# ECE 432/532
# Programming for Parallel Processors

Dr. Iraklis Anagnostopoulos

# Welcome

- Welcome to 432/532 course “Programming for Parallel Processors”

- I wish you all good luck!!

# Logistics

- This course will focus on parallel programming:
  - What is parallel programming?
  - Why we need parallel programming
  - How can we execute parallel code?
  - Models for parallel programming
  - Parallel computer architectures
  - Parallel algorithms modeling and analysis of parallel programs and systems
  - and more…

# Logistics - Prerequisites

- Programming in C, C++, or similar
- Basics of data structures
- Basics of machine architecture
- Basic Unix/Linux knowledge

- See me if you have any concerns

# Logistics

- Instructor:
  - Dr. Iraklis Anagnostopoulos, iraklis.anagno@siu.edu
- TA
  - Sheheeda Mariam Manakkadu, sheheeda@siu.edu
- Lectures:
  - Monday, Wednesday & Friday, 4:00 pm – 4:50 pm
  - Class E-136
  - Unix lab will also be used (announcement will follow)
- Office hours:
  - Wednesday & Friday, 10:00 am – 1:00 pm

# Logistics

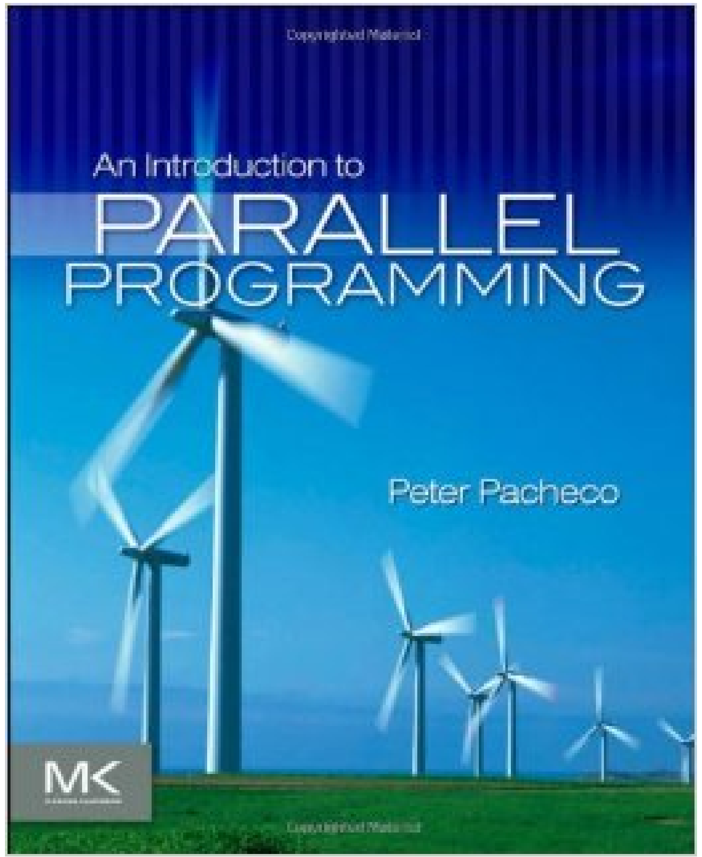
- Homeworks - 30%
- Midterm - 30% of the final grade
- Final exam - 40% of the final grade

- Final assignment of grades will be based on a curve.

# Main bibliography

Peter S. Pacheco,

“An Introduction to Parallel Programming

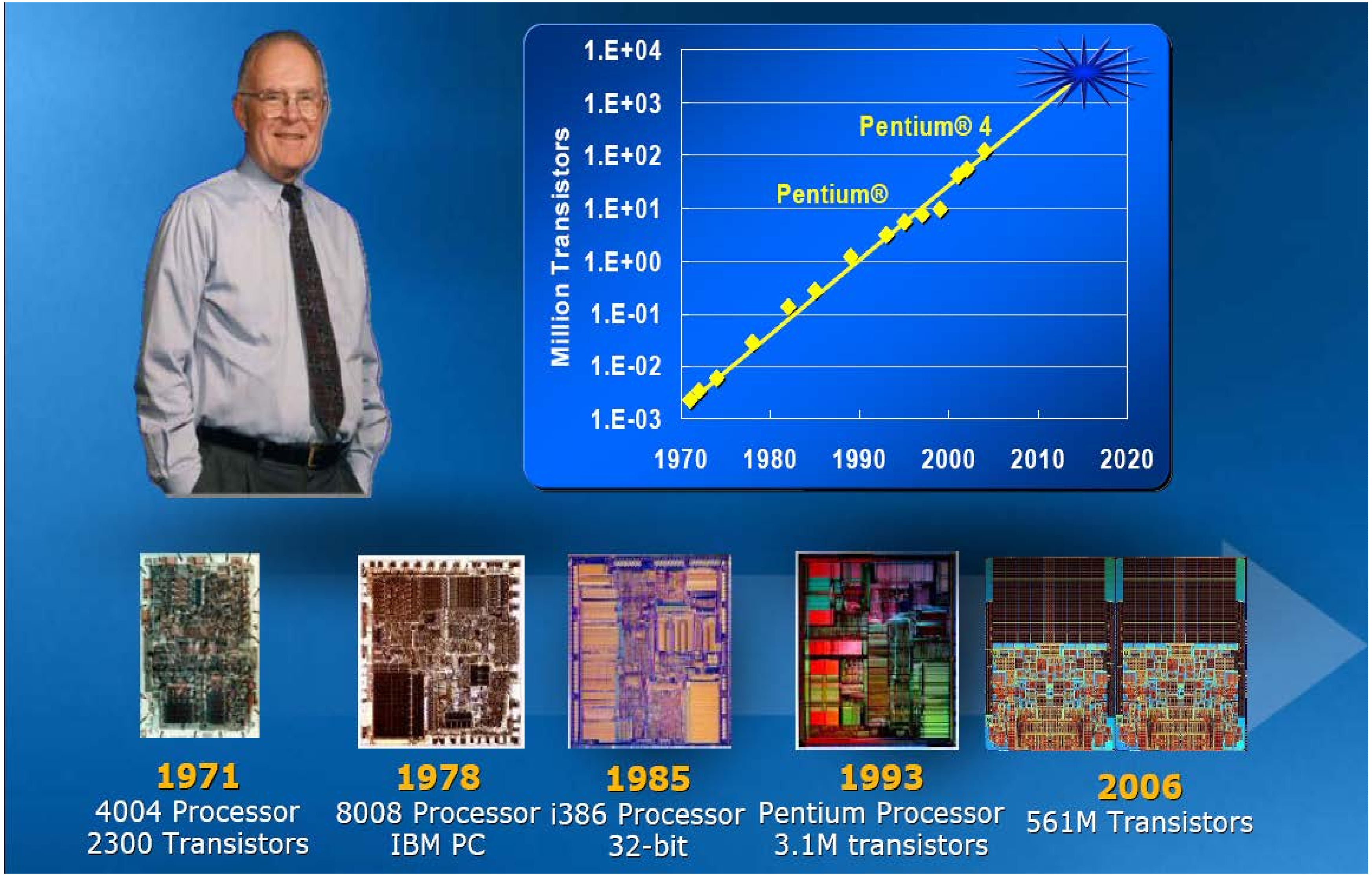


# So, let’s start

# Technological trends: From Moore's law…

- The number of transistors on integrated circuits doubles approximately **every two years** (Moore, 1965)
- More good news! **transistors become faster as well**!
  - CPU speed doubles every 18 months
- What to do with so many, fast transistors?
  - Instruction Level Parallelism! (ILP)
    - Deeper pipelines
    - Faster clock speeds
    - Better branch predictors
    - Out of order execution
    - Superscalar
  - Larger caches – More caching levels
  - Vector units
- Faster processing cores at no programming cost

# Technological trends: From Moore's law…



**1971**
4004 Processor
2300 Transistors

**1978**
8008 Processor
IBM PC

**1985**
i386 Processor
32-bit

**1993**
Pentium Processor
3.1M transistors

**2006**
561M Transistors

# Technological trends: … to Dennard Scaling

- Wikipedia:

*MOSFETs continue to function as voltage-controlled switches while all key figures of merit such as layout density, operating speed, and energy efficiency improve – provided geometric dimensions, voltages, and doping concentrations are consistently scaled to maintain the same electric field (Dennard 1974)*

