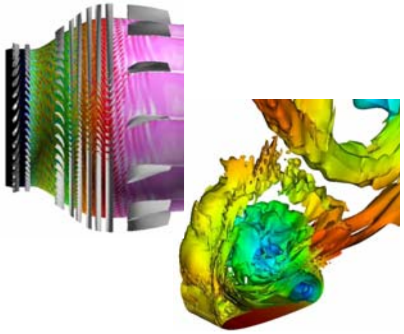
Parallel Computations in Fluid/Thermal Sciences



- **Welcome to MAE267!**
- **My name is Roger L. Davis**
  - 2104 Bainer Hall
  - Phone 530-752-2264
  - davisrl@ucdavis.edu
  - Office Hours by appointment
- **This purpose of this course is to teach engineers how to develop engineering software using parallel computers.**

Grades

- **Grades will be based upon**
  - 20% on homework
  - 80% on projects
- **Schedule of homework, lecture notes, etc. on smartsite**
- **Classes will consist of lectures, programming workshops, and discussion of homework/projects**
- **Lecture notes, etc. can be downloaded from web at:**

smartsite.ucdavis.edu

References

- **Here are some references used for this course:**
  - "UNIX In a Nutshell," Arnold Robbins, 3rd Edition, O'Reilly
  - "Introduction to Parallel Computing", Ananth Grama, Anshul Gupta, George Karypis, and Vipin Kumar, Second Edition, Addison Wesley
  - Introduction to Fortran 90/95," Stephen J. Chapman, First Edition, WCB-McGraw Hill
  - "Using MPI – Portable Programming with the Message-Passing Interface," William Gropp, Ewing Lusk, and Anthony Skjellum, Second Edition, MIT Press
  - "Parallel Programming with MPI," Peter S. Pacheco, Morgan Kaufmann Publishers, Inc.
  - "Numerical Linear Algebra for High-Performance Computers," Jack Dongarra, Iain Duff, Danny Sorensen, and Henk van der Vorst

References (Cont)

- **Here are some more references used for this course:**
  - "Parallel Methods in Numerical Analysis," Stanford University
  - "Computer Architecture – A Quantitative Approach" John L Hennessy and David A Patterson, Second Edition, Morgan Kaufmann
  - "Sourcebook of Parallel Computing", edited by Jack Dongarra, Ian Foster, William Gropp, Ken Kennedy, Linda Torczon, and Andy White, Morgan Kaufmann
  - "Numerical Recipes in Fortran 90 – The Art of Parallel Scientific Computing" William Press, Saul Teukolsky, William Vetterling, and Brian Flannery
  - "Object-Oriented Programmin via Fortran90/95", Ed Akin, Cambridge University Press, 2008

Overview of Course

- **Introduction**
  - Overview of Course, Engineering problem types that must consider parallel computing
- **Overview of Fortran 90/95 for engineering programs**
  - structure of statements and programs, assignment statements, intrinsic functions, I/O, branches and loops, arrays, modules, data-types, pointers, memory allocation
- **Parallel Computer Architectures**
  - vector processors, SMPs, distributed-memory, Beowulf clusters, advantages/disadvantages
- **Parallel Performance Models and Analysis**
  - bandwidth, latency, speedup, Amdahl's law, performance analysis tools
- **MPI (distributed-memory) vs OpenMP (shared-memory) computers and programs**
- **Overview of Data Structures**
  - multi-block structured, unstructured, hybrid, mesh refinement, implicit and explicit algorithms

Overview of Course (cont)

- **Shared memory programming and computing with cores and graphical processing units (GPUs)**
- **Domain decomposition**
  - graph partitioning, bisection, Metis, ParMetis, Chaco
- **Distributed memory programming**
  - message passing interface, MPI
  - message passing interface, issues with multi-block structured solvers
  - message passing interface, issues with unstructured-grid solvers
- **SPMD vs MPMD programming**
  - considerations for multi-disciplinary engineering simulations
- **Other parallel engineering applications – optimization and sorting**
- **Impact of parallel computing on implicit and explicit solution algorithms, parallel computing algorithm libraries**
- **Parallel visualization tools, parallel pre- and post-processing**

