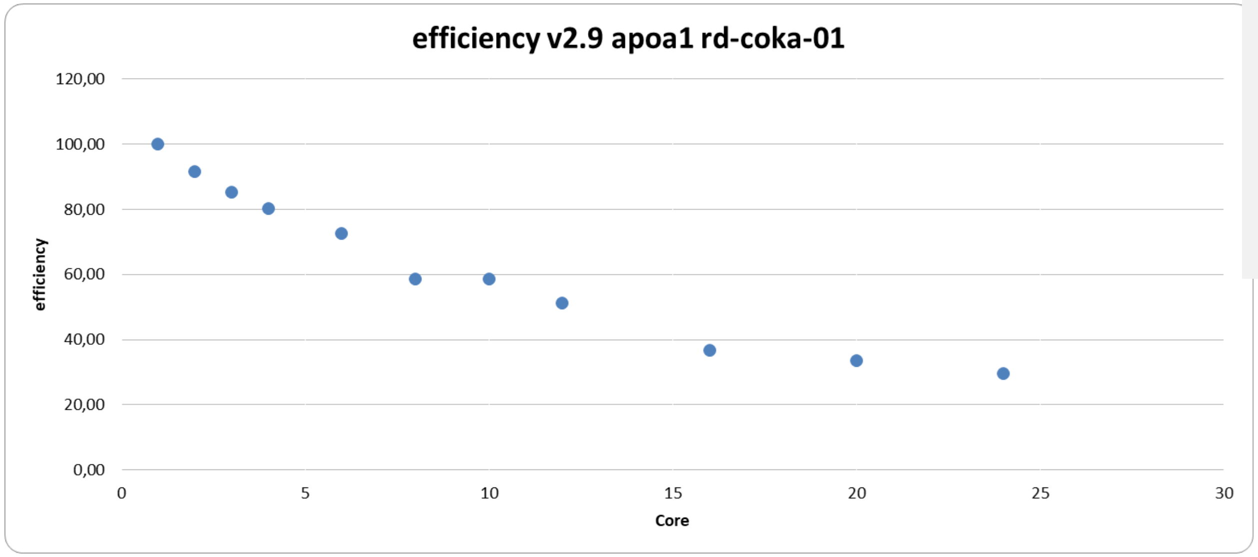
$$\varepsilon = \frac{T_{Seq}}{P * T_{Par}(P)} = \frac{S(P)}{P}$$

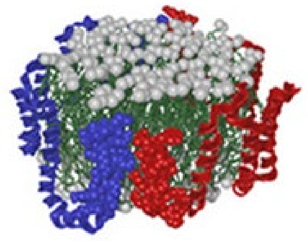
T = Elapsed Time
P = Number of processors Used



Speedup examples – NAMD APOA1





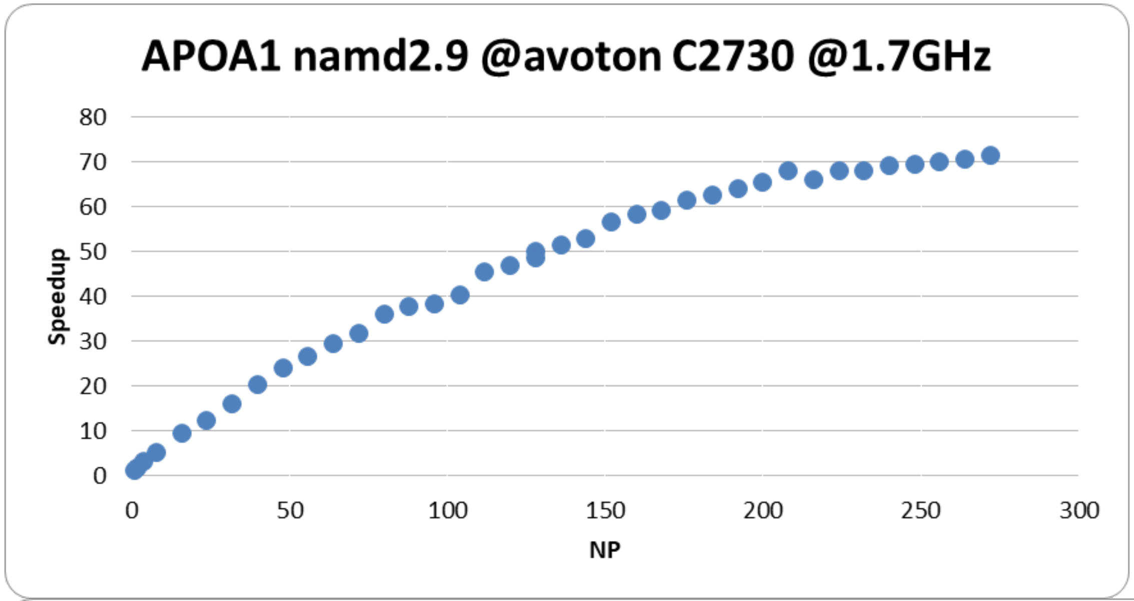


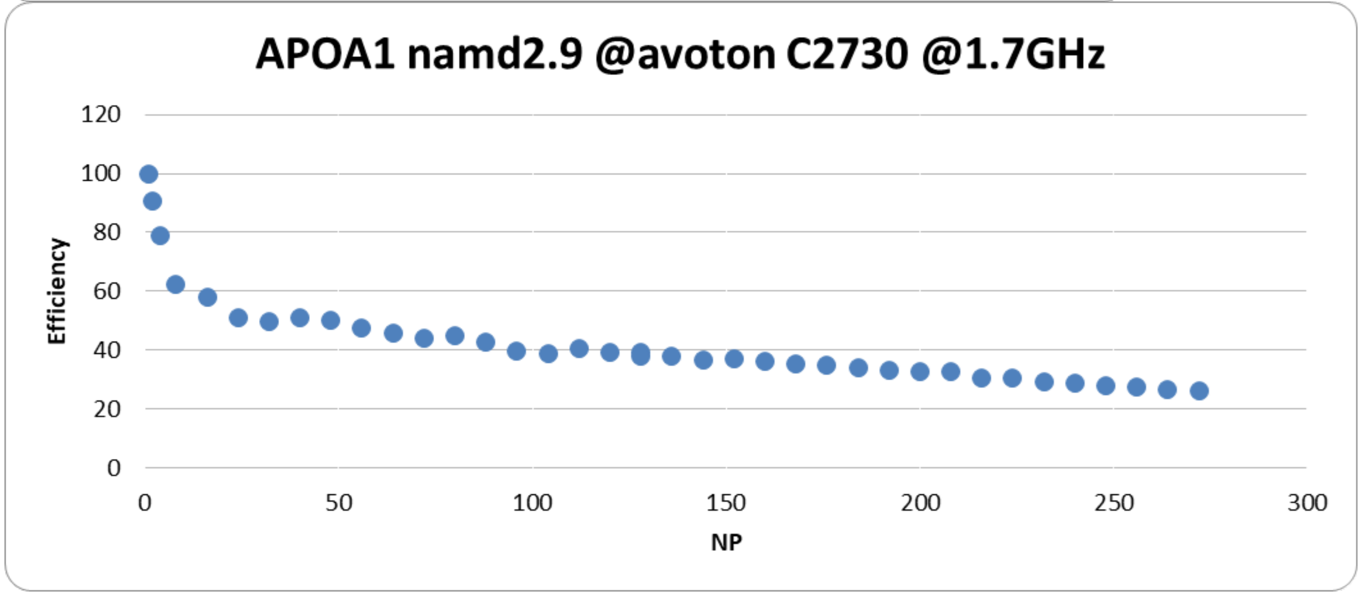
# of Atoms	92,224

APOA1

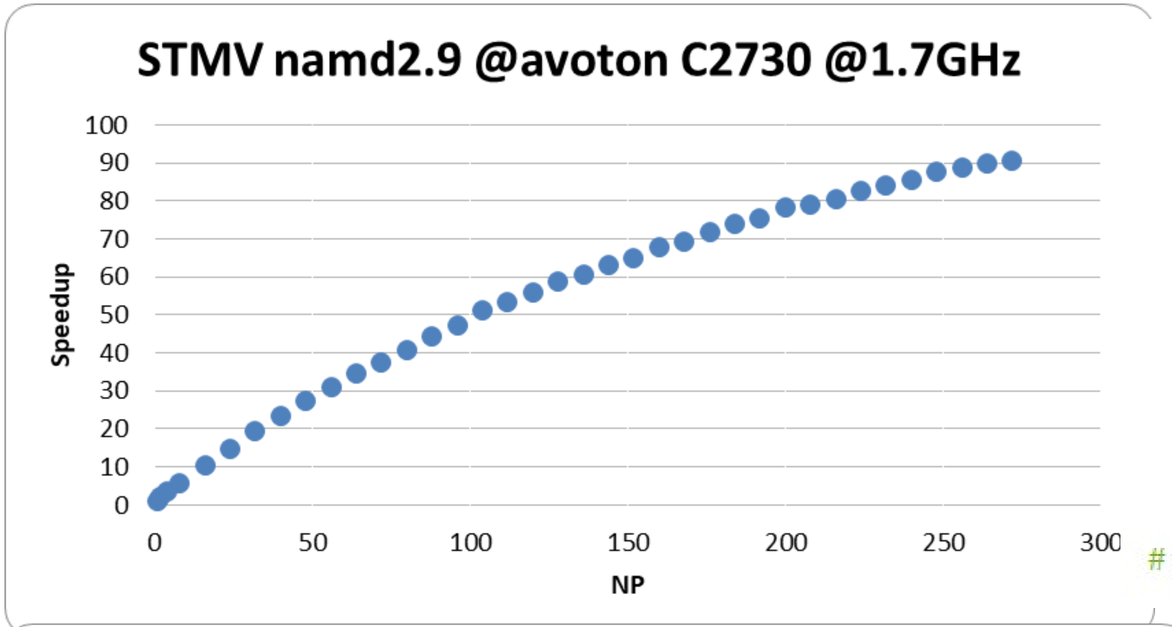
Apolipoprotein A1 (ApoA1) is the major protein component of high-density lipoprotein (HDL) in the bloodstream and plays a specific role in lipid metabolism. The ApoA1 benchmark consists of 92,224 atoms and has been a standard NAMD cross-platform benchmark for years.