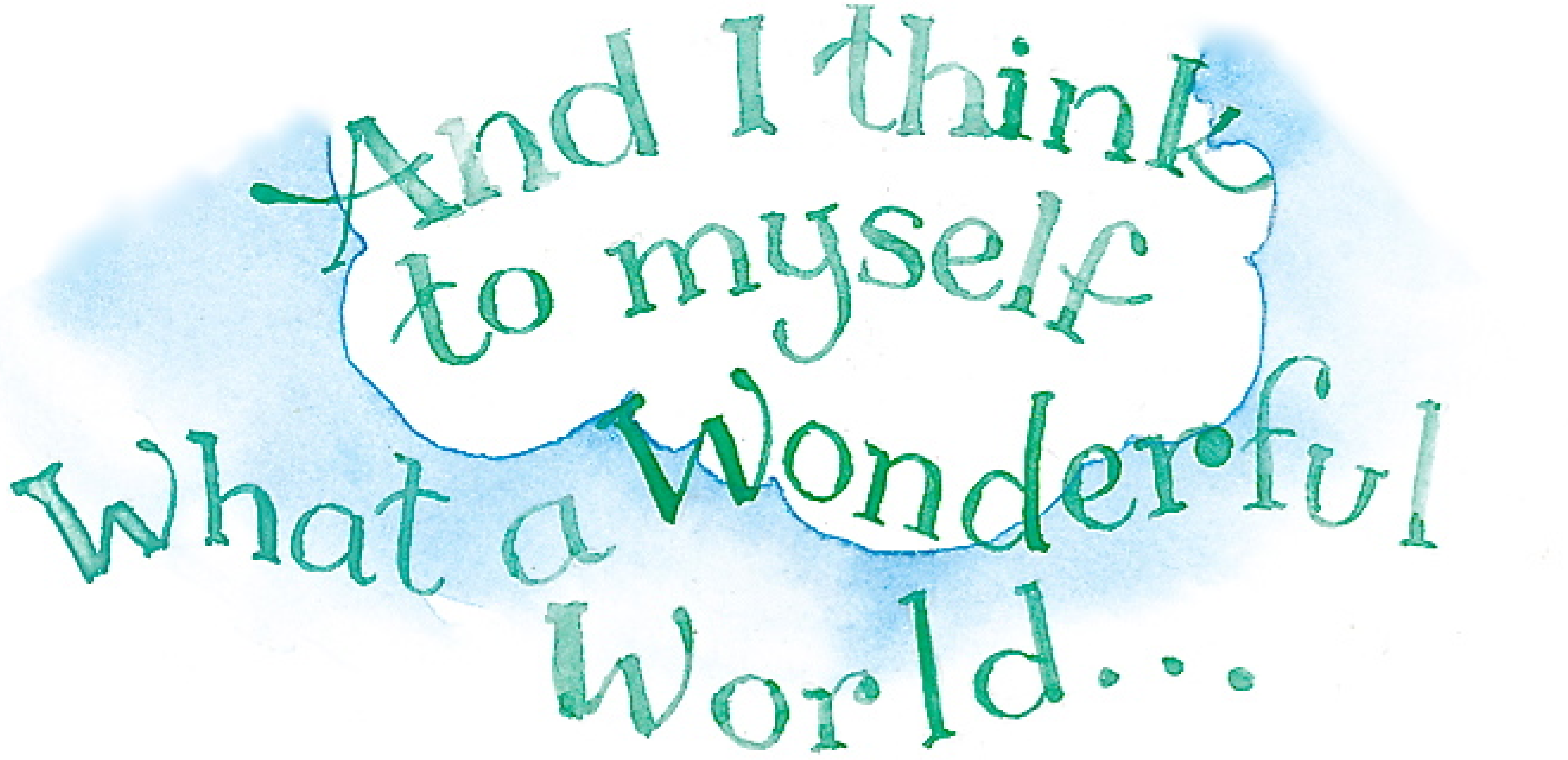
# Technological trends: … to Dennard Scaling

- The dynamic (switching) power consumption of CMOS circuits is proportional to frequency
- Historically, the transistor power reduction allowed manufacturers to drastically raise clock frequencies without significantly increasing overall circuit power consumption.

# Technological trends: … to Dennard Scaling

- $P = CV^2f$ (C = count, V = voltage, f = frequency)
  - Increase in device count
  - Higher operating frequencies
  - Lower supply voltages
  - **Constant power / chip!!!**

And I think
to myself
What a Wonderful
World...



**Is it?**

# What a wonderful world…

- In ~2004 we hit the ILP wall
  - Transistors could not be utilized to increase serial performance
  - Logic became too complex
  - Performance attained was very low compared to power consumption

# What a wonderful world…

- In ~2004 we hit the ILP wall
  - Transistors could not be utilized to increase serial performance
  - Logic became too complex
  - Performance attained was very low compared to power consumption

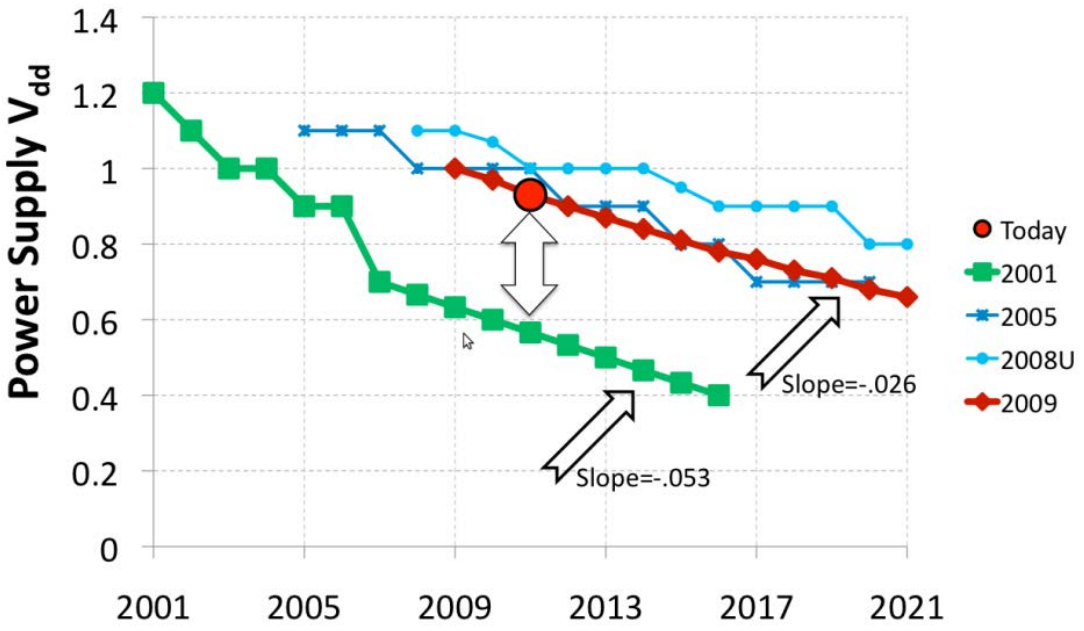
# What a wonderful world…

- In ~2004 we hit the ILP wall
    - Transistors could not be utilized to increase serial performance
    - Logic became too complex
    - Performance attained was very low compared to power consumption

- Solution:
    - Multicore CPUs!
    - But.. The free lunch is over… welcome to the jungle!
      (http://herbsutter.com/welcome-to-the-jungle/)
      (http://www.gotw.ca/publications/concurrency-ddj.htm)
    - We need parallel software

# The end of Dennard scaling?

- Transistors are becoming too small
- A lot of energy is lost in leakage
- Voltage has not dropped significantly during the last few years



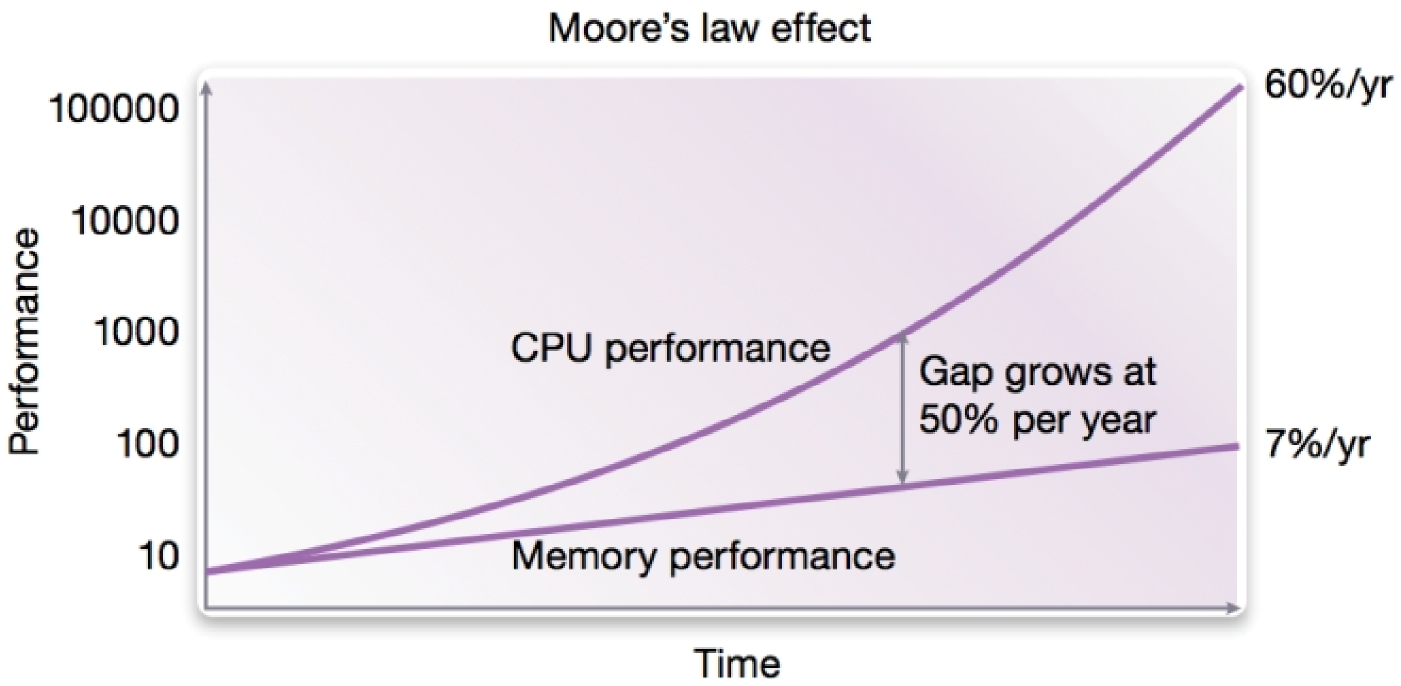
# The power wall…

- If P = $CV^2f$
  - C can still increase as predicted my Moore’s law
    - transistors get shorter
    - number of cores increase
  - V cannot drop drastically
  - We need to keep f low
  - But still P may take off….
- Dark silicon? (functionality is there but we cannot switch it on)