Let's Get Started…

- **Objectives of parallel computing:**
  - Ultimately, in parallel computing, we intend to achieve:
    - Faster execution speed
      - Enable multiple analyses in a fixed amount of time
      - Decrease time necessary to complete one solution
      - Increase the level of modeling of our physical system
  - Larger problem size
    - Enable higher grid resolution than possible in single processor machines
    - Introduce additional physical models that were impossible to tackle before
  - Numerical experiments
    - Simulate phenomena that cannot be recreated or measured in the laboratory

Lecture 1A – Objectives of Parallel Computing

- Lower cost
  - Strictly speaking, it is nearly impossible to obtain lower cost when using a parallel computer (parallel processing overhead, additional expense of interconnection network, etc.)
  - Lower cost can be derived from additional benefits that result from the ability to execute a given program in a shorter amount of time
  - However, we must strive to maintain the cost of parallel computing from departing severely from single processor computations
  - Economies of scale?

Objectives of Parallel Computing

- **What parallel computing is NOT**
  - A brute force method to overcome limitations of your baseline algorithms/solution procedure
  - A “cool” way of getting solutions faster
  - An absolute need for every existing application

Objectives of Parallel Computing

- **Our objective in this course is to:**
  - Introduce students to the basic tool-set used in parallel computing for engineering problems
  - Determine what engineering problems can take advantage of parallel computing
  - How to set up an engineering problem to compute in parallel
  - How to program a computer code to solve an engineering problem with parallel computers

Current Industry Practice

- **Parallel computing is used extensively in certain industry:**
  - Aerospace:
    - Boeing, General Electric, Pratt & Whitney
    - Computational fluid dynamic simulations
      - External (fuselage, wing, nacelle, etc.)
      - Internal (engine compressor, turbine, combustor, inlet, nozzles)
    - Design optimization
  - Automobile:
    - Ford, General Motors
    - Computational fluid dynamic simulations
      - External (body drag)
      - Internal (underhood, passenger compartment, fans)
  - Electronics:
    - HP, Silicon Graphics
      - Chip and board heat transfer

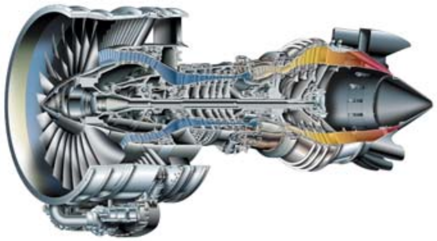
Examples

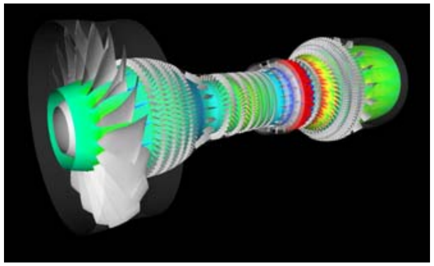
- **The examples that I will show in the following slides focus on parallel computing of fluid dynamic problems.**
- **However, there are many other sciences such as**
  - Heat transfer
  - Chemically reaction kinetics
  - Electromagnetics
  - Acoustics
  - Structural dynamics

  where parallel computing is used just as much or even more!