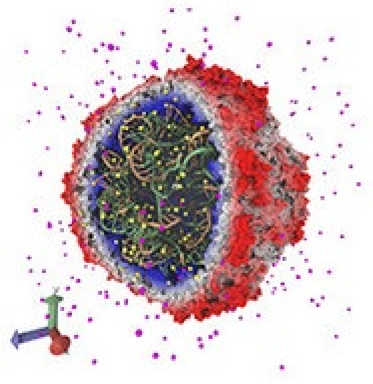
Speedup example – NAMD APOA1



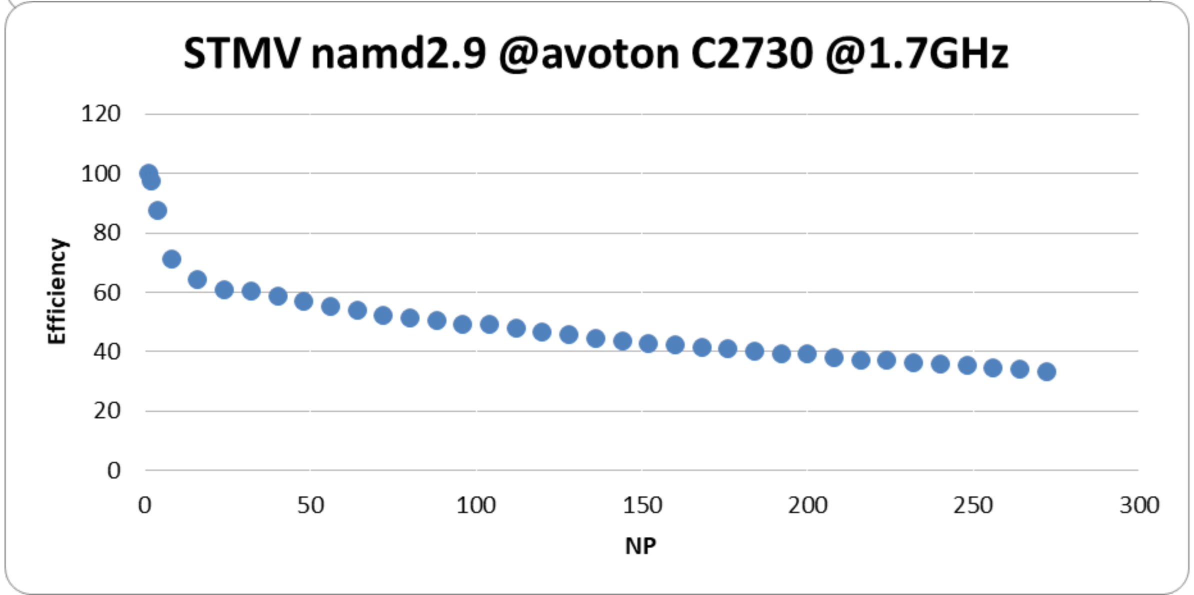


Speedup – NAMD STMV





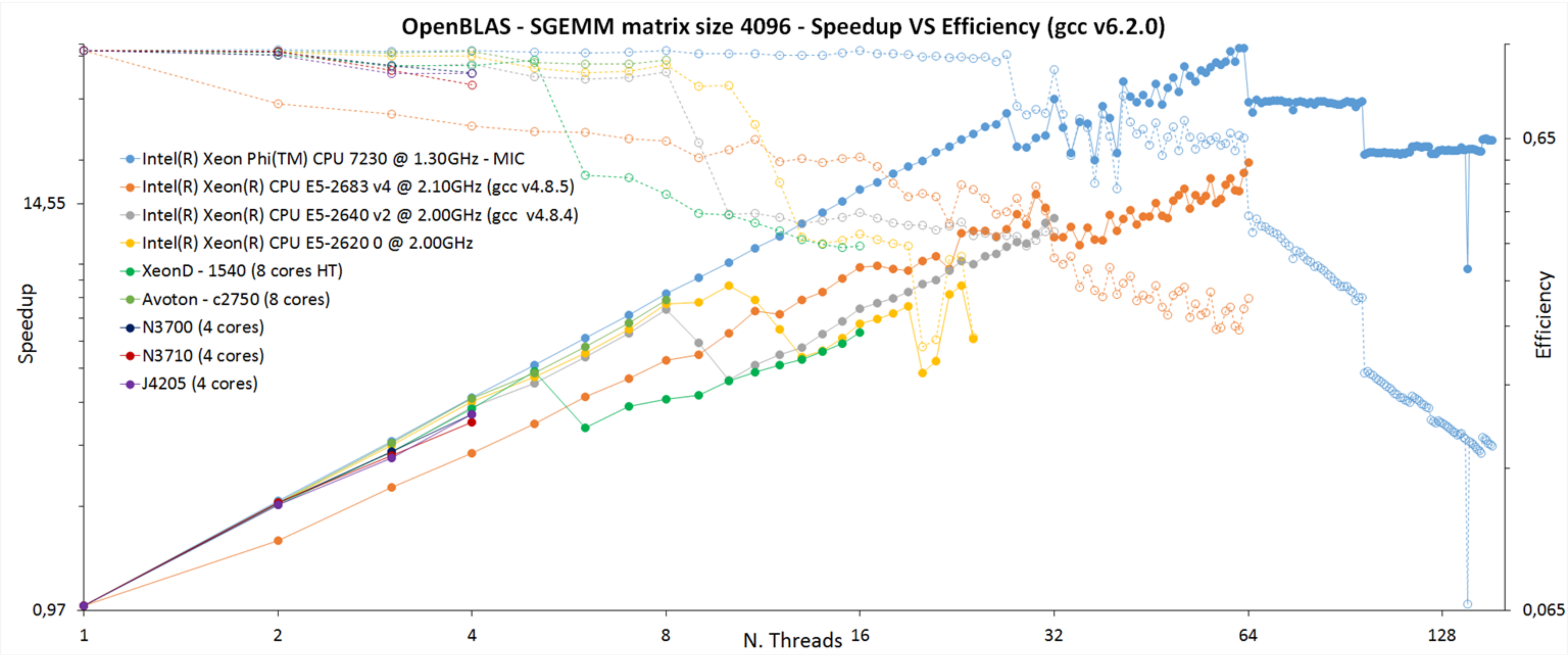
# of Atoms 1,066,628

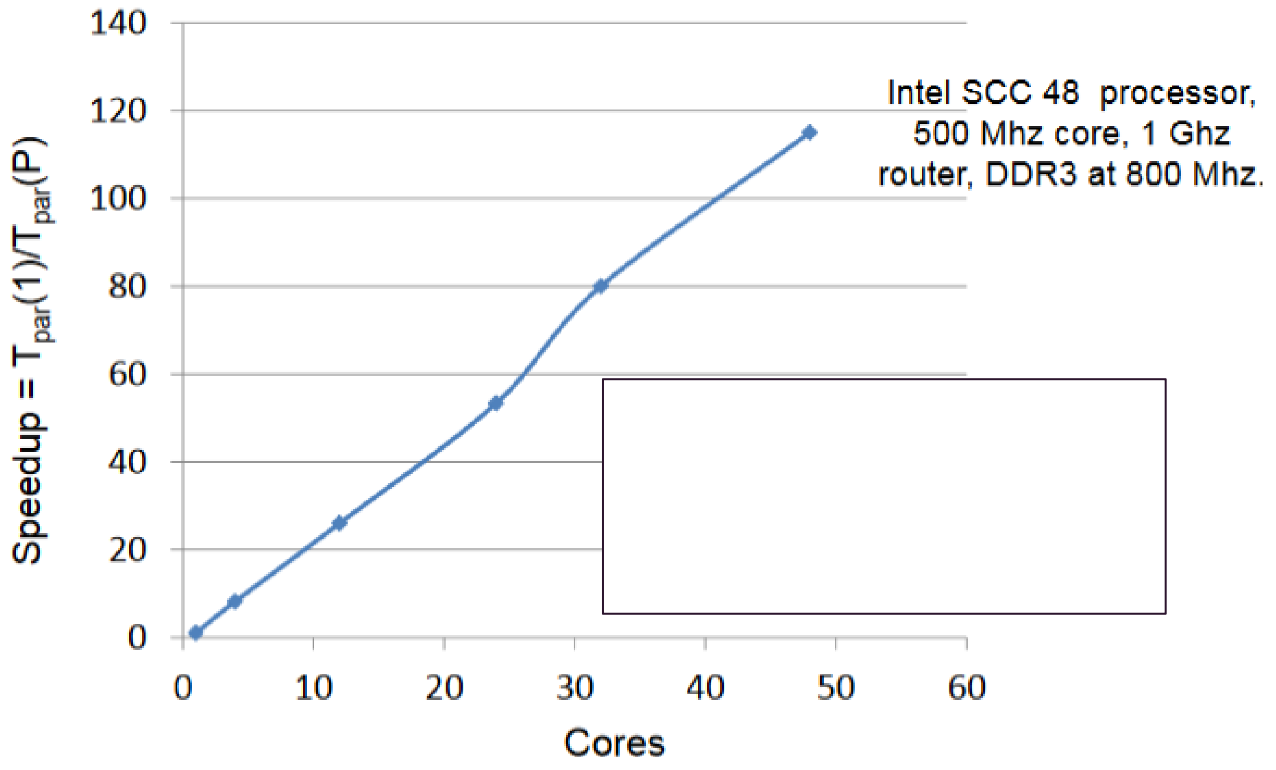


STMV

Satellite Tobacco Mosaic Virus (STMV) is a small, icosahedral plant virus that worsens the symptoms of infection by Tobacco Mosaic Virus (TMV). SMTV is an excellent candidate for research in molecular dynamics because it is relatively small for a virus and is on the medium to high end of what is feasible to simulate using traditional molecular dynamics in a workstation or a small server.

KNL test with synthetic tests - SGEMM



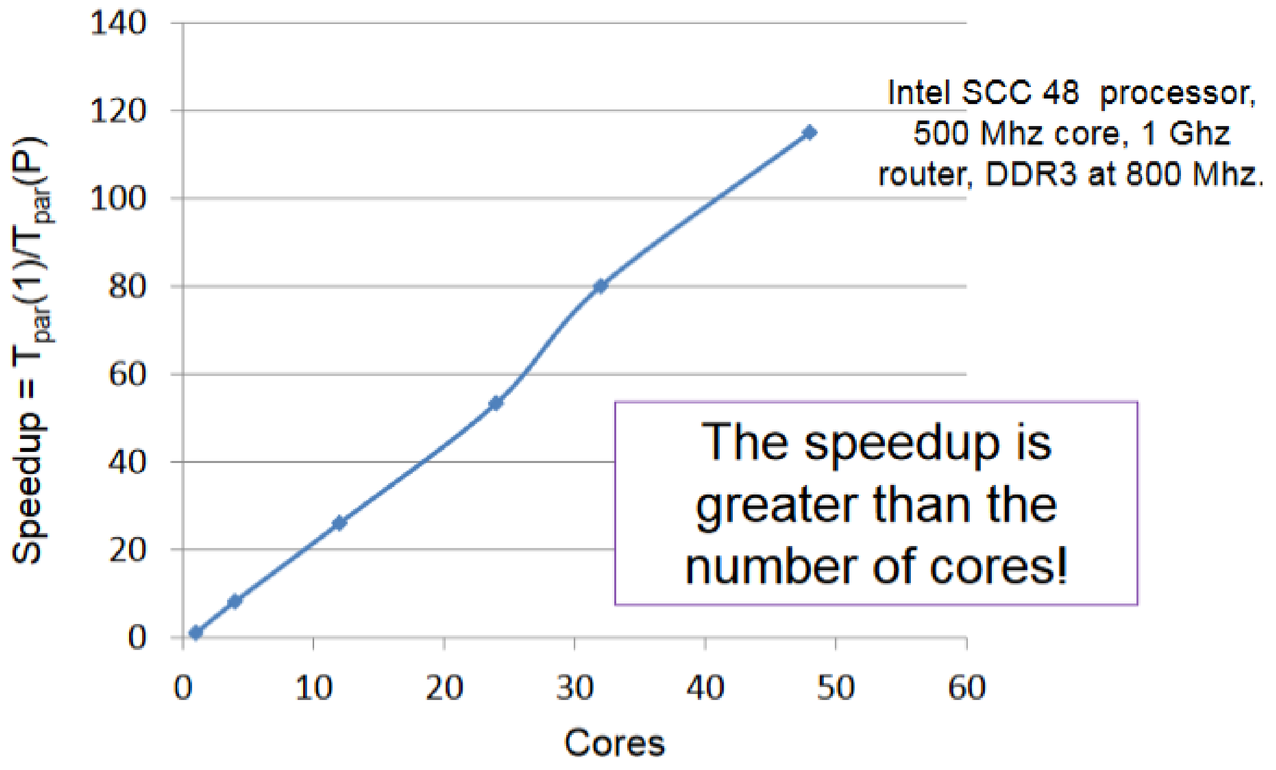


The 48-core SCC processor: the programmer's view, T, G. Mattson, R. F. Van der Wijngaart, M. Riepen, T. Lehnig, P. Brett, W. Haas, P. Kennedy, J. Howard, S. Vangal, N. Borkar, G. Ruhl, S. Dighe, Proceedings SC10, New Orleans 2010

46

Super Linear Speedup

HP Linpack benchmark, order 1000 matrix (solve a dense system of linear equations … the dense linear algebra computational pattern).