# Ok.. Is that all?

# Ok.. Is that all?

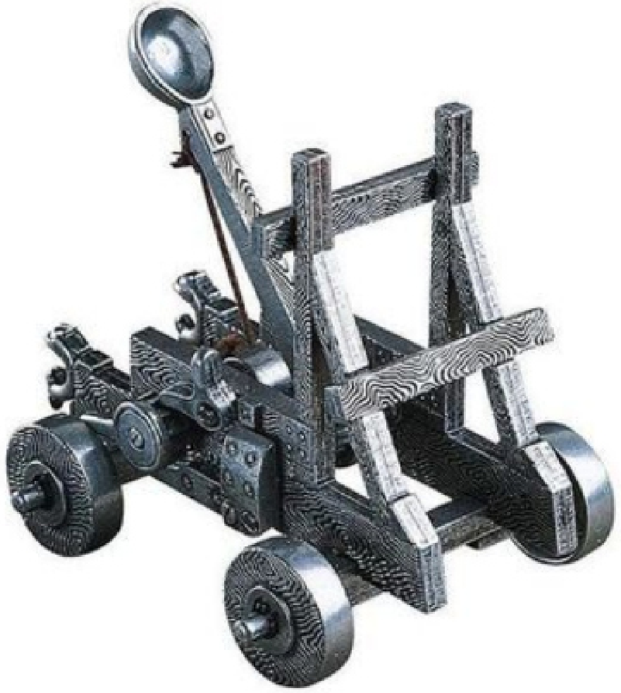
- No ☹
- CPU to memory gap



- **The memory wall**: CPUs are much faster than memory and applications may starve waiting for data from main memory…

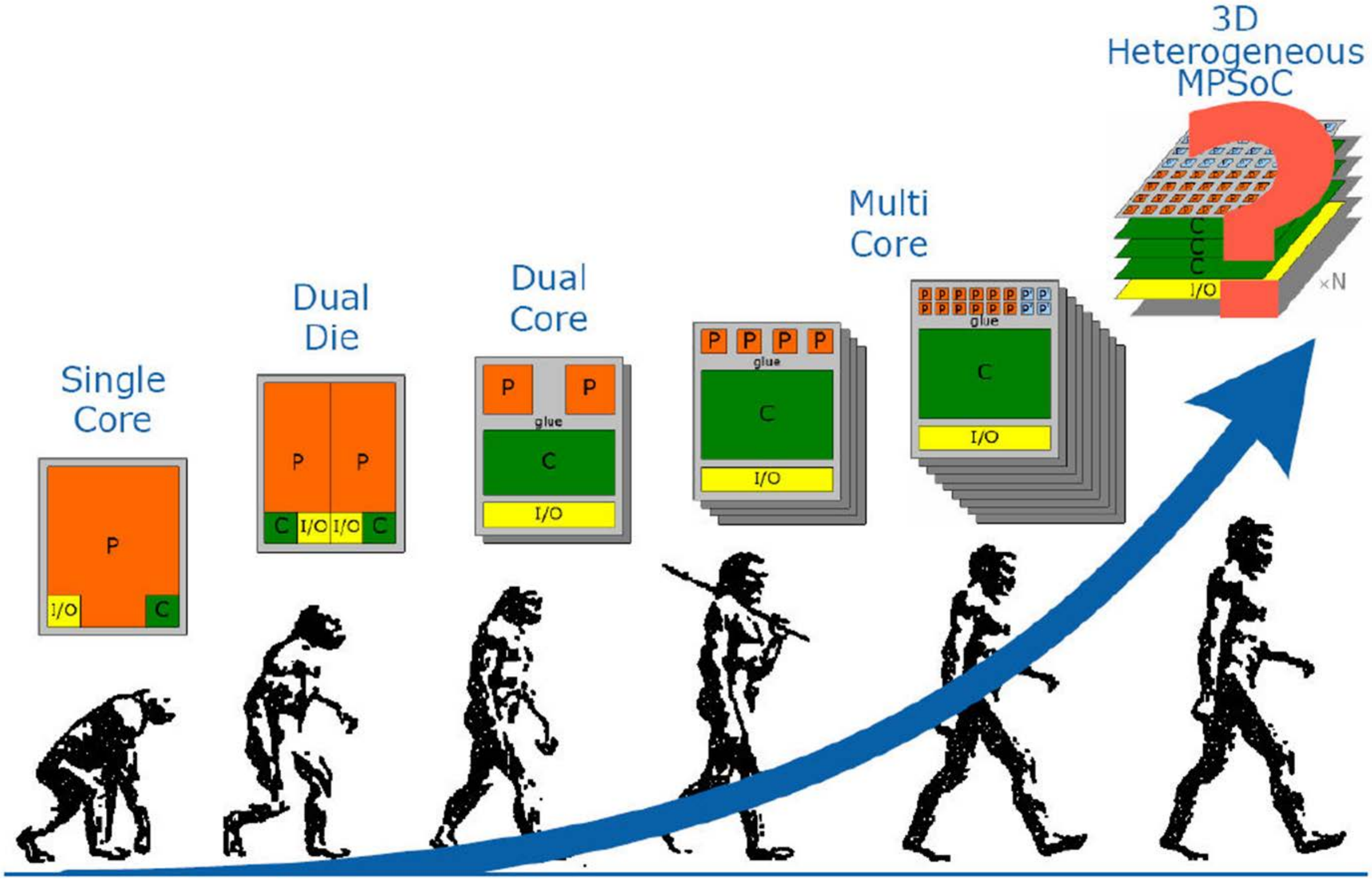
# Two ways ahead…

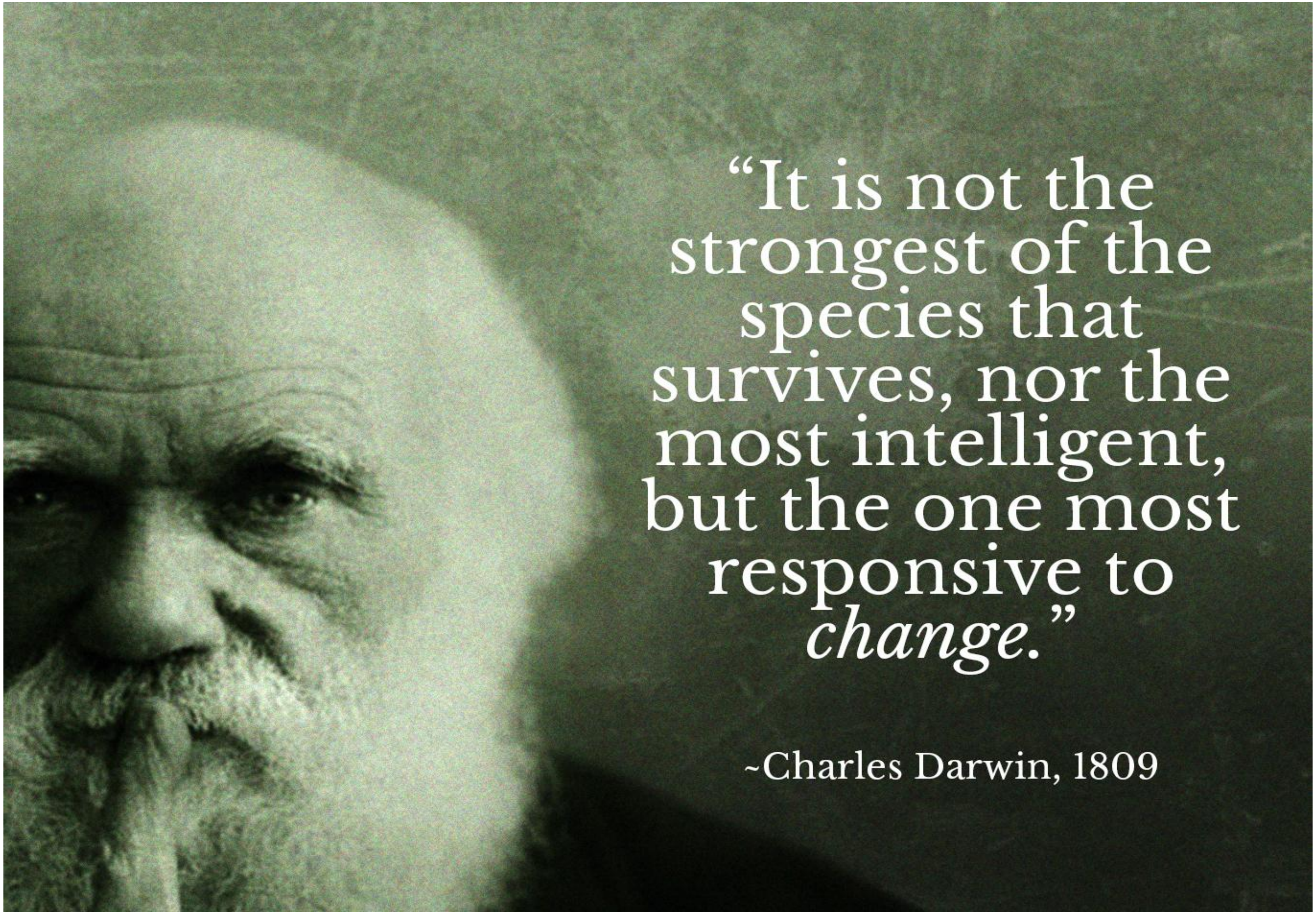
- **The catapult way**: Advances through technological breakthrough



- **The parkour way**: Redesign software and algorithms

# It’s the evolution after all





"It is not the strongest of the species that survives, nor the most intelligent, but the one most responsive to *change*."