High-End Examples of Parallel Computing
Entire Jet-Engine Main Flow-Path

- **DoE Accelerated Computing Initiative:**
  - Develop the parallel computer hardware and software technology to solve large-scale, multi-disciplinary scientific problems never-before attempted
  - Example: 3-D flow through an entire jet engine



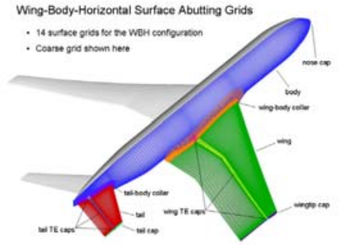


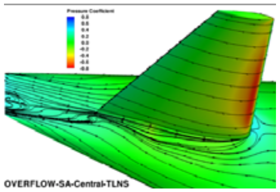
- **20^0 Sector Unsteady-Flow Simulation**
- **14 M Points (coarse-grid)**
- **75M Points (fine-grid)**
- **700 Processors (coarse-grid)**
- **4,000 Processors (fine-grid)**
- **~14 days Turn-around**

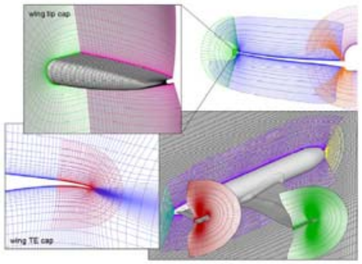
Stanford University

Entire Aircraft

- **AIAA Drag Prediction Workshop (Vassberg et al. at Boeing, NASA, and universities)**
  - Prediction of C_L and C_D of complete aircraft
  - Accuracy issues due to
    - Grid quality and topology
    - Solver





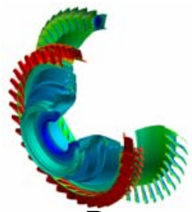


- **7 M Points (coarse-grid)**
- **2.4B Points (ultra-fine-grid)**
- **16 Processors (coarse-grid)**
- **4140 Processors (fine-grid)**

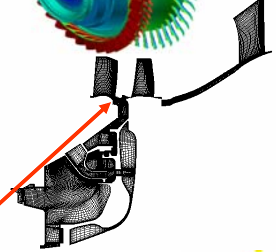
Boeing, NASA

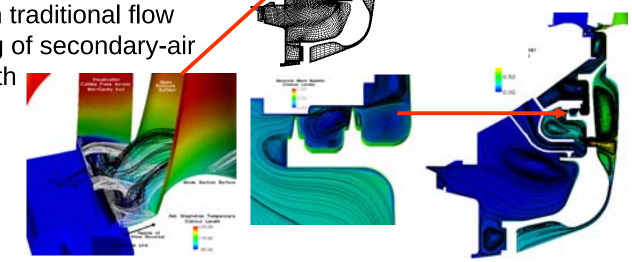
Main/Secondary Turbine Flow-Path Simulation

- **Integrated High-Pressure Turbine Main and Secondary-Air System Flow Paths**
- **Investigation Featured:**
  - Description of main/secondary-air interaction and a source of hot-gas ingestion into disk cavity
  - Flow physics in/around seals of secondary-air system
  - Weaknesses with traditional flow leakage modeling of secondary-air into main flow path



UC Davis and
Stanford University

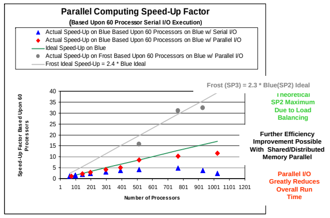


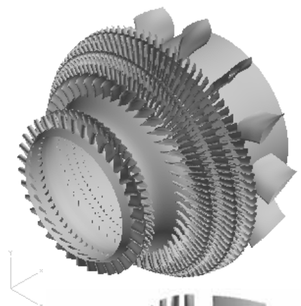


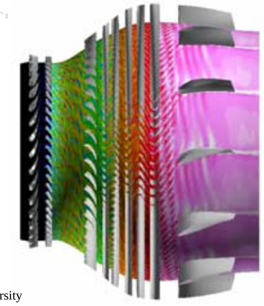
- **Steady-Flow Simulation**
- **9.4 M Points**
- **238 Blocks**
- **144 Processors, IBM SP3**
- **~100 days Turn-around**