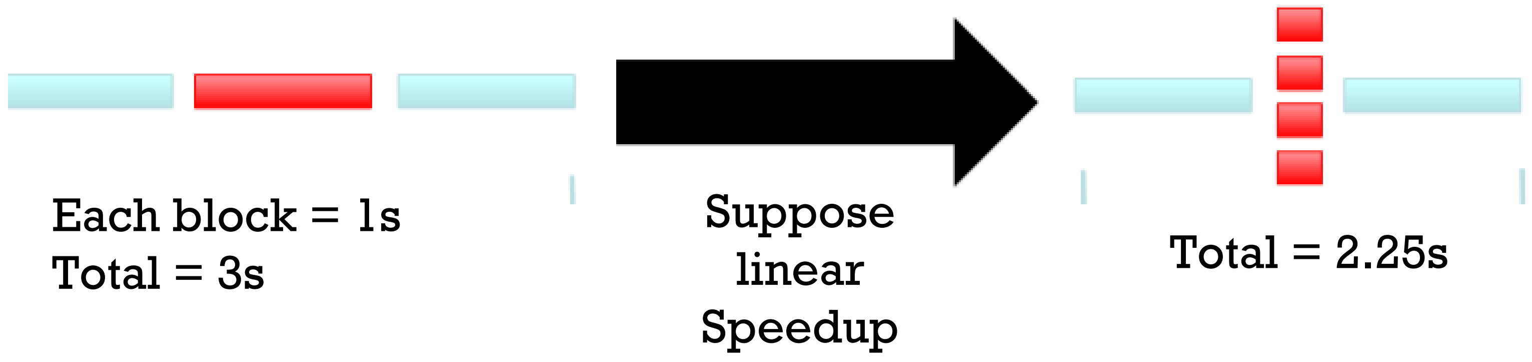
The 48-core SCC processor: the programmer's view, T, G. Mattson, R. F. Van der Wijngaart, M. Riepen, T. Lehnig, P. Brett, W. Haas, P. Kennedy, J. Howard, S. Vangal, N. Borkar, G. Ruhl, S. Dighe, Proceedings SC10, New Orleans 2010

46

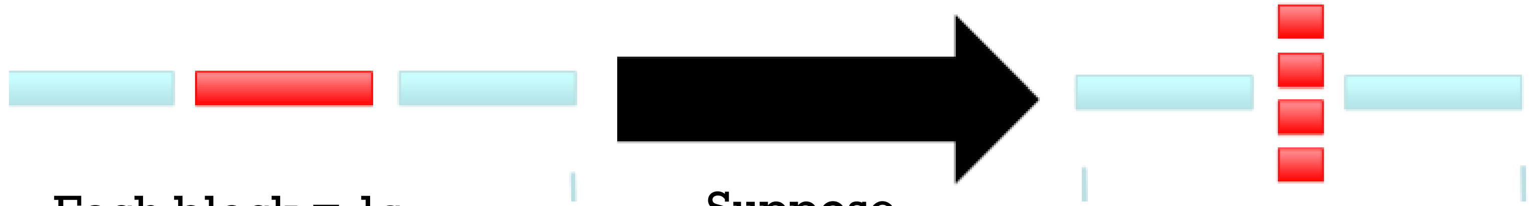
Normally due to memory caching effects

Amdahl's Law

- Predicts the theoretical speedup when using multiple processors



Amdahl's Law



Each block = 1s
Total = 3s

Suppose
linear
Speedup

Total = 2.25s

$$Time_{par}(P) = (serial\_fraction + \frac{parallel\_fraction}{P}) * Time_{seq}$$

serial_fraction is α and the parallel_fraction is $(1- \alpha)$.

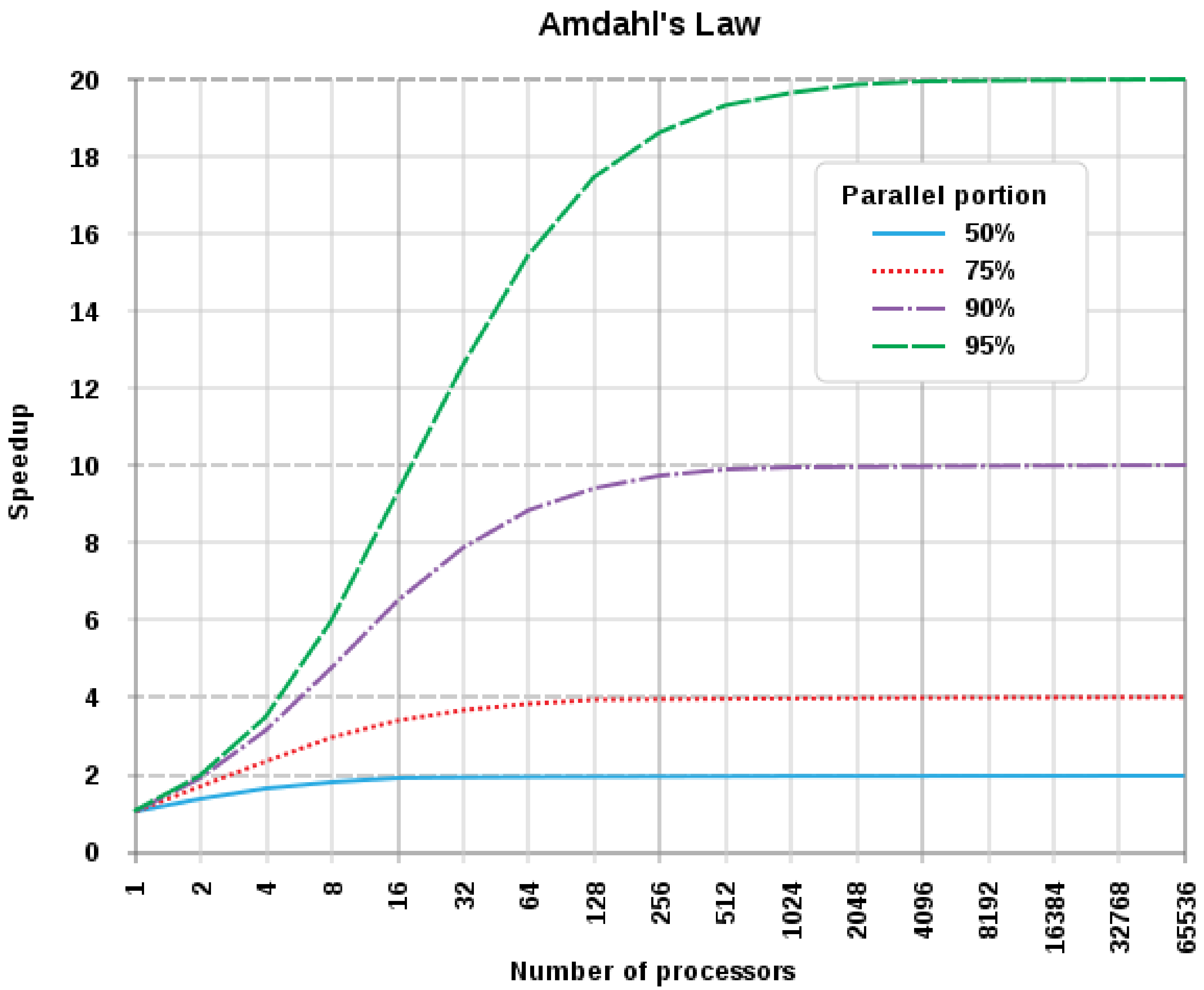
$$S(P) = \frac{Time_{seq}}{Time_{par}(P)} = \frac{Time_{seq}}{\left(\alpha + \frac{1-\alpha}{P}\right) * Time_{seq}} = \frac{1}{(\alpha + \overset{0}{\cancel{\frac{1-\alpha}{P}}})}$$

$$\lim_{P \to \infty} \frac{1-\alpha}{P} = 0$$

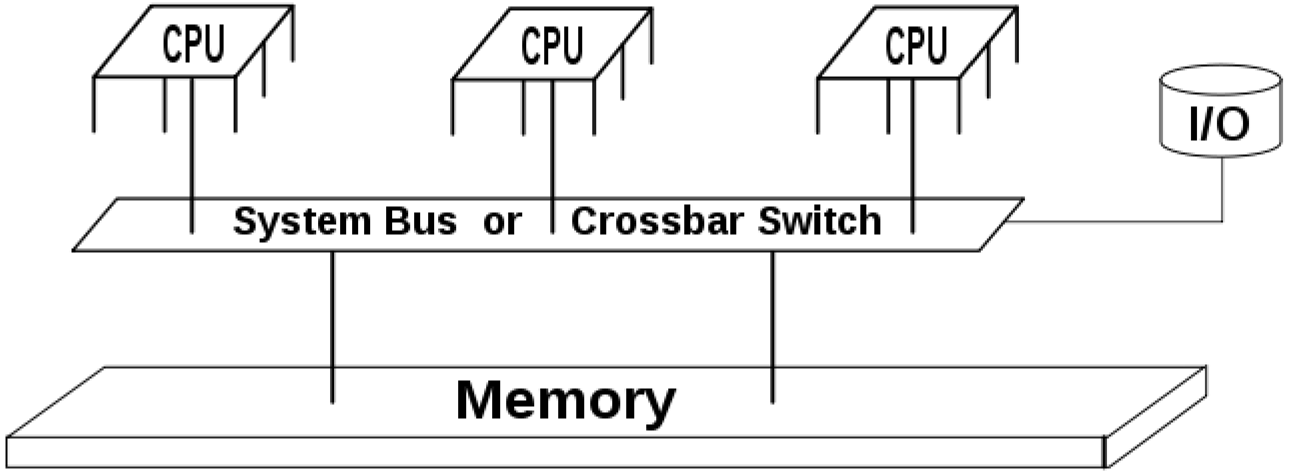
The maximum possible speedup is: $S = \frac{1}{\alpha}$ $\leftarrow$ $\boxed{\text{Amdahl's Law}}$

+

Amdahl's Law


Amdahl's Law
Parallel portion
50%
75%
90%
95%
Speedup
0
2
4
6
8
10
12
14
16
18
20
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
Number of processors

Shared Memory Systems



Shared Memory Systems

- Shared memory is memory that may be simultaneously accessed by multiple programs with an intent to provide communication among them or avoid redundant copies
- Shared memory is an efficient means of passing data between programs
- Depending on context, programs may run on a single processor or on multiple separate processors
- Using memory for communication inside a single program, e.g. among its multiple threads, is also referred to as shared memory

Shared Memory Systems

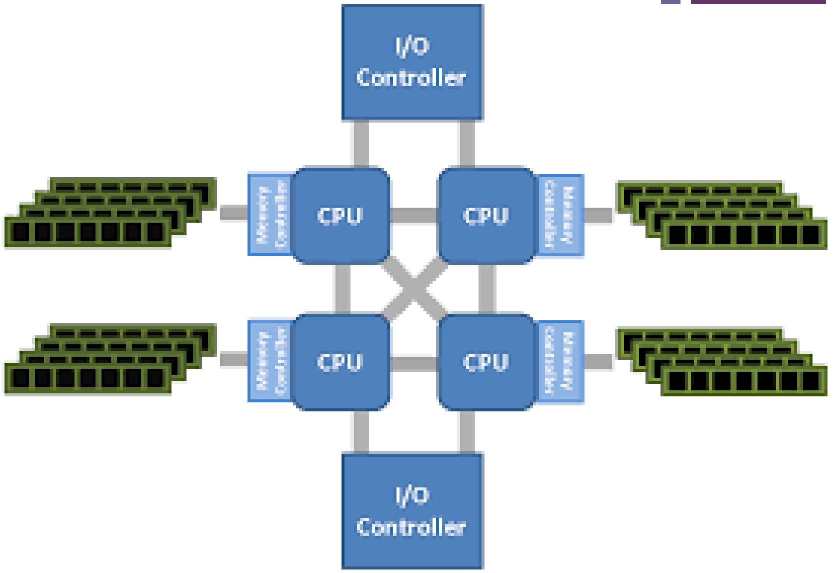
- Shared memory systems may use uniform memory access architecture (UMA): all the processors share the physical memory uniformly
- Non-uniform memory access (NUMA): memory access time depends on the memory location relative to a processor
- A shared memory system is relatively easy to program since all processors share a single view of data and the communication between processors can be as fast as memory accesses to a same location.
- The issue with shared memory systems is that many CPUs need fast access to memory and will likely cache memory
  - Issues that may arise:
    - Cache coherency
    - Race conditions

NUMA Architecture programming

- A programmer can set an allocation policy for its program using a component of NUMA API called libnuma.
  - a user space shared library that can be linked to applications
  - provides explicit control of allocation policies to user programs.