~Charles Darwin, 1809

# Motivations for Parallel Computing

- Technology push
- Application pull

# Microprocessor Architecture (Mid 90’s)

- Superscalar (SS) designs were the state of the art
  - multiple functional units (e.g., int, float, branch, load/store)
  - multiple instruction issue
  - dynamic scheduling: HW tracks instruction dependencies
  - speculative execution: look past predicted branches
  - non-blocking caches: multiple outstanding memory operations
- Apparent path to higher performance?
  - wider instruction issue
  - support for more speculation

# Microprocessor Architecture (Mid 90’s)

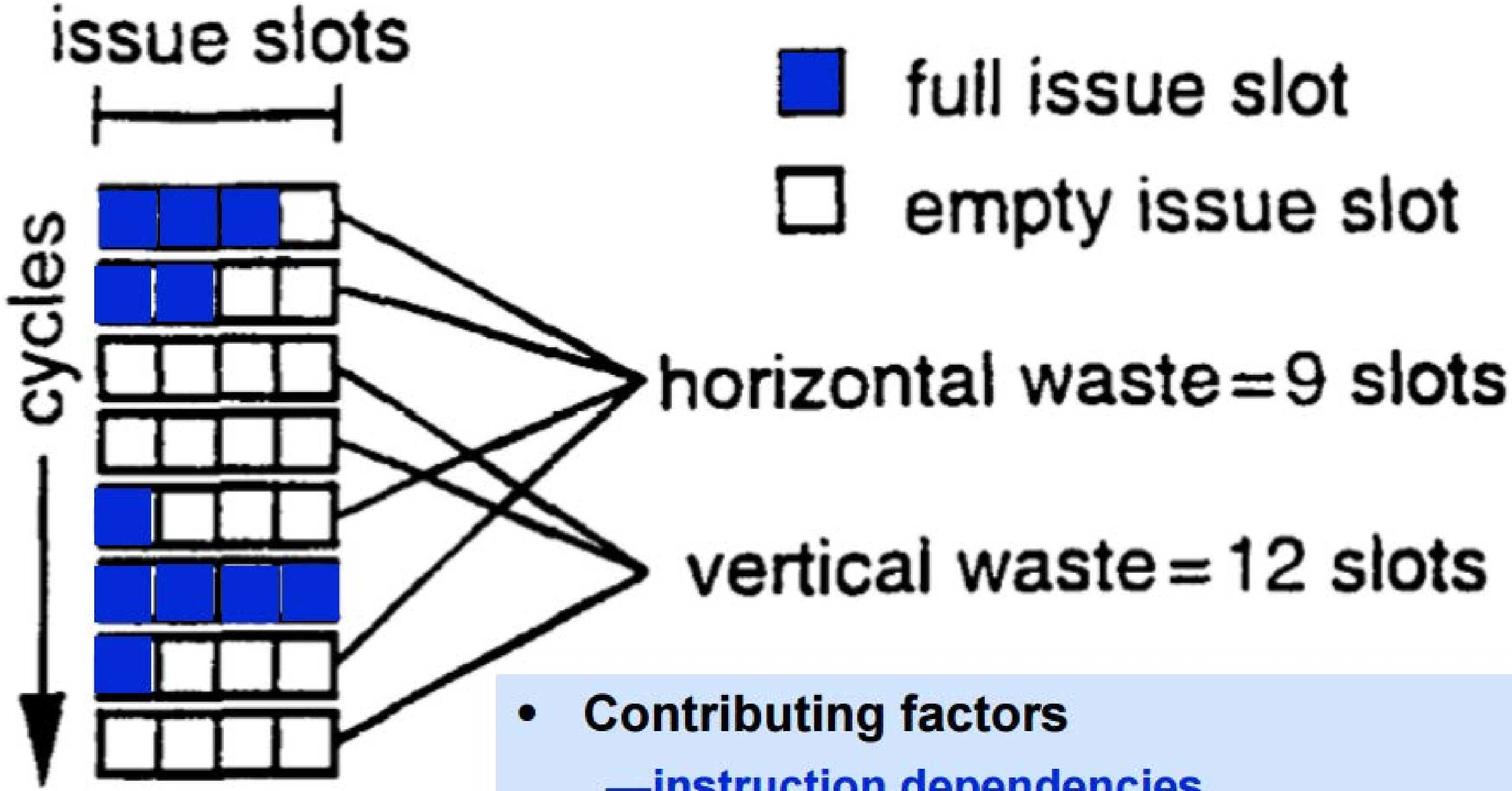
Increasing issue width provides diminishing returns

Two factors

- Fundamental circuit limitations
    - delays ⇑ as issue queues ⇑ and multi-port register files ⇑
    - increasing delays limit performance returns from wider issue
- Limited amount of instruction-level parallelism
    - inefficient for codes with difficult-to-predict branches

# Instruction-level Parallelism Concerns

## Issue Waste



- **Contributing factors**
  - **—instruction dependencies**
  - **—long-latency operations within a thread**

# Some Sources of Wasted Issue Slots

- Translation Lookaside Buffer (TLB) miss
- I cache miss
- D cache miss
- Load delays (L1 hits)

**Memory**

- Branch misprediction

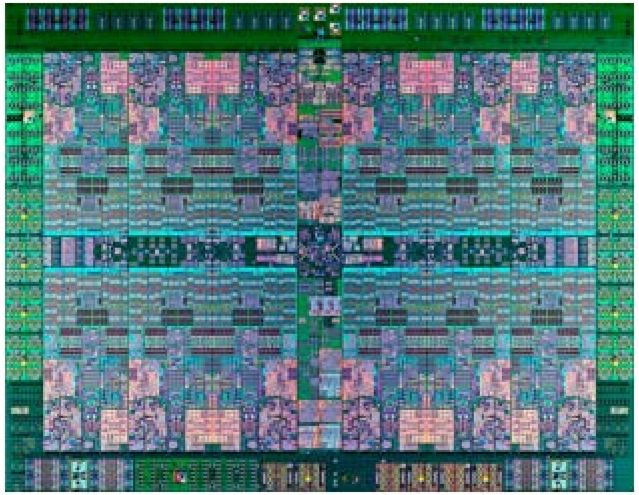
**Control Flow**

- Instruction dependences
- Memory conflict

**Instruction Stream**

# Recent Multicore Processors

- **2016: Intel Knight's Landing**
  —72 cores; 4-way SMT; 16GB (on pkg)
- **2015: Oracle SPARC M7**
  —32 cores; 8-way fine-grain MT; 64MB cache
- **Fall 14: Intel Haswell**
  —18 cores; 2-way SMT; 45MB cache
- **June 14: IBM Power8**
  —12 cores; 8-way SMT; 96MB cache
- **Sept 13: SPARC M6**
  —12 cores; 8-way fine-grain MT; 48MB cache
- **May 12: AMD Trinity**
  —4 CPU cores; 384 graphics cores
- **Q2 13: Intel Knight's Corner (coprocessor)**
  —61 cores; 2-way SMT; 16MB cache
- **Feb 12: Blue Gene/Q**
  —16+1+1 cores; 4-way SMT; 32MB cache



Simultaneous multithreading (SMT)