Unsteady-Flow Simulation
Integrated High- and Low-Pressure Turbine

- **1/6 Circumference Modeled**
- **93.8 M Points**
- **2192 Blocks**
- **384-640 Processors**
- **5,700 Time-Steps (w/ 30 inner iterations per time-step) Required**
- **~85 days (clock time), ~1.3M Hours (cpu time) on Frost (IBM SP3)**
  - 640 processors (40 nodes)

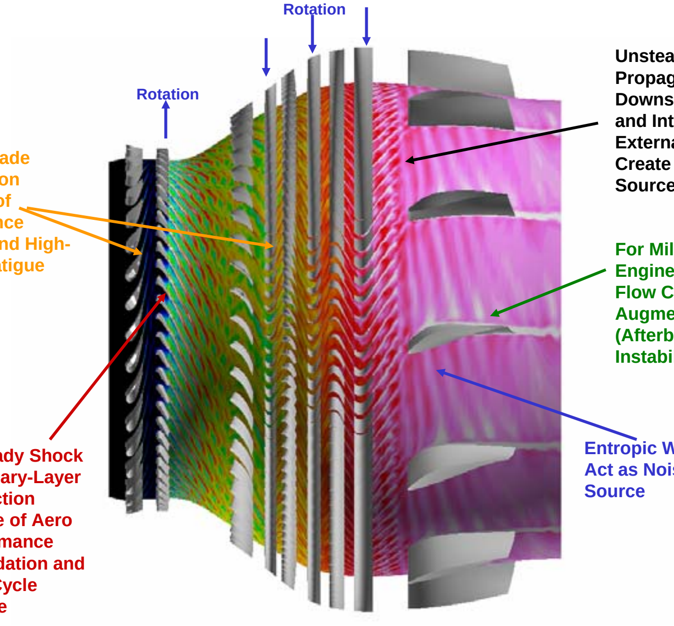






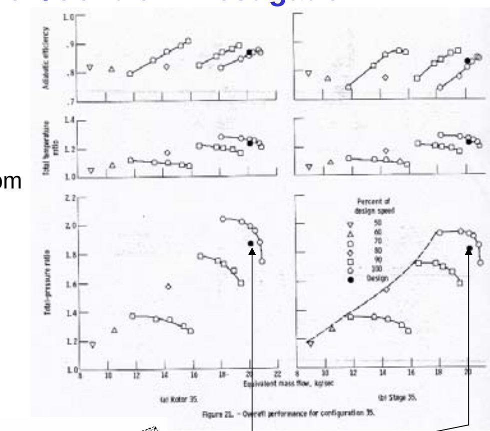
UC Davis and
Stanford University

Flow Features Highlighted by Simulation

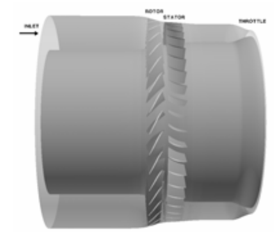


Stage 35 Stall-Inception/Control Investigation

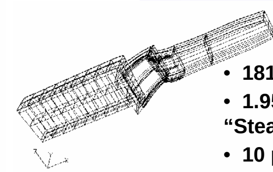
- **NASA Stage-35 Compressor**
  - Picked by GE and NASA as meta-goal target problem
    - Flow physics leading to stall
    - Flow control of stall
  - Geometry and reports acquired from NASA
  - Single-Stage
    - 36 rotors, 46 stators



