

---

# Performance Characterization

# Lab 1

---

- Deploy and perform real HW evaluation with SpecCPU2006
  - 29 Applications
  - Compiler effect
- Study
  - ISA and optimization influence
  - Bottleneck analysis, Energy waste
- Deploy and evaluation with SpecWeb2005 and SpecJBB2005
  - 3 Applications: banking website, e-commerce website and a vendor support website
  - Appserver

# CPU Benchmarks

---

- Designed to stress and compare processor/memory architecture
  - Little to none influence of other computer components (e.g. I/O)
  - “Supposedly” low OS impact
- Most widely use SpecCPU2006
  - Programs try to cover a ample aspect of use
  - Mostly single core (no multithreaded) although might be used to evaluate a multi-core in throughput computing environment
- Designed against bench-marketing
  - Every bit of it is designed to avoid “tampering” with the results

# Everything is distilled down a number :the SPECmark

---

- Reference machine: Sun Ultra Enterprise II (1997/ 297 Mhz USII)
- Latency SPECmark (SPECint & SPECfp)
  - For each benchmark
    - Take odd number of samples: on both machines
    - Choose median
    - Take latency ratio (Sun Ultra Enterprise II / your machine)
  - Take GMEAN of ratios over all benchmarks
- Throughput SPECmark (SPECint\_rate & SPECfp\_rate)
  - Run multiple benchmarks in parallel on multiple-processor system
- Recent (latency) leaders
  - SPECint: [Intel Xeon E5-2699](#) (63.6)
  - SPECfp: Intel Xeon E5-2699 (101)

# Components of SPECcpu 2006

---

## ■ Floating Point

- <http://www.spec.org/cpu2006/CFP2006/>
- 17 programs, C, C++ and Fortran.
- Most cases physical phenomena, modeling
- Quite regular, and “usually” memory bound

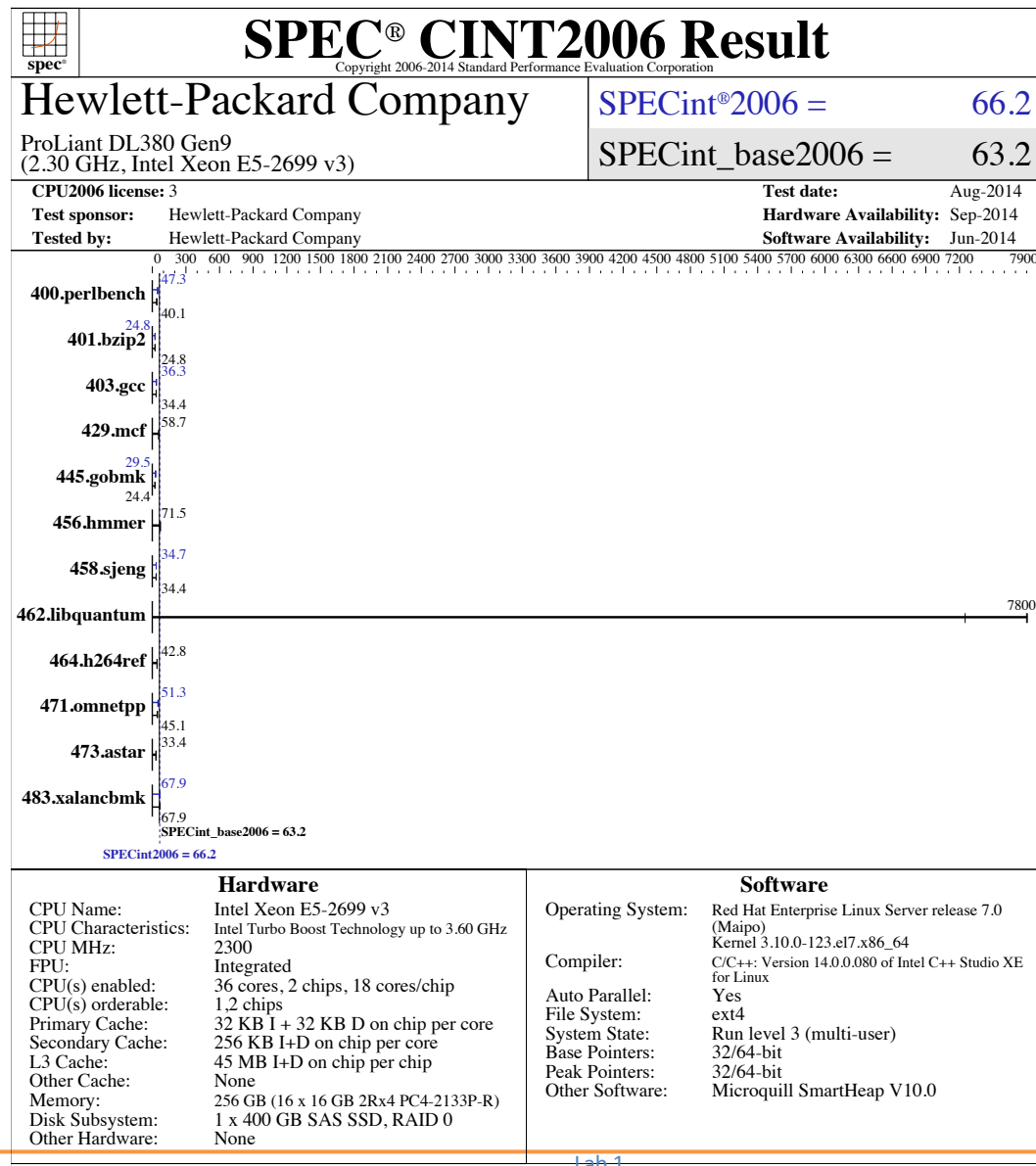
## ■ Integer

- <http://www.spec.org/cpu2006/CINT2006/>
- 12 programs, C and C++
- Most cases data processing or “logic” simulations
- Quite irregular code, and “usually” CPU bound

## ■ OS & Compiler agnostic

- Via portability flags (conditional compilation)

# Example Report



# Some notes

---

- Peak performance
  - Represents the results for “best” app dependent compiler options
- Base
  - Represents the results for a “common” set of options across all the apps in the suite (using the same compiler)
- Modes
  - Default
    - Only affected by single core performance
    - Beware! Today's processors use core independent DVFS (turbo boost)
      - 3.6Ghz versus 2.3Ghz
  - Rate
    - Is usually used to compare performance in a throughput environment
    - Affected by the number of cores, threads per core, etc...
    - Interesting for servers in cloud-computing tasks

# Task 1

---

- Download code
  - <http://www.atc.unican.es/%7evpuente/SOA/SPECcpu2006.tar.gz>
- Run the whole SPEC suite
  - Setup the environment for C/C++/Fortran
    - Use the provided scripts to produce the report of your hardware
  - Determine the influence of the ISA (from x86 to x86-64)
    - Compare a 32-bits and 64-bit compilation
  - Compare profile-guided compilation (runspec supports it)
- Choose one of benchmarks in each suite (both INT & FP)
  - Determine which benchmarks are memory bounded and which CPU bounded (i.e. determine IPC)
  - Determine how much energy is wasted due to speculation (i.e. obtain the efficiency of the branch predictor and memory speculation)
- Use “train” input for whole benchmark runs
  - Choose the most remarkable application for “ref.” input



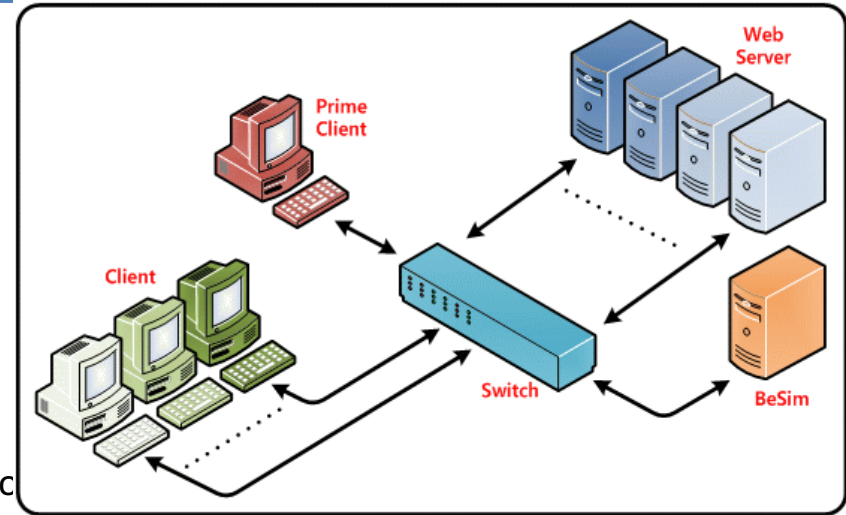
# Deployment of Complex Workloads

---

- Real life systems aren't simple cpu/mem bounded apps
- More like a combination of pieces working together to provide a results with substantial influence from I/O and other SW components (OS, DBMS, WebServer, JVM, etc...)
  - Hard to deploy and maintain
  - Hard to optimize
- Although, not widely used, there are also benchmarks
  - SpecVIRT (SpecWeb, SpecJbb)
  - TPC