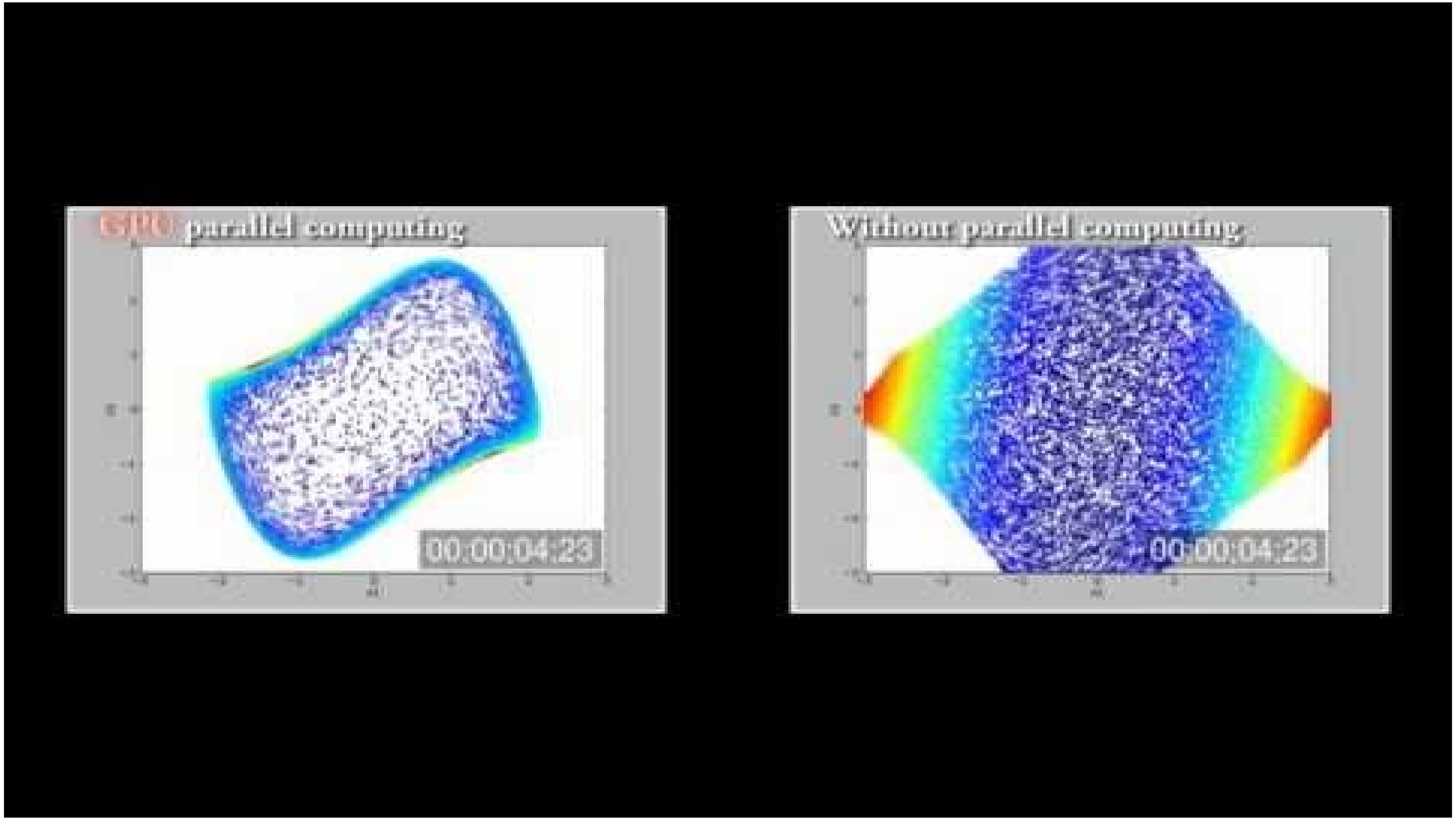
# Application Pull

- Complex problems require computation on large-scale data
- Sufficient performance available only through massive parallelism

# The Need for Speed: Complex Problems

- Science
  - understanding matter from elementary particles to cosmology
  - storm forecasting and climate prediction
  - understanding biochemical processes of living organisms
- Engineering
  - combustion and engine design
  - computational fluid dynamics and airplane design
  - earthquake and structural modeling
  - pollution modeling and remediation planning
  - molecular nanotechnology
- Business
  - computational finance
  - high frequency trading
  - information retrieval
  - data mining “big data”

# The Need for Speed: Complex Problems



# Challenges of Explicit Parallelism

- Algorithm development is harder
  - complexity of specifying and coordinating concurrent activities
- Software development is much harder
  - lack of standardized & effective development tools and programming models
  - subtle program errors: race conditions
- Rapid pace of change in computer system architecture
  - a great parallel algorithm for one machine may not be suitable for another
  - example: homogeneous multicore processors vs. GPUs

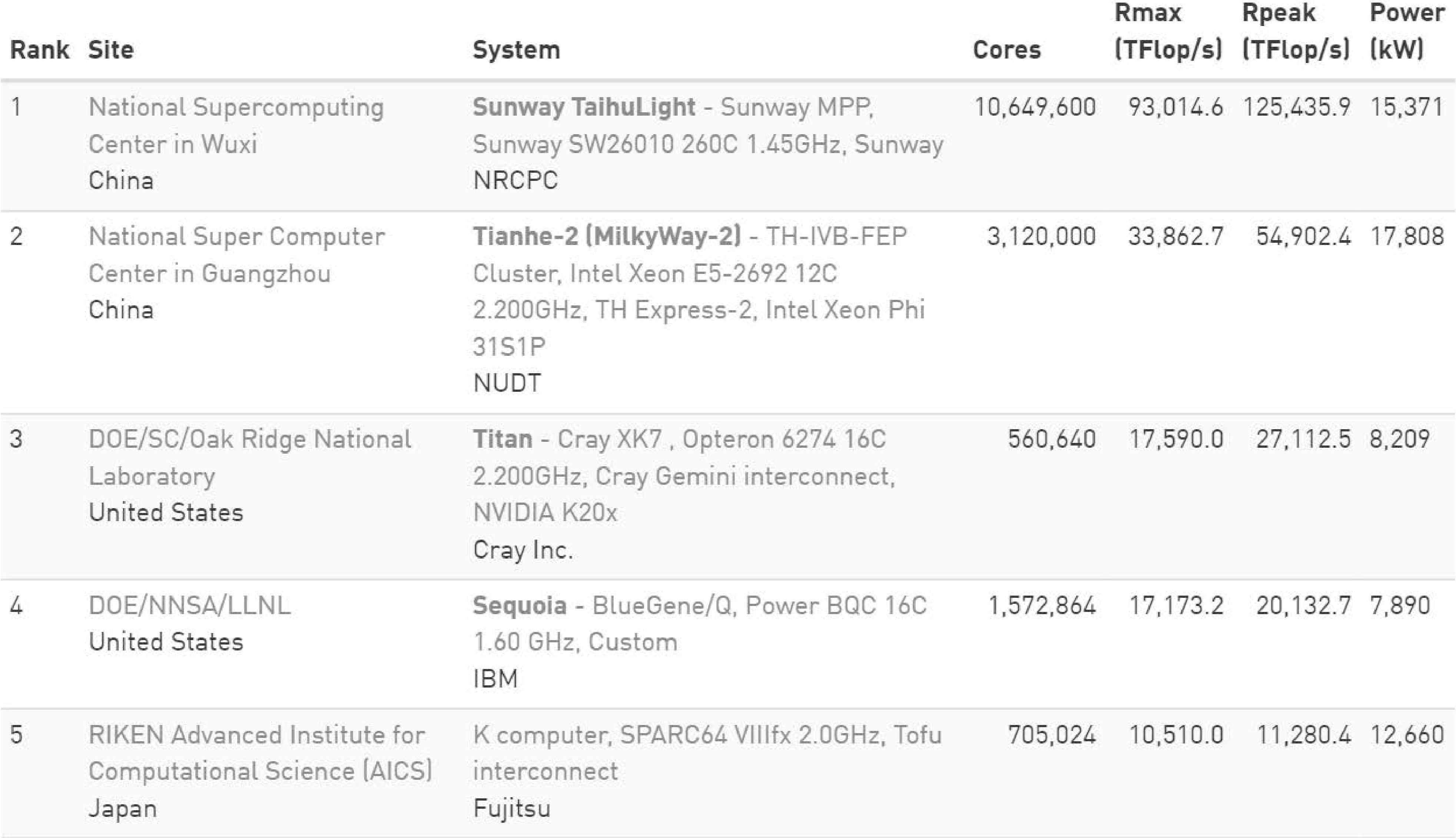
# Parallel Hardware in the Large

# Super computers

- For decades: parallel computing = supercomputing
  - Cooperation of multiple processors
- Super computers
  - Computers with thousand computing elements.