Current Investigation



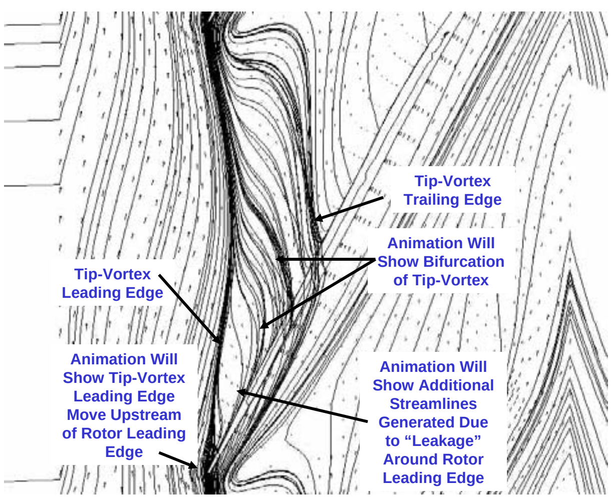
Computational Model



UC Davis and GE

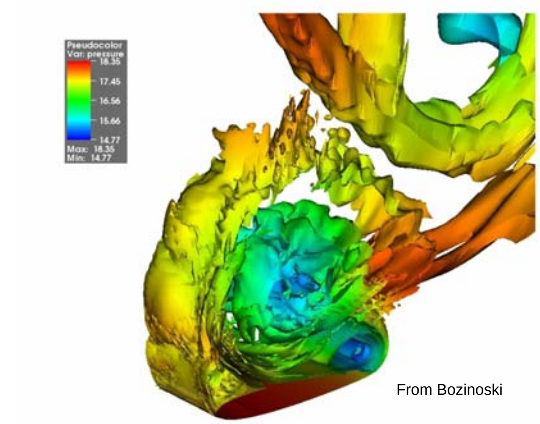
- **181 Grid Blocks**
- **1.95M Grid Points for "Steady" Simulation**
- **10 processors**

Stall Inception

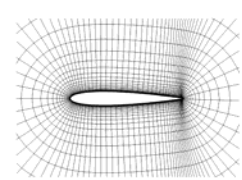


Airfoil Stall

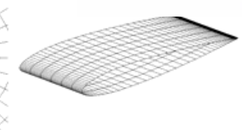
- **Detached-Eddy Simulation of NACA0012 at high angle of attack**
  - 1.8M grid points
  - 20 processors on wopr/vortex