- The NUMA execution environment for a process can also be set up by using the numactl tool

- Numactl can be used to control process mapping to cpuset and restrict memory allocation to specific nodes without altering the program's source code



http://halobates.de/numaapi3.pdf

Shared Memory programming: OpenMP

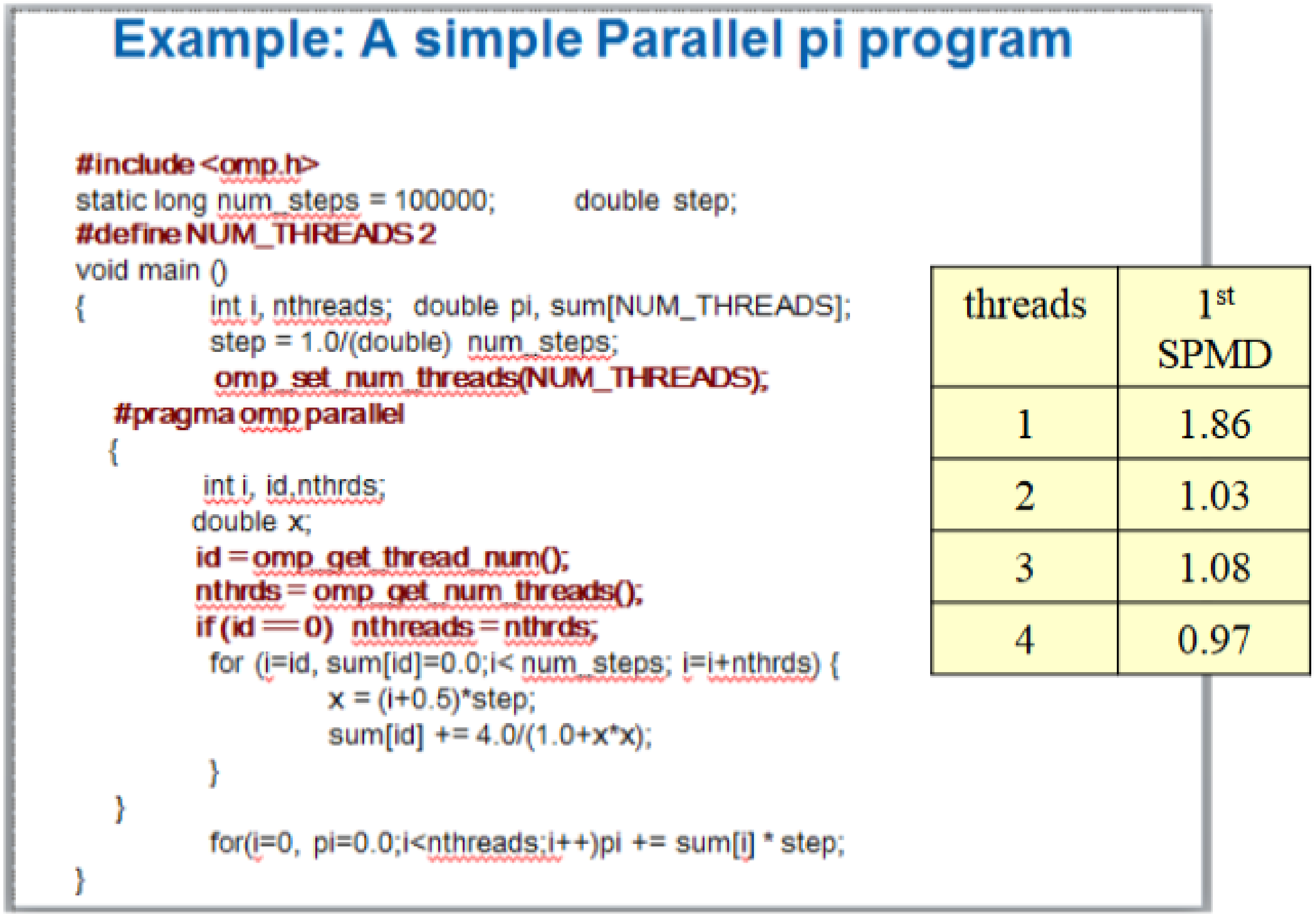
- An API for Writing Multithreaded Applications
- A set of compiler directives and library routines for parallel application programmers
- Greatly simplifies writing multi-threaded (MT) programs in Fortran, C and C++

OpenMP

- A multi-threading, shared address model
- Threads communicate by sharing variables
- Unintended sharing of data causes race conditions
- Race condition: when the program’s outcome changes as the threads are scheduled differently
- To control race conditions:
  - Use synchronization to protect data conflicts
    - Synchronization is an expensive operation

OpenMP example

- Original Serial pi program with 100000000 steps ran in 1.83 seconds.

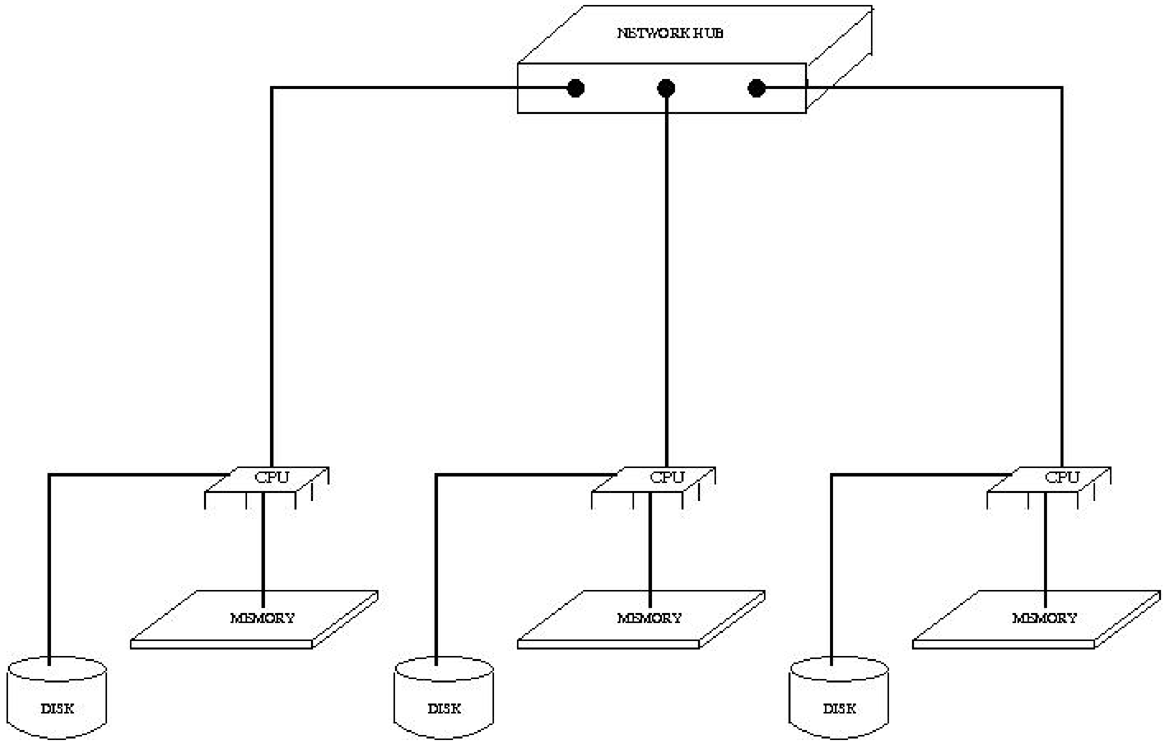


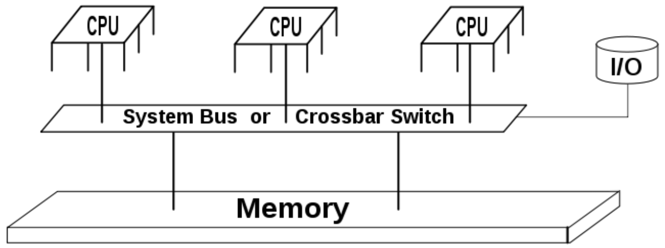
Example: A simple Parallel pi program

```
#include <omp.h>
static long num_steps = 100000;        double  step;
#define NUM_THREADS 2
void main ()
{             int i, nthreads;  double  pi, sum[NUM_THREADS];
              step = 1.0/(double)  num_steps;
              omp_set_num_threads(NUM_THREADS);
  #pragma omp parallel
  {
             int i, id,nthrds;
            double  x;
            id = omp_get_thread_num();
            nthrds = omp_get_num_threads();
            if (id == 0)  nthreads = nthrds;
             for  (i=id, sum[id]=0.0;i< num_steps; i=i+nthrds) {
                       x = (i+0.5)*step;
                       sum[id] += 4.0/(1.0+x*x);
             }
  }
             for(i=0,  pi=0.0;i<nthreads;i++)pi += sum[i] * step;
}
```

threads	1st SPMD
1	1.86
2	1.03
3	1.08
4	0.97

+
Distributed memory systems





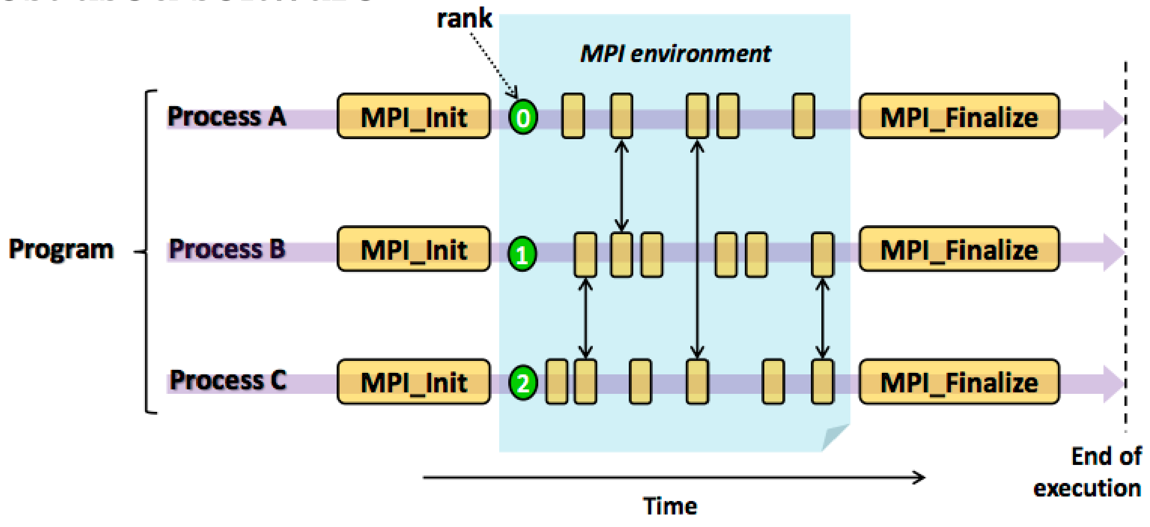
+

Distributed memory systems

- Distributed memory refers to a multiprocessor computer system in which each processor has its own private memory
- Computational tasks can only operate on local data
- if remote data is required, the computational task must communicate with one or more remote processors
- In contrast, a shared memory multiprocessor offers a single memory space used by all processors

Programming model for distributed memory systems: MPI

- The program consists of a collection of named processes
  - Number of processors is typically fixed at startup
- Each process has a local address space – no physical memory is shared
- Communication happens by explicit send/receive statements
  - MPI is the most used software



MPI

- MPI: An API for Writing Clustered Applications
- A library of routines to coordinate the execution of multiple processes
- Provides point to point and collective communication in Fortran, C and C++

+

Sending and Receiving Data

```
int MPI_Send (void* buf, int count,
        MPI_Datatype datatype, int dest,
        int tag, MPI_Comm comm)

int MPI_Recv (void* buf, int count,
        MPI_Datatype datatype, int source,
        int tag, MPI_Comm comm,
        MPI_Status* status)
```

- **MPI_Send** performs a blocking send of the specified data (“count” copies of type “datatype,” stored in “buf”) to the specified destination (rank “dest” within communicator “comm”), with message ID “tag”

- **MPI_Recv** performs a blocking receive of specified data from specified source whose parameters match the send; information about transfer is stored in “status”

By “blocking” we mean the functions return as soon as the buffer, “buf”, can be safely used.