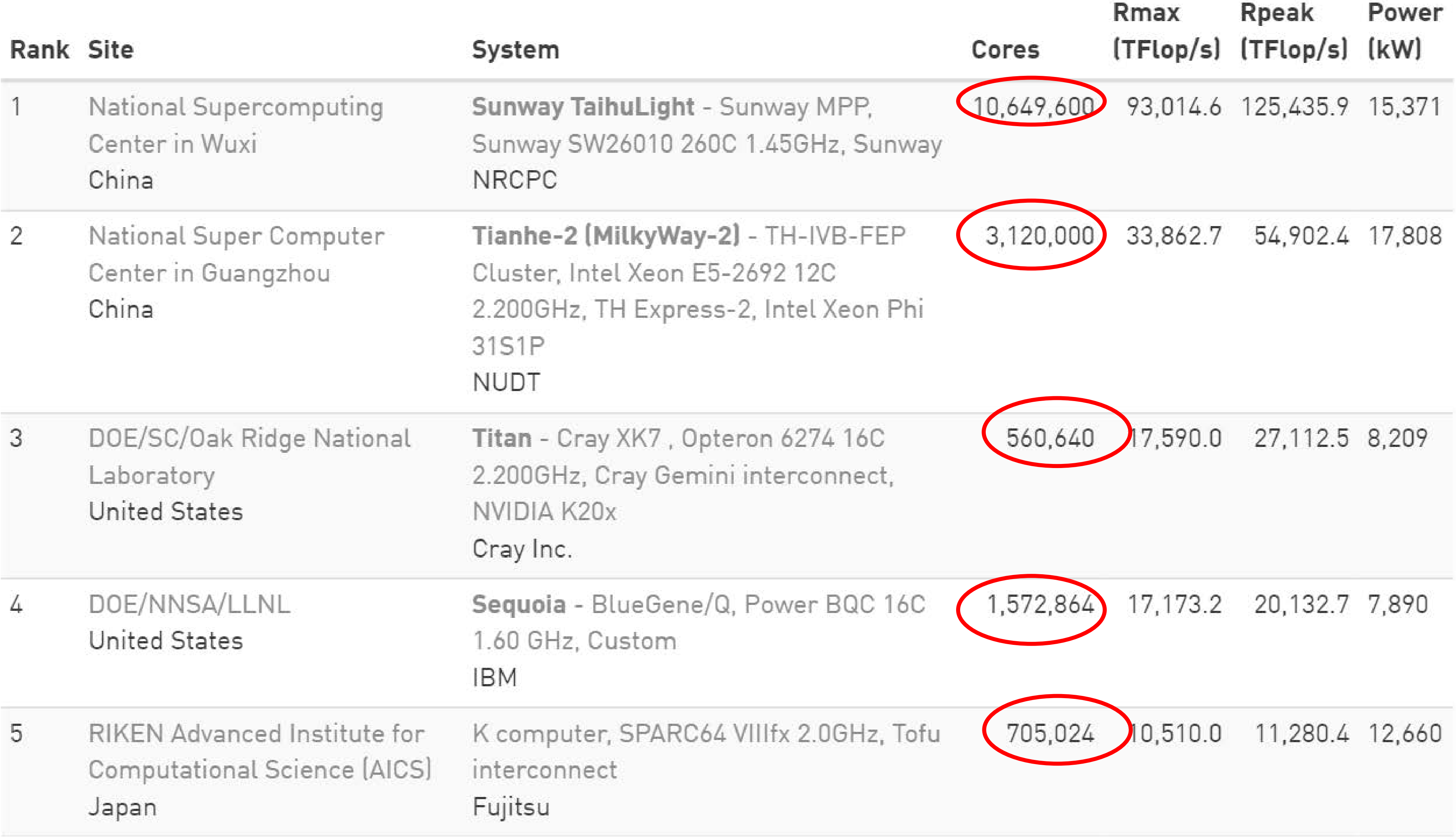
**https://www.top500.org/**

# Top500 -JUNE 2016



| Rank | Site | System | Cores | Rmax<br>(TFlop/s) | Rpeak<br>(TFlop/s) | Power<br>(kW) |
|---|---|---|---|---|---|---|
| 1 | National Supercomputing Center in Wuxi<br>China | **Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway<br>NRCPC | 10,649,600 | 93,014.6 | 125,435.9 | 15,371 |
| 2 | National Super Computer Center in Guangzhou<br>China | **Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P<br>NUDT | 3,120,000 | 33,862.7 | 54,902.4 | 17,808 |
| 3 | DOE/SC/Oak Ridge National Laboratory<br>United States | **Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x<br>Cray Inc. | 560,640 | 17,590.0 | 27,112.5 | 8,209 |
| 4 | DOE/NNSA/LLNL<br>United States | **Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom<br>IBM | 1,572,864 | 17,173.2 | 20,132.7 | 7,890 |
| 5 | RIKEN Advanced Institute for Computational Science (AICS)<br>Japan | K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect<br>Fujitsu | 705,024 | 10,510.0 | 11,280.4 | 12,660 |

# Top500 -JUNE 2016



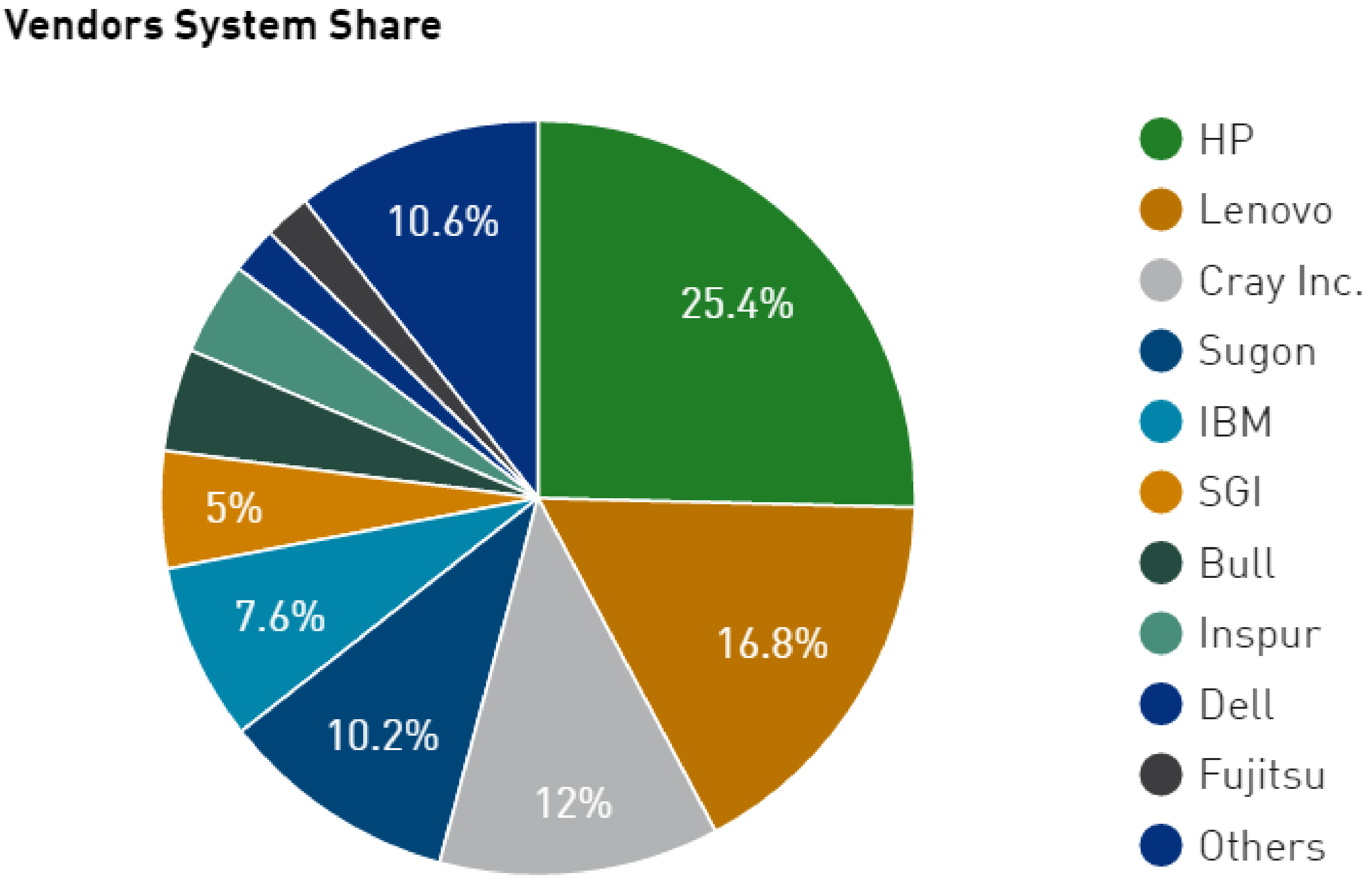
| Rank | Site | System | Cores | Rmax<br>(TFlop/s) | Rpeak<br>(TFlop/s) | Power<br>(kW) |
|---|---|---|---|---|---|---|
| 1 | National Supercomputing Center in Wuxi<br>China | **Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway<br>NRCPC | 10,649,600 | 93,014.6 | 125,435.9 | 15,371 |
| 2 | National Super Computer Center in Guangzhou<br>China | **Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P<br>NUDT | 3,120,000 | 33,862.7 | 54,902.4 | 17,808 |
| 3 | DOE/SC/Oak Ridge National Laboratory<br>United States | **Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x<br>Cray Inc. | 560,640 | 17,590.0 | 27,112.5 | 8,209 |
| 4 | DOE/NNSA/LLNL<br>United States | **Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom<br>IBM | 1,572,864 | 17,173.2 | 20,132.7 | 7,890 |
| 5 | RIKEN Advanced Institute for Computational Science (AICS)<br>Japan | K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect<br>Fujitsu | 705,024 | 10,510.0 | 11,280.4 | 12,660 |