From Bozinoski



(a) Stream-wise view

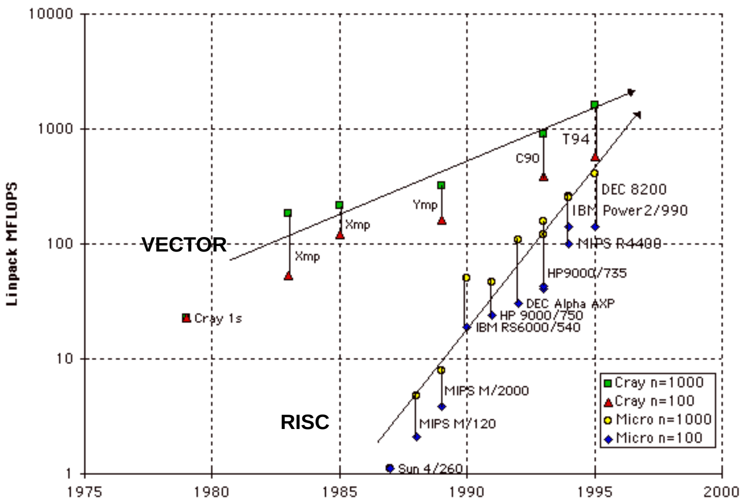


(b) Span-wise view

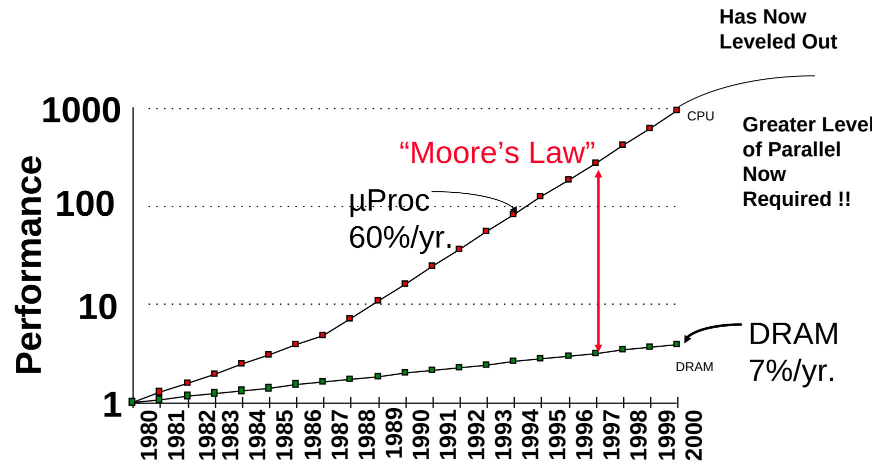
UC Davis

	C_L	C_D
2D URANS	1.32	2.24
2D DES	1.34	2.28
3D URANS	1.40	2.37
3D DES	1.08	1.87
Experiment	0.90	1.60