Reduction

```
        int MPI_Reduce (void* sendbuf,
              void* recvbuf, int count,
              MPI_Datatype datatype, MPI_Op op,
              int root, MPI_Comm comm)
```

- **`MPI_Reduce`** performs specified reduction operation on specified data from all processes in communicator, places result in process "root" only.
- **`MPI_Allreduce`** places result in all processes (avoid unless necessary)

Operation	Function
`MPI_SUM`	Summation
`MPI_PROD`	Product
`MPI_MIN`	Minimum value
`MPI_MINLOC`	Minimum value and location
`MPI_MAX`	Maximum value
`MPI_MAXLOC`	Maximum value and location
`MPI_LAND`	Logical AND

Operation	Function
`MPI_BAND`	Bitwise AND
`MPI_LOR`	Logical OR
`MPI_BOR`	Bitwise OR
`MPI_LXOR`	Logical exclusive OR
`MPI_BXOR`	Bitwise exclusive OR
User-defined	It is possible to define new reduction operations

Deadlock

```
if (rank == 0) {
     MPI_Send(..., 1, tag, MPI_COMM_WORLD);
     MPI_Recv(..., 1, tag, MPI_COMM_WORLD, &status);
} else if (rank == 1) {
     MPI_Send(..., 0, tag, MPI_COMM_WORLD);
     MPI_Recv(..., 0, tag, MPI_COMM_WORLD, &status);
}
```

Consider the following and assume that the MPI_Send does not complete until the corresponding MPI_Recv is posted and visa versa. The MPI_Send commands will never be completed and the program will deadlock.

https://www.dartmouth.edu/~rc/classes/intro_mpi/mpi_race_conditions.html

+ Deadlock

One way is to reverse the order of one of the send/receive pairs:

```
 if (rank == 0) {
              MPI_Send(..., 1, tag, MPI_COMM_WORLD);
              MPI_Recv(..., 1, tag, MPI_COMM_WORLD, &status);
 } else if (rank == 1) {
              MPI_Recv(..., 0, tag, MPI_COMM_WORLD, &status);
              MPI_Send(..., 0, tag, MPI_COMM_WORLD);
}
```

Another way is to make the send be a non-blocking one (MPI_Isend)

```
if (rank == 0) {
              MPI_Isend(..., 1, tag, MPI_COMM_WORLD, &req);
              MPI_Recv(..., 1, tag, MPI_COMM_WORLD, &status);
              MPI_Wait(&req, &status);
 } else if (rank == 1) {
              MPI_Recv(..., 0, tag, MPI_COMM_WORLD, &status);
              MPI_Send(..., 0, tag, MPI_COMM_WORLD);
}
```

The12 core functions in MPI

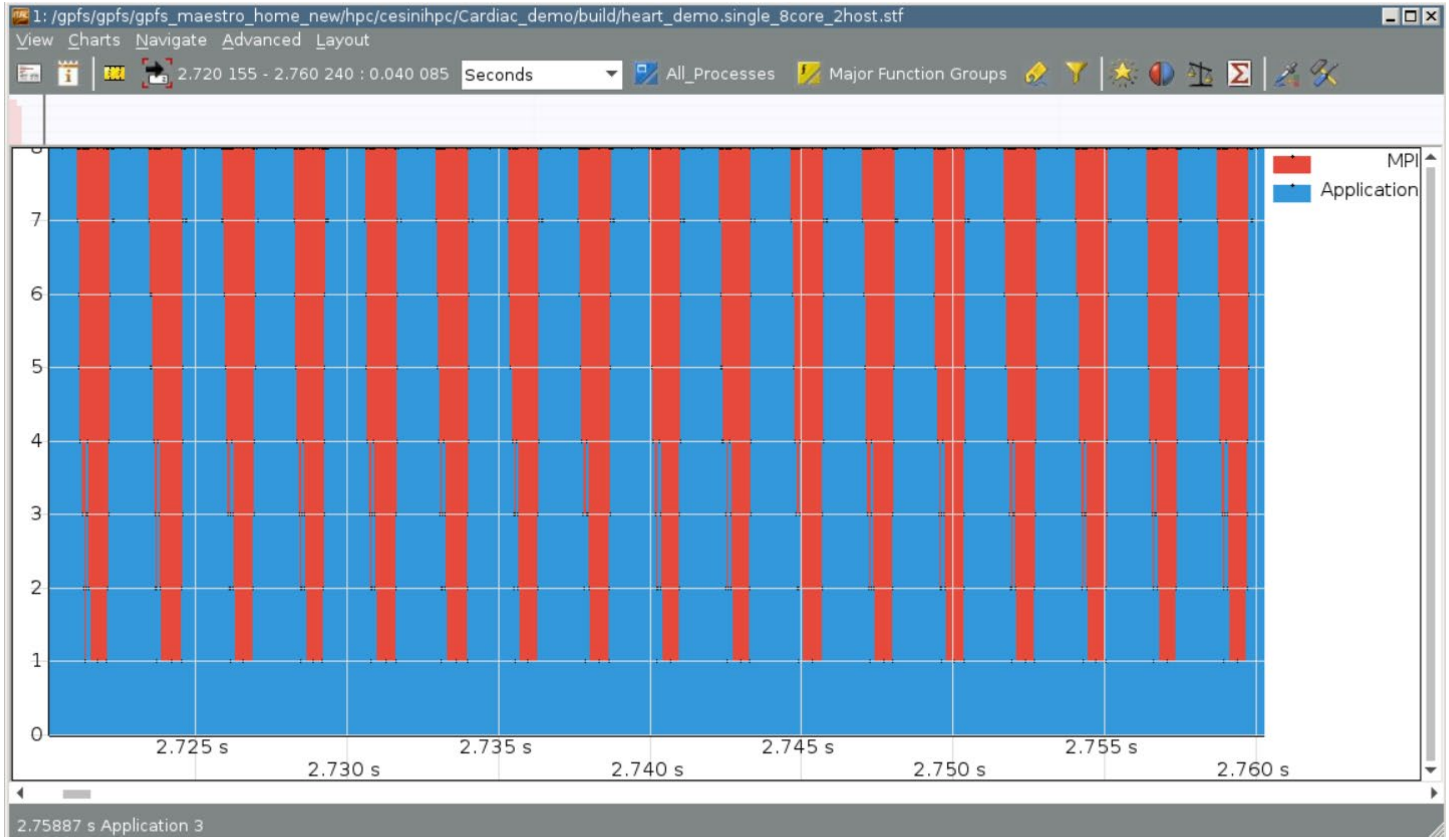
- MPI_Init
- MPI_Finish
- MPI_Comm_size
- MPI_Comm_rank
- MPI_Send
- MPI_Recv
- MPI_Reduce
- MPI_Isend
- MPI_Irecv
- MPI_Wait
- MPI_Wtime
- MPI_Bcast

Should be avoided in favor of Isend, Irecv
For asyncronohous data transfers

Process while transferring

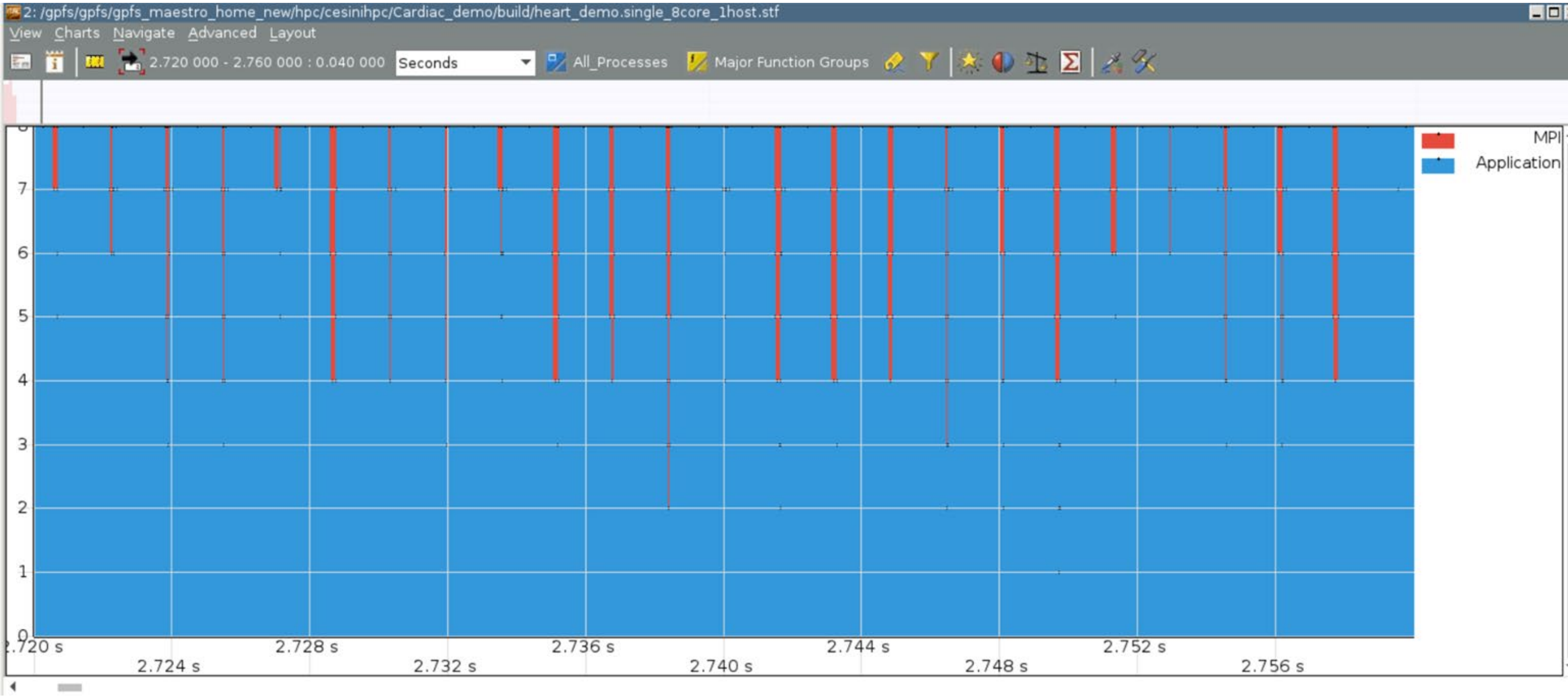
Communication performance in MPI Applications

- 8 process – 2 hosts – MPI messages sent/received over ethernet



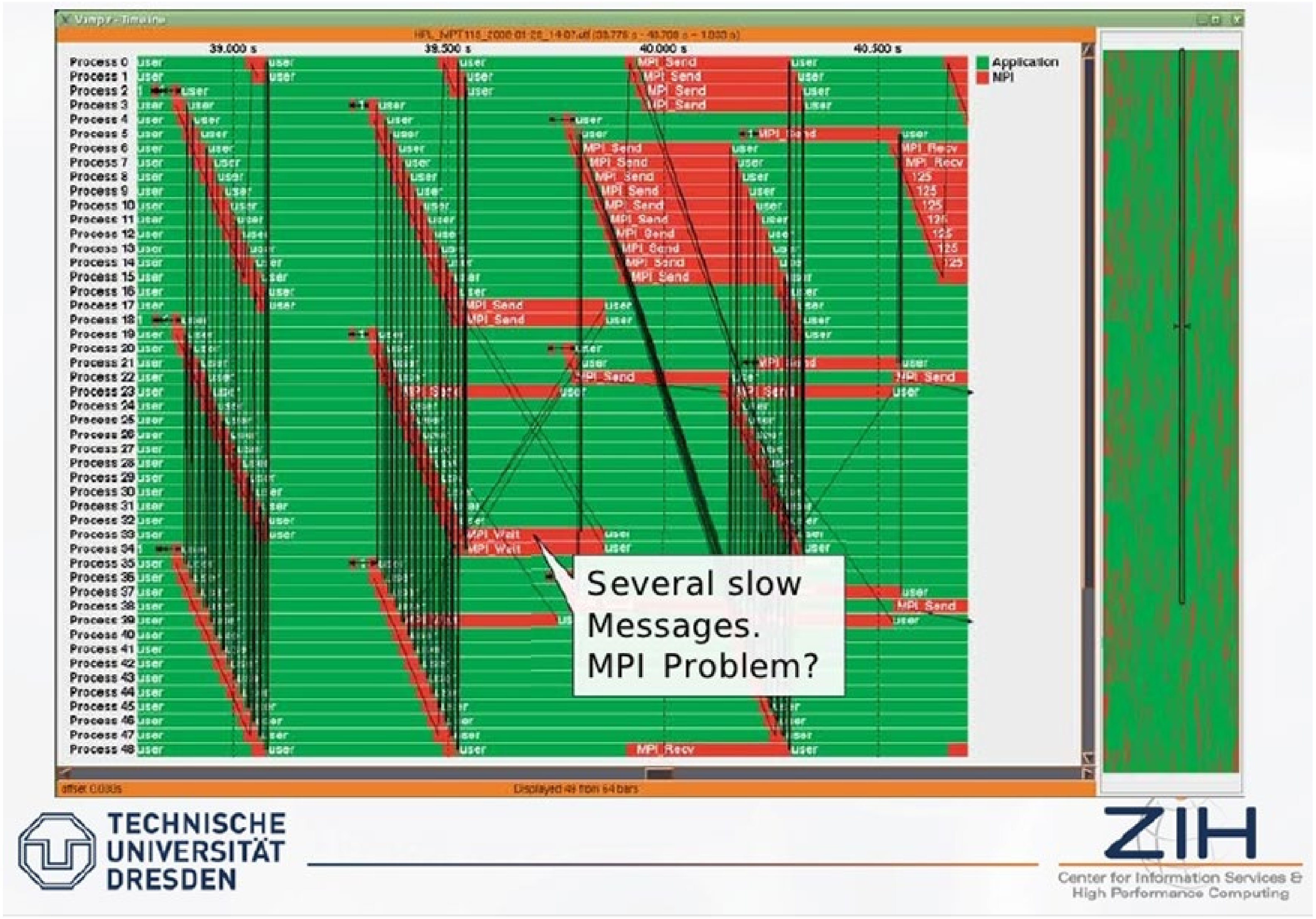
Communication performance in MPI Applications