Sunway TaihuLight: National Supercomputing Center in Wuxi

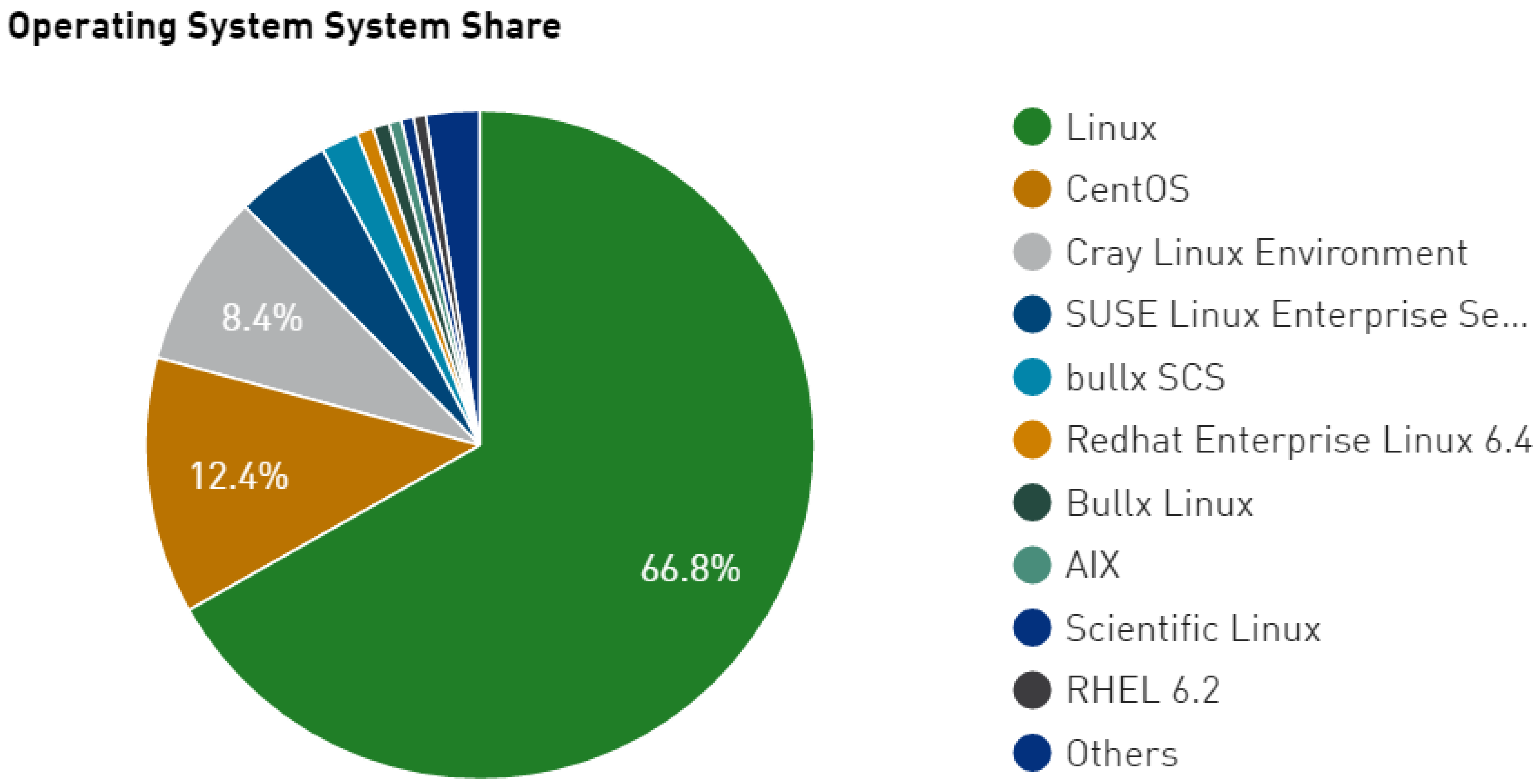
Tianhe-2 (MilkyWay-2) : National University of Defense Technology