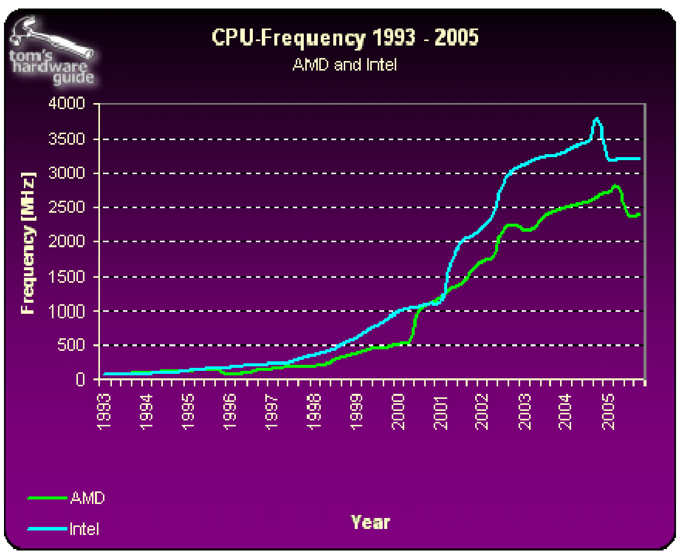
Microprocessor Performance



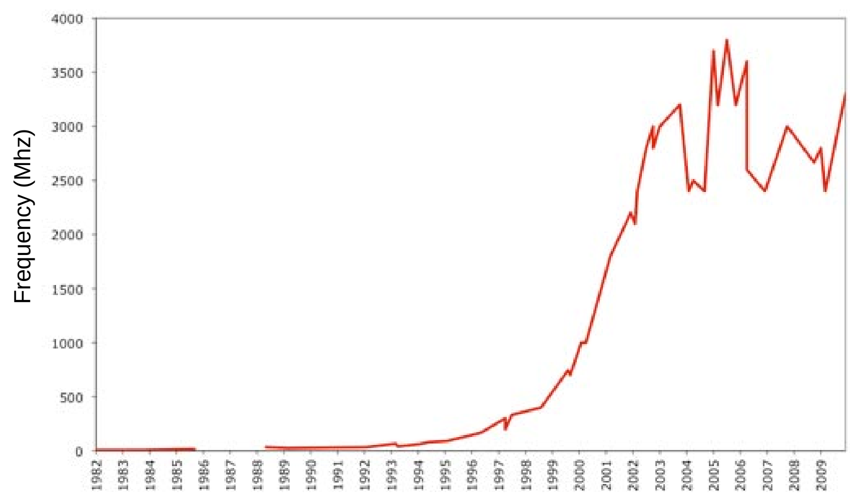
Memory Subsystems



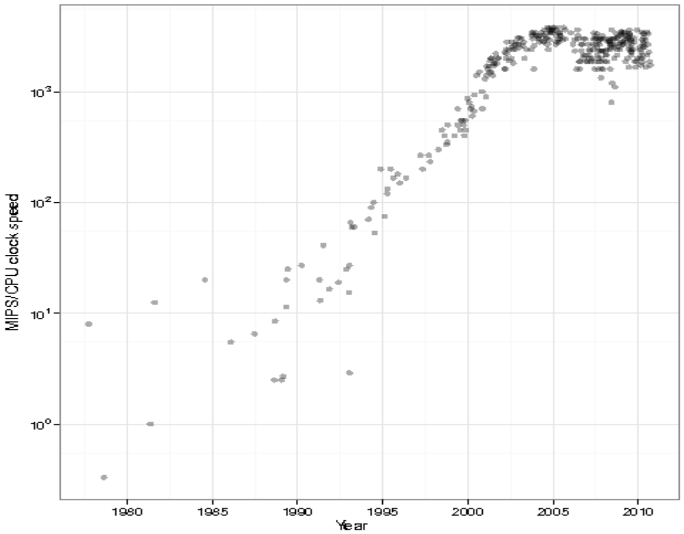
Microprocessor Speed



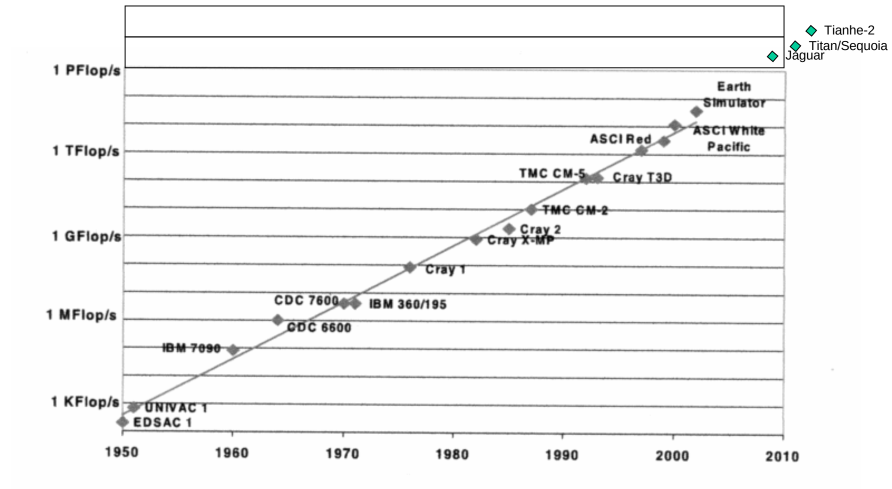
Microprocessor Speed



Microprocessor Speed

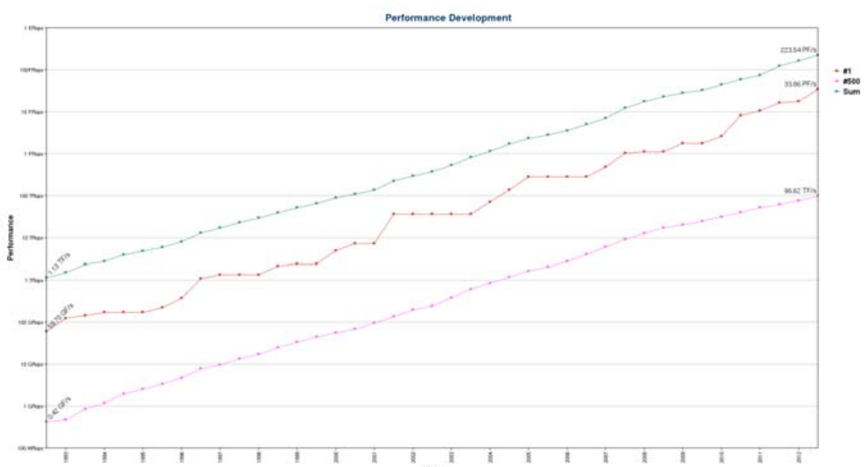


Computer Performance History



From Dongarra
and top500.org

Computer Performance History



From top500.org