- 8 process – 1 hosts – MPI messages sent/received via shared memory



+

Communication performance in MPI Applications

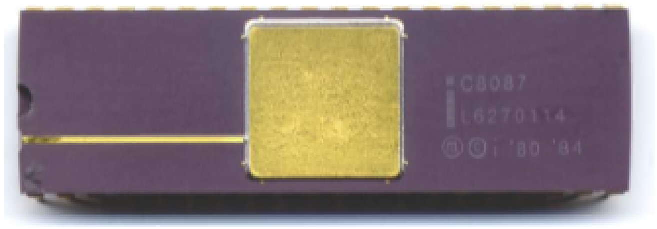

Vampir - Timeline
39.000 s
39.500 s
40.000 s
40.500 s
Application
MPI
Process 0
Process 48
user
MPI_Send
MPI_Recv
MPI_Wait
Several slow Messages. MPI Problem?
Displayed 49 from 64 bars
TECHNISCHE UNIVERSITÄT DRESDEN
ZIH
Center for Information Services & High Performance Computing

HPC - accelerators

Hardware Coprocessors to accelerate FLOPS

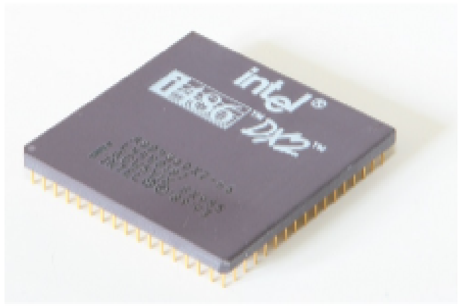
Not a new idea!!

- The coprocessor: 8087 introduced in 1980.
  - The first x87 floating point coprocessor for the 8086 line of microprocessors.
  - Performance enhancements: 20% to 500%, depending on the workload.



- Related Standards:
  - Partnership between industry and academia led to IEEE 754 …. The most important standard in the history of HPC. IEEE 754 first supported by x87.

- Intel® 80486DX, Pentium®, and later processors include floating-point functionality in the CPU … the end of the line for the X87 processors.



Intel® 486DX2™ processor, March 1992.

Intel® Pentium™ processor, Spring 1993.

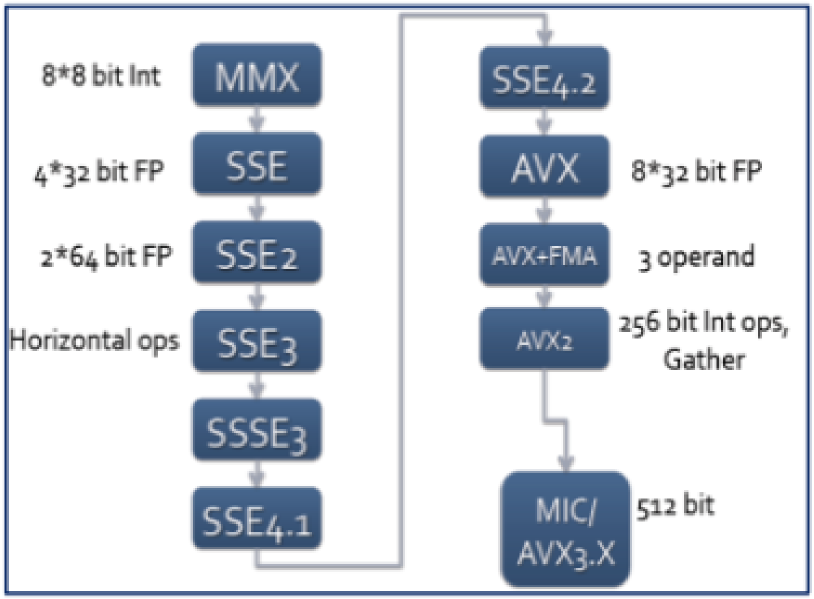
Intel C8087 Math Coprocessor. Source: http://www.cpu-collection.de/?l0=i&i=1698&sd=1

Vector Processing accelerators

- Vector Coprocessor:
  - Vector co-processor (from Sky computer) for Intel iPSC/2 MPP (~1987)
  - Floating point systems array processors (late 80s's)

Absorbed into the CPU

The Intel® i860 processor (early 90's) with integrated vector instructions.



*third party names are the property of their owners

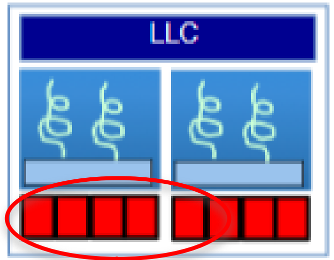
And now vector instructions are a ubiquitous element of CPUs.

© Tim Matson @ESC school

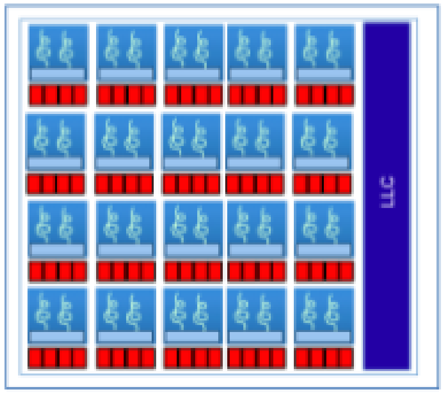
Source: http://www.cpu-collection.de/?l0=co&l1=Intel&l2=i860

Hardware Diversity

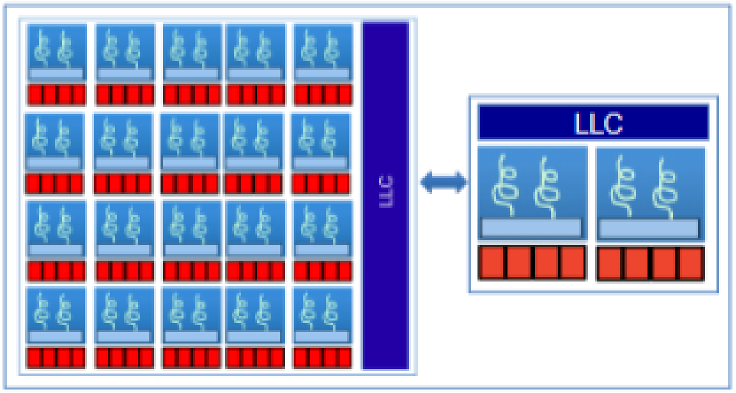
Vector instructions:
SIMD – single instruction multiple data



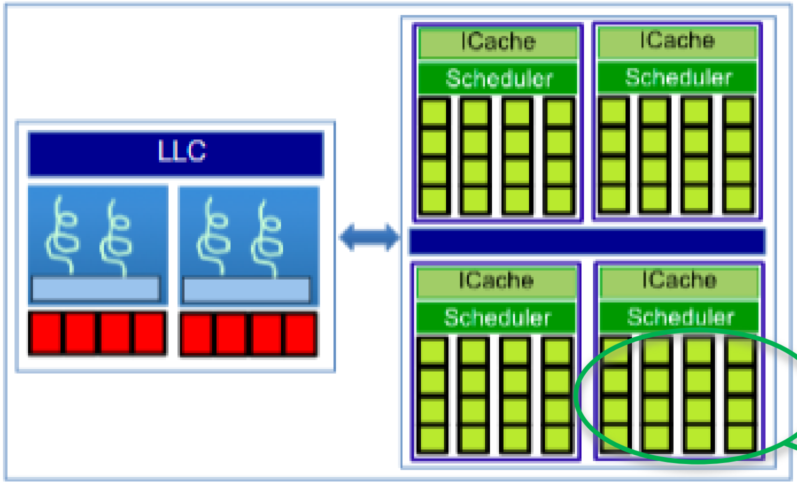
Multicore CPU



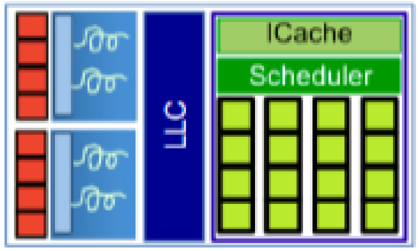
Manycore CPU



Heterogeneous:
CPU + manycore coprocessor



Heterogeneous: CPU+GPU

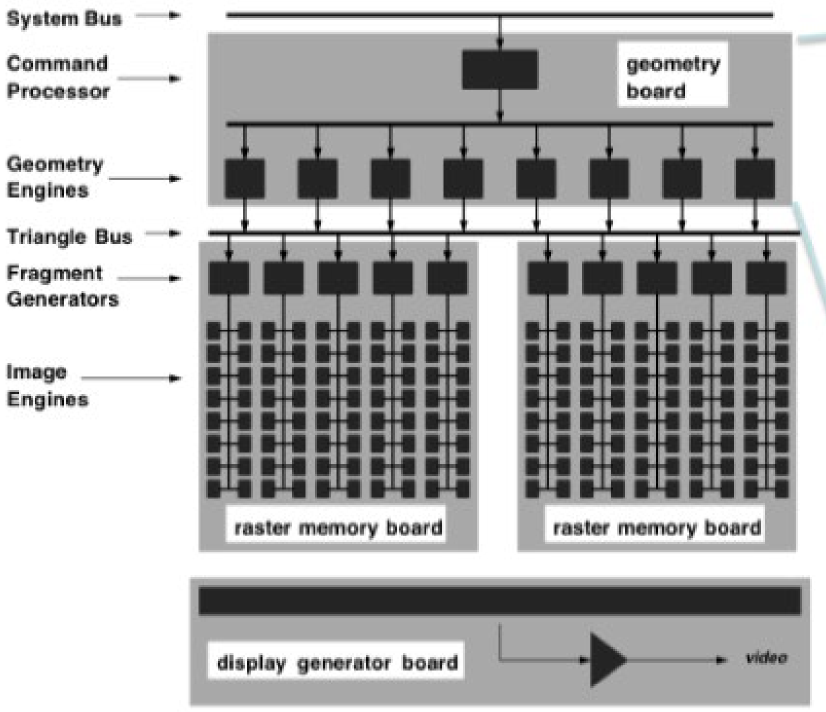


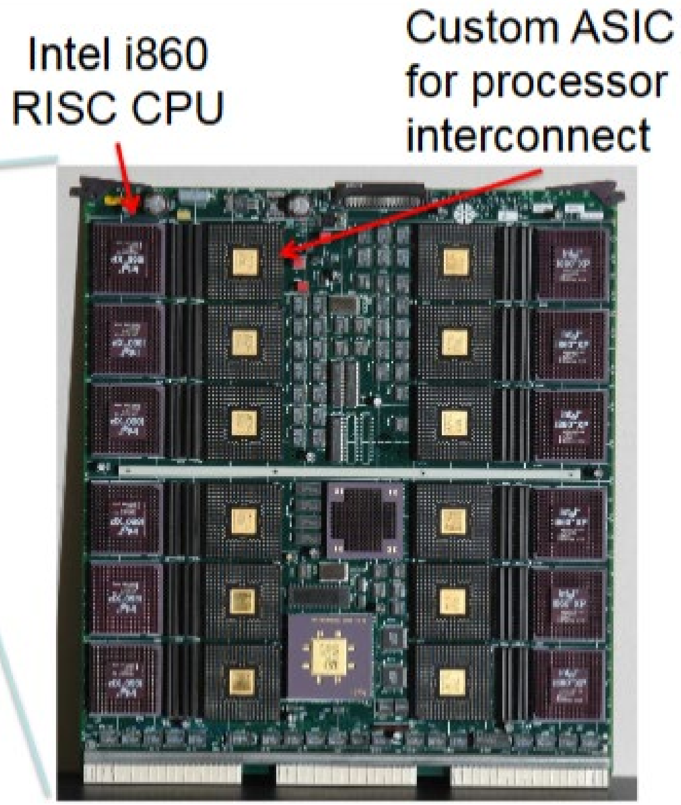
Heterogeneous:
Integrated CPU+GPU

SIMT: Single Instruction Multiple Threads

High-end (historical) programmable GPUs

reduced-instruction-set computing (*RISC*)
application specific integrated circuit (ASIC)

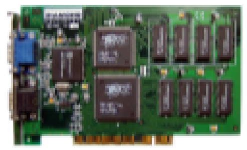




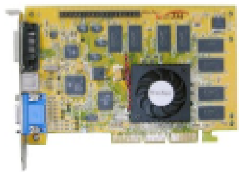
Silicon Graphics RealityEngine GPU
1993

- I860 billed as a "Cray-on-a-chip"
  0.80 micron technology
  2.5M transistors

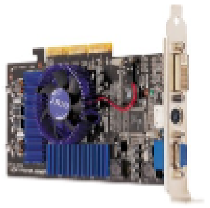
The evolutions of the GPU – before CUDA



1st generation: Voodoo 3dfx (1996)



2nd Generation:
GeForce 256/Radeon 7500 (1998)



3rd Generation: GeForce3/Radeon 8500 (2001). The first GPU to allow a limited programmability in the vertex pipeline.

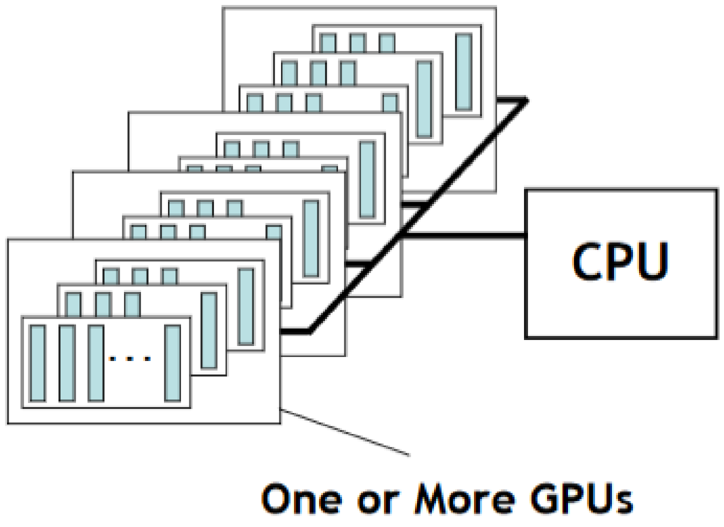
4th Generation: Radeon 9700/GeForce FX (2002): The first generation of "fully-programmable" graphics cards.

5th Generation: GeForce 8800/HD2900 (2006) and the birth of CUDA

Third party names are the property of their owners

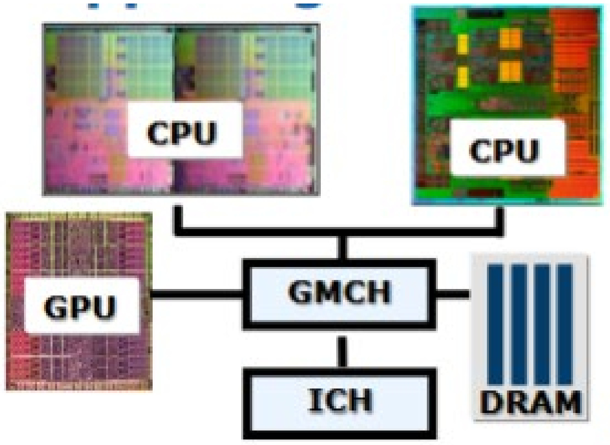
+
GPU-Based Platforms

GPU Platform Model



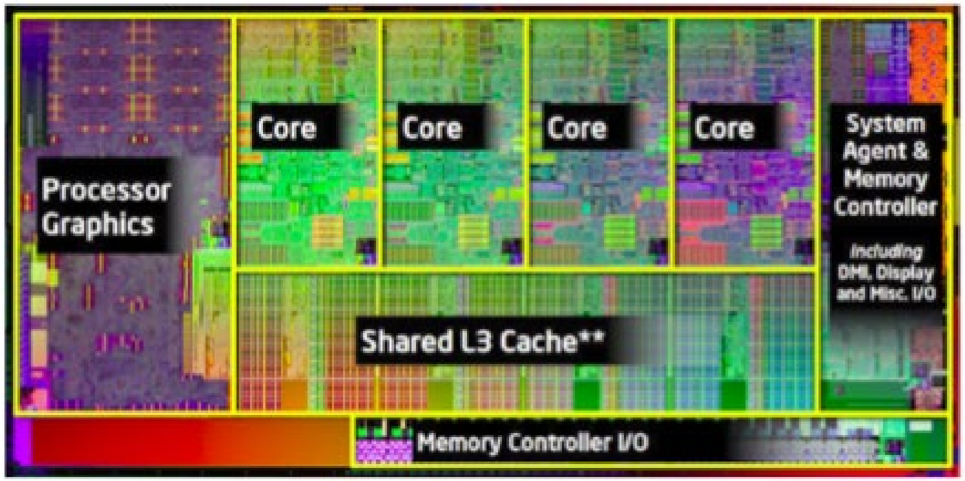
- The GPUs are driven by a CPU which …
  - Manages the code to execute on the GPUs
  - Maintains a queue of kernels to execute
  - Manages memory on the GPU and movement between the CPU and the GPU

Heterogeneity in the same silicon die



- A modern platform has:
  - CPU(s)
  - GPU(s)
  - DSP processors
  - … other?

- Current designs put this functionality onto a single chip … mitigates the PCIe bottleneck in GPGPU computing!



Intel® Core™ i5-2500K Desktop Processor (Sandy Bridge) Intel HD Graphics 3000 *(2011)*

GMCH = graphics memory control hub,
ICH = Input/output control hub

GPU/Accelerators Speedup

- Optimized applications are either compute or bandwidth bounded
- For memory bound applications:

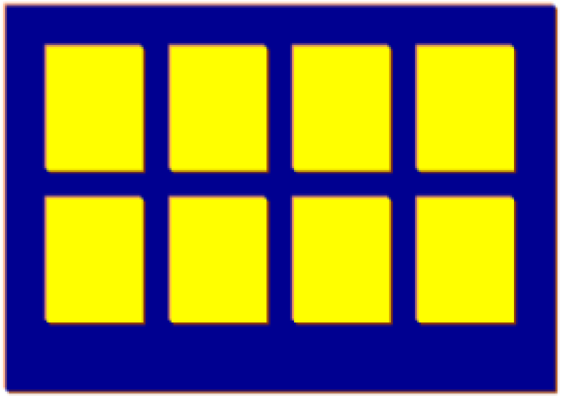
*Performance = Arch efficiency * Peak Bandwidth Capability*

- For compute bound applications

*Performance = Arch efficiency * Peak Compute Capability*

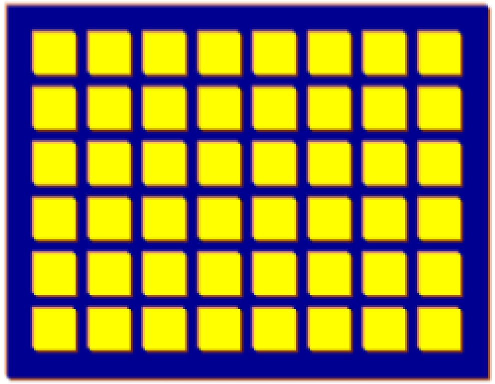
Reasonable Speedup

- Chip A



$Perf_A = Eff_A * Peak_A(\text{Comp or BW})$

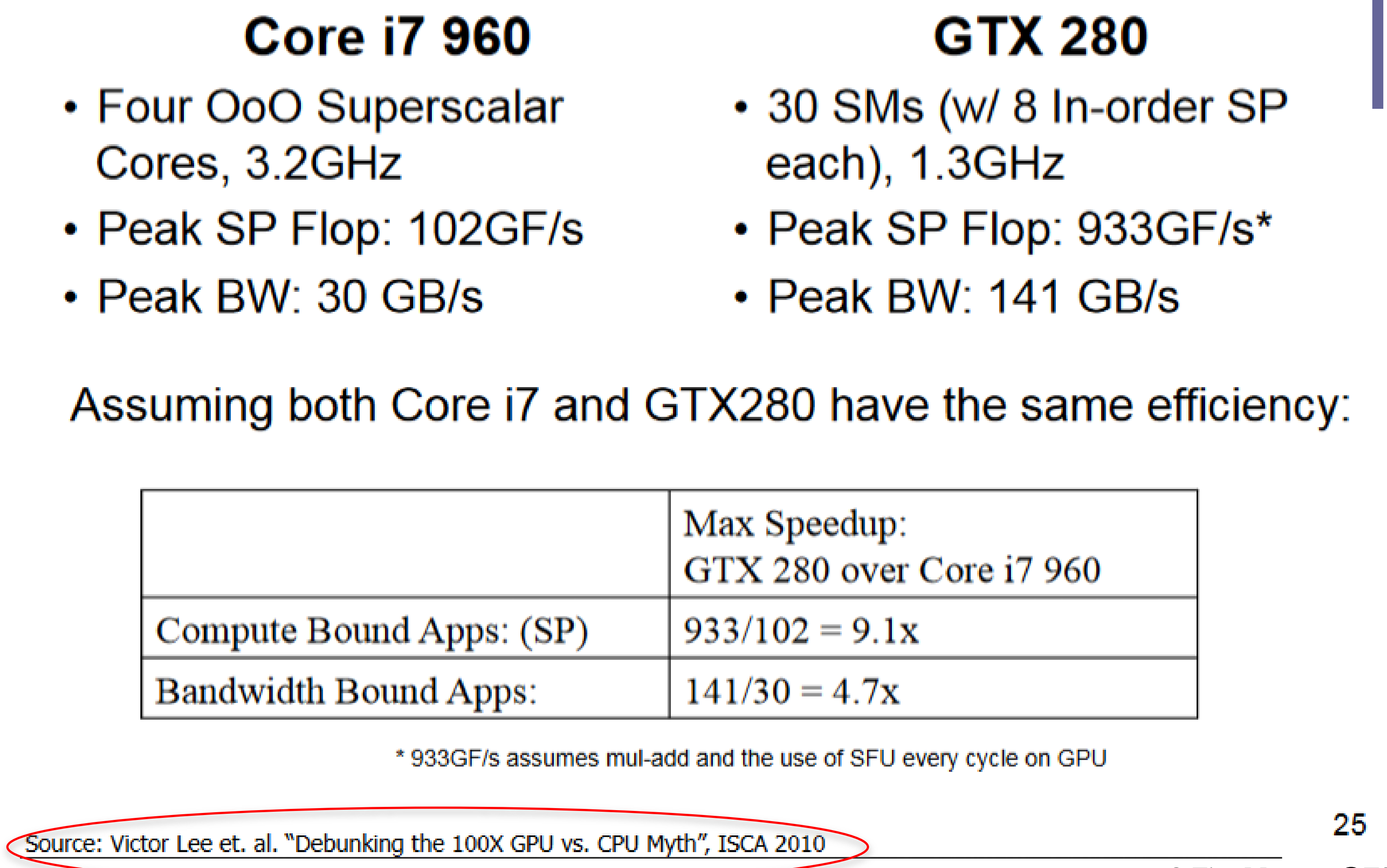
- Chip B



$Perf_B = Eff_B * Peak_B(\text{Comp or BW})$

$$Speedup\frac{B}{A} = \frac{Perf_B}{Perf_A} = \frac{Eff_B}{Eff_A} * \frac{Peak_A(Comp\_or\_BW)}{Peak_B(Comp\_or\_BW)}$$

Reasonable Speedups



Core i7 960

- Four OoO Superscalar Cores, 3.2GHz
- Peak SP Flop: 102GF/s
- Peak BW: 30 GB/s

GTX 280

- 30 SMs (w/ 8 In-order SP each), 1.3GHz
- Peak SP Flop: 933GF/s*
- Peak BW: 141 GB/s

Assuming both Core i7 and GTX280 have the same efficiency:

	Max Speedup: GTX 280 over Core i7 960
Compute Bound Apps: (SP)	933/102 = 9.1x
Bandwidth Bound Apps:	141/30 = 4.7x

* 933GF/s assumes mul-add and the use of SFU every cycle on GPU

Source: Victor Lee et. al. "Debunking the 100X GPU vs. CPU Myth", ISCA 2010

25

A Fair Comparison

- Start with previously best published code / algorithm
- Validate claims by others
- **Optimize BOTH CPU and GPU versions**
- Collect and analysis performance data

https://www.hpcwire.com/2011/12/13/ten_ways_to_fool_the_masses_when_givin
g_performance_results_on_gpus/

Common Mistakes in Comparing CPU and GPU performances

- Compare the latest GPU against an old CPU
- Highly optimized GPU code vs. unoptimized CPU code
- Compare optimized CUDA vs. Matlab or python
- Parallel GPU code vs. serial, unvectorized CPU code
- Ignore the GPU penalty of moving data across the PCI bus from the CPU to the GPU
- **GPUs and other accelerators can be great but be suspicious when people claim speedups of 100+**

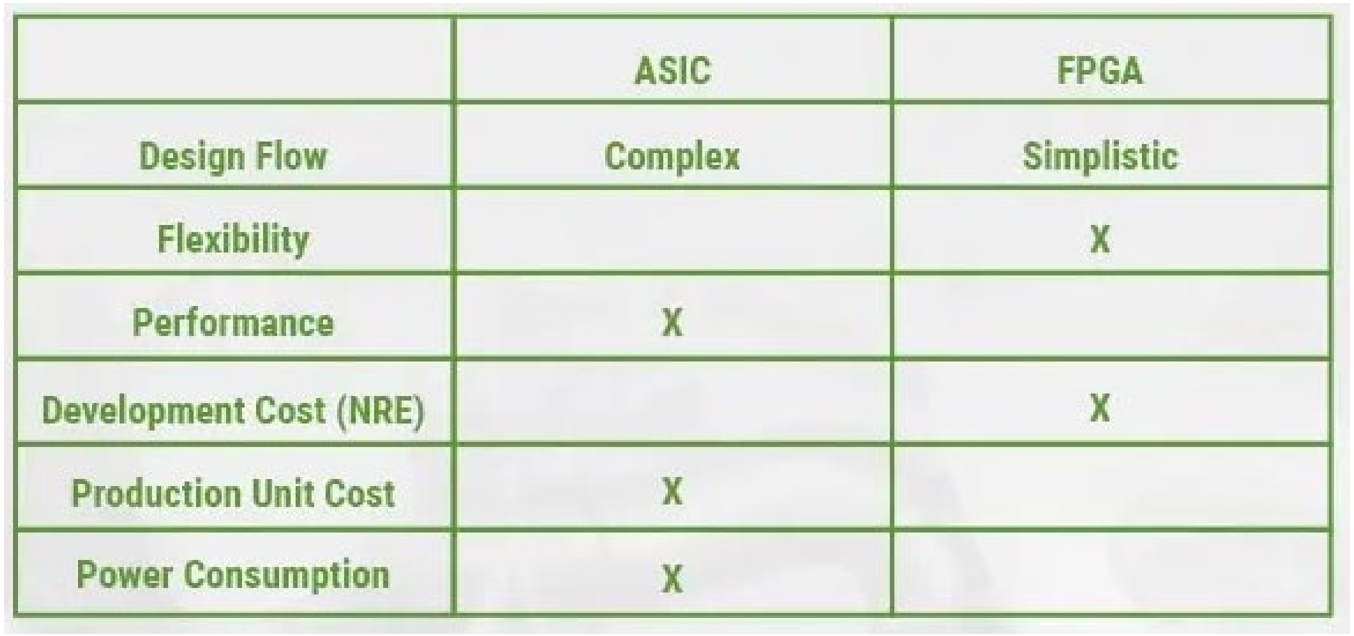
Field Programmable Gate Array

- Field programmable gate arrays (FPGAs) are digital integrated circuits (ICs) that contain configurable (programmable) blocks of logic along with configurable interconnects between these blocks
- Design engineers can configure (program) such devices to perform a tremendous variety of tasks
- Depending from their implementation, some FPGAs may only be programmed a single time (one-time programmable (OP)), while others may be reprogrammed over and over again

+ FPGA vs ASIC

- The major advantage of FPGAs over Application Specific Integrated Circuit (ASIC) is its programmability feature and low time market
  - ASIC takes months to manufacture and is not flexible in terms of functionality
- An ASIC is designed for a specific application while an FPGA is a multipurpose microchip you can reprogram for multiple applications.

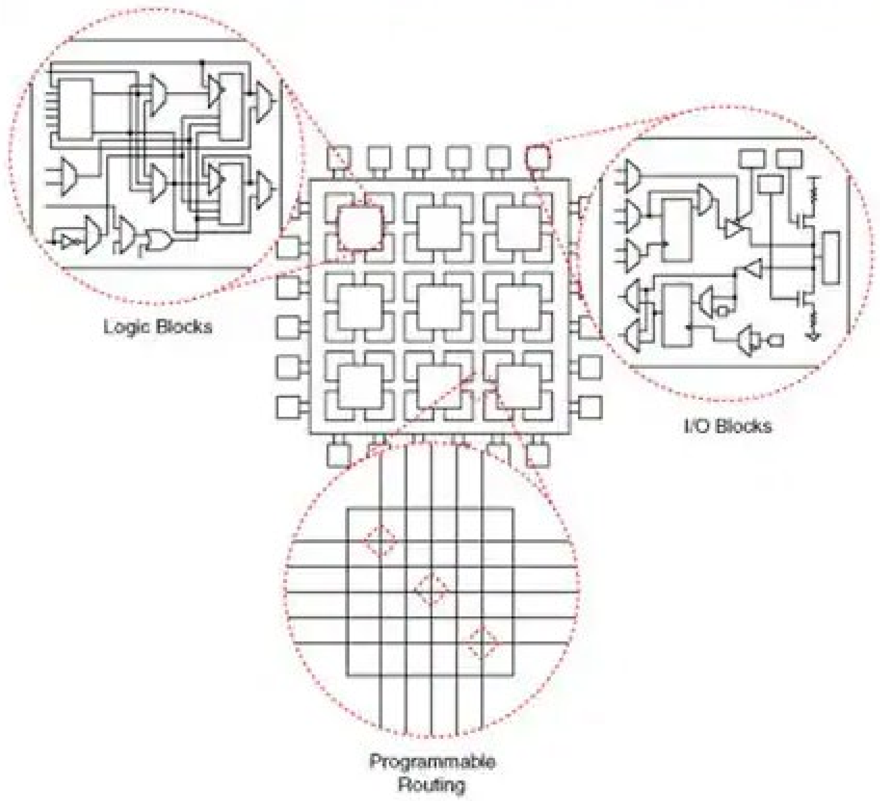
FPGA vs ASIC



	ASIC	FPGA
Design Flow	Complex	Simplistic
Flexibility		X
Performance	X	
Development Cost (NRE)		X
Production Unit Cost	X	
Power Consumption	X	

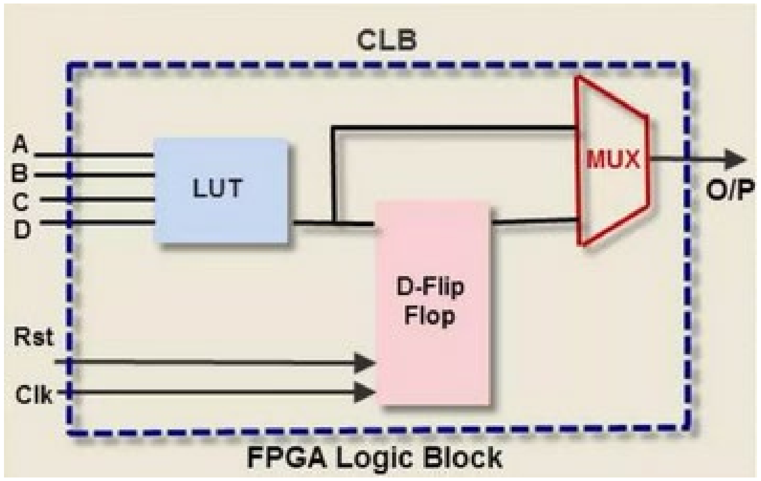
FPGA architecture

- A basic FPGA architecture consists of 3 components:
- thousands of fundamental elements called configurable logic blocks (CLBs)
- CLBs are surrounded by a system of programmable interconnects, that routes signals between CLBs
- Input/output (I/O) blocks interface between the FPGA and external devices



Field Programmable Gate Array

- An individual CLB is made up of several logic blocks.
  - A lookup table (LUT) stores a predefined list of logic outputs for any combination of inputs and implements the combinational logical functions
  - MUX is used for selection logic
  - D-Flip Flop stores the output of the LUT



Field Programmable Gate Array

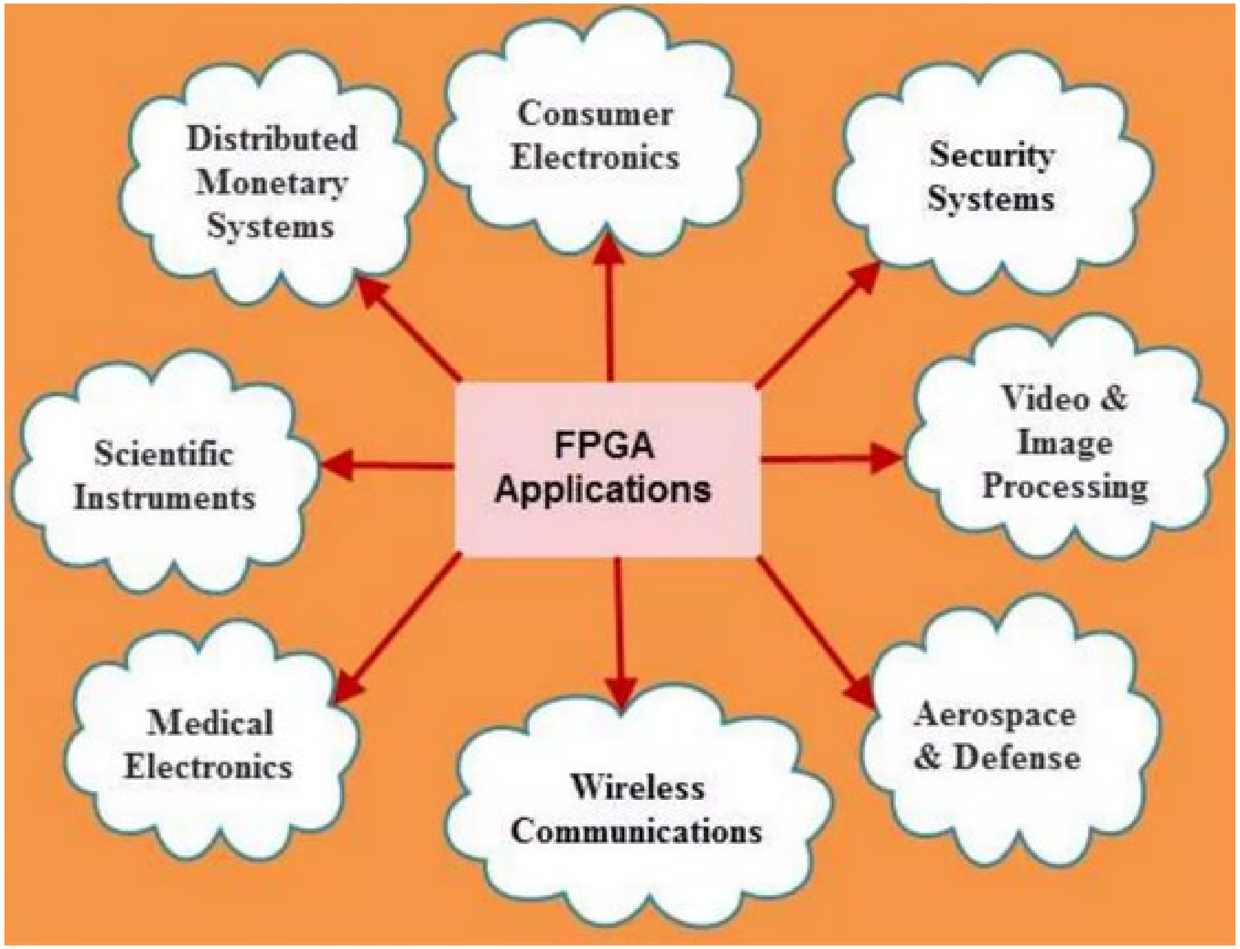
- Programmable routing
  - Establish the connection between configurable logic blocks and Input/Output blocks to complete a user-defined design unit
  - It consists of multiplexers, pass transistors and tri-state buffers
    - Used in a logic cluster to connect the logic elements

Field Programmable Gate Array

- I/O blocks
  - Used to interface the logic blocks and routing architecture to the external components
  - The I/O pad and the surrounding logic circuit form as an I/O cell
  - These cells consume a large portion of the FPGA’s area

FPGA Applications

- Application domain



FPGA Applications

- Application domain
  - Emulation of entire large HW systems via the use of many interconnected FPGAs
  - ASIC prototyping or SoC prototyping
    - HW verification and early SW development
  - Parallel computing to meet requirements in machine learning, computer vision, etc…