Titan: Oak Ridge National Laboratory

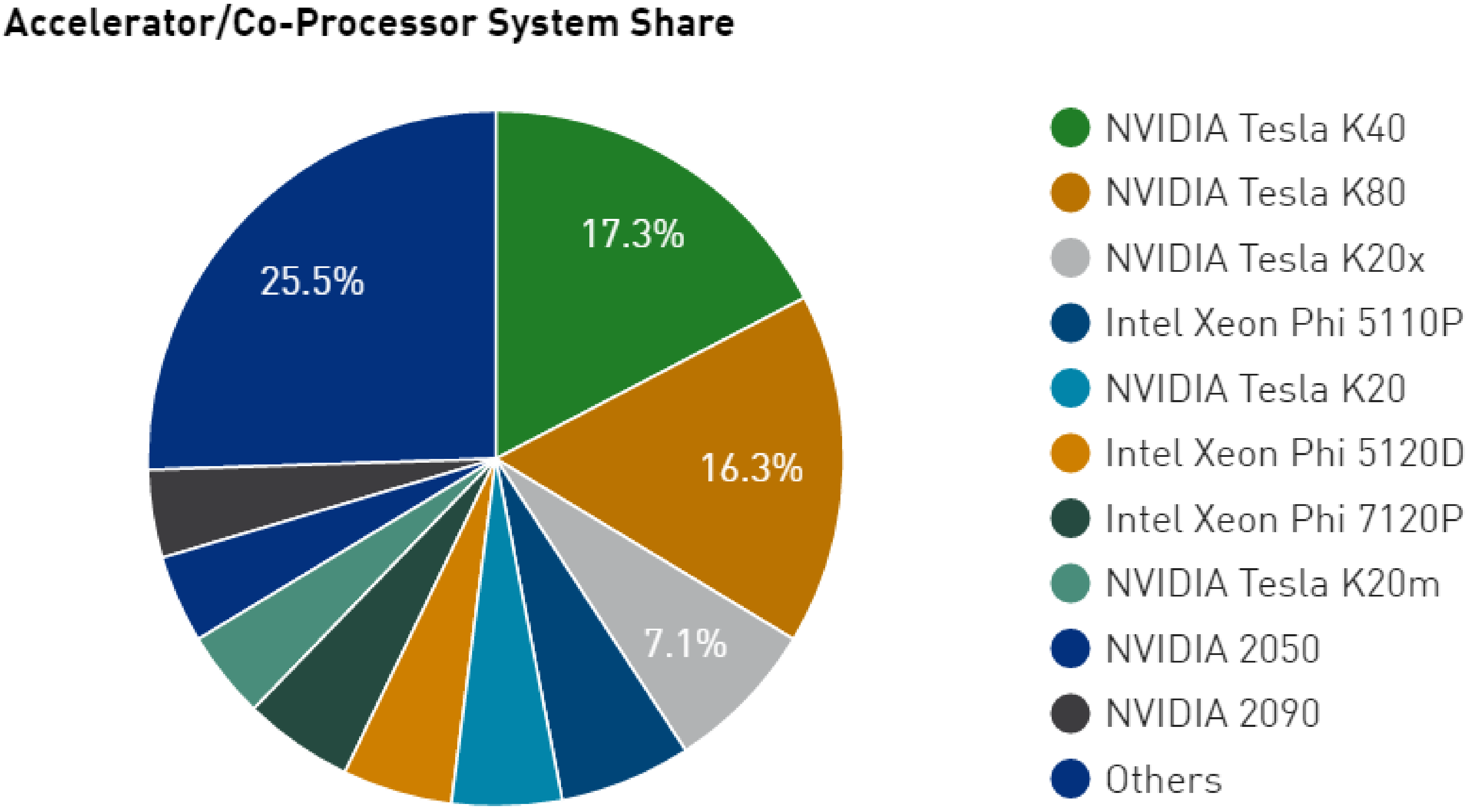
# Top500 - statistics



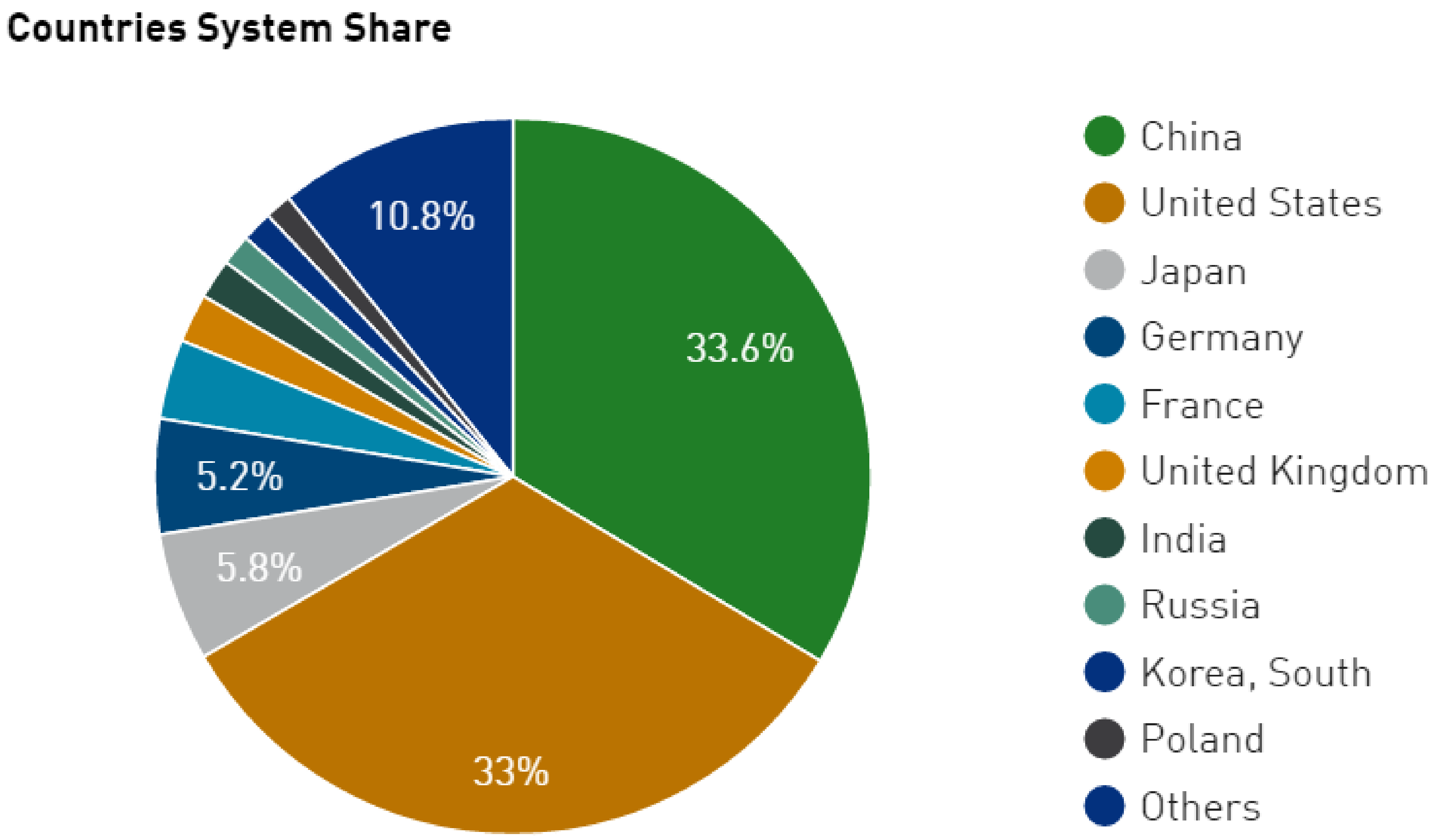
# Top500 - statistics



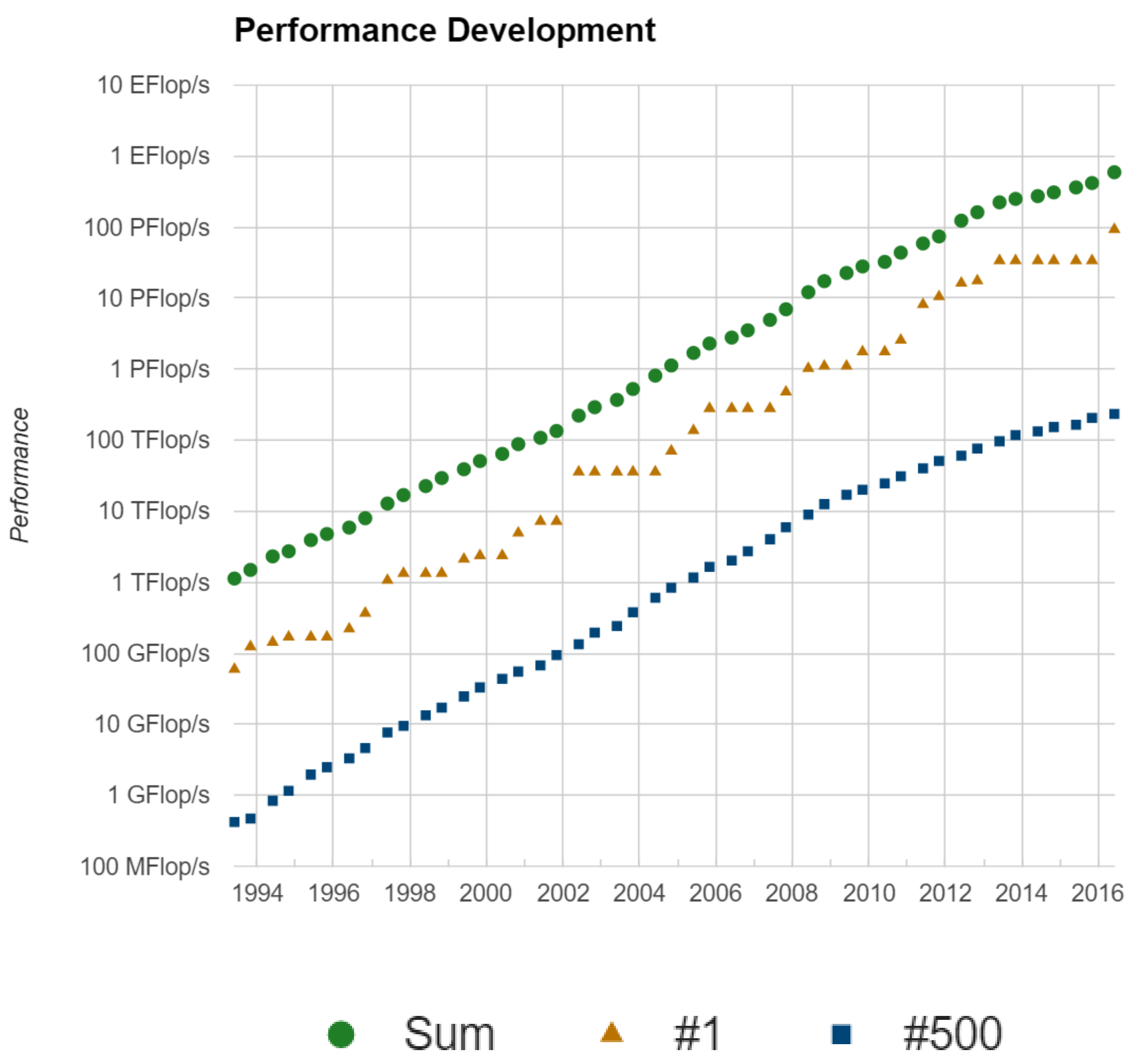
# Top500 - statistics

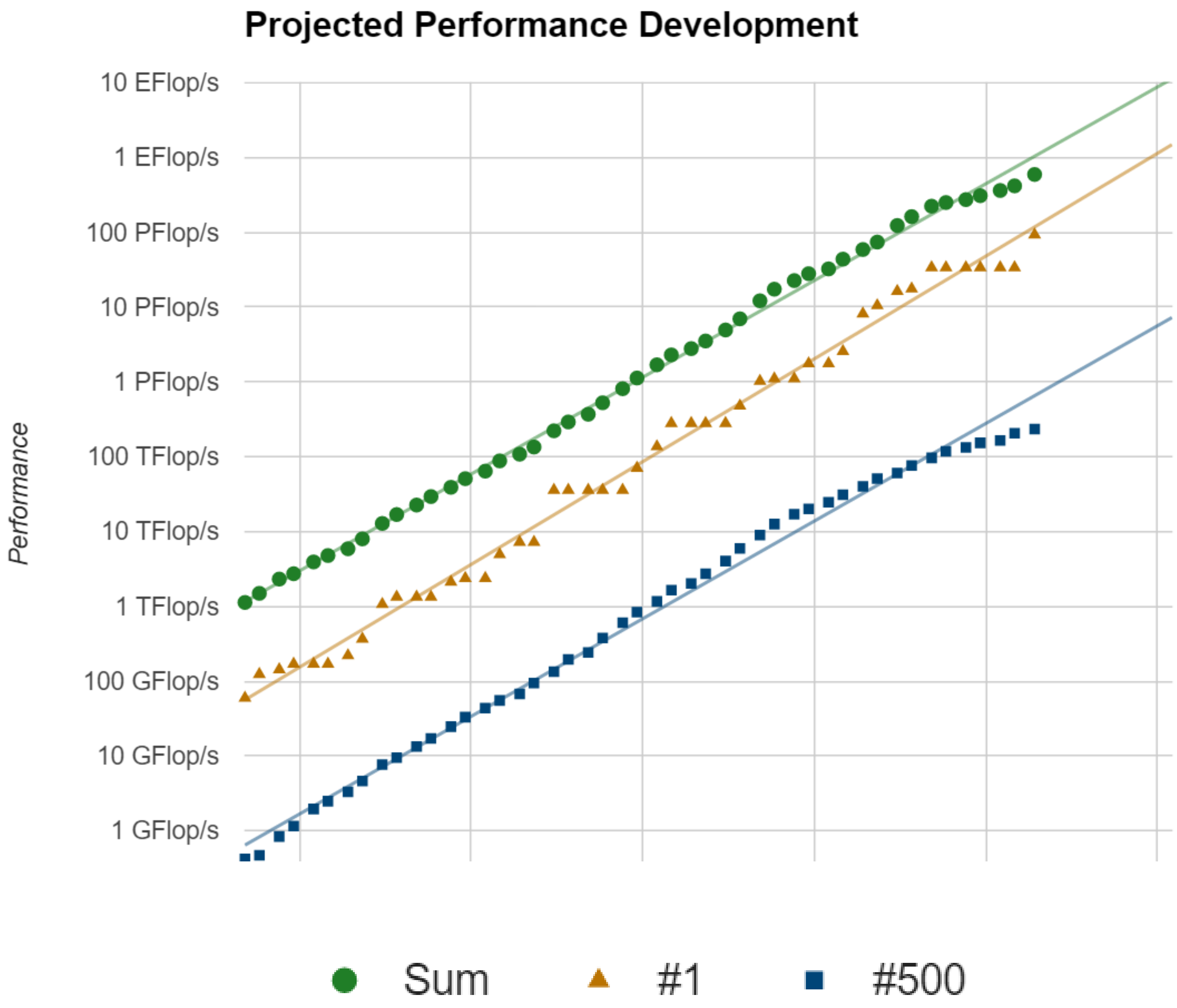


# Top500 - statistics



# Top500 - statistics


Performance Development
10 EFlop/s
1 EFlop/s
100 PFlop/s
10 PFlop/s
1 PFlop/s
100 TFlop/s
10 TFlop/s
1 TFlop/s
100 GFlop/s
10 GFlop/s
1 GFlop/s
100 MFlop/s
Performance
● Sum
▲ #1
■ #500


Projected Performance Development
10 EFlop/s
1 EFlop/s
100 PFlop/s
10 PFlop/s
1 PFlop/s
100 TFlop/s
10 TFlop/s
1 TFlop/s
100 GFlop/s
10 GFlop/s
1 GFlop/s
Performance
● Sum
▲ #1
■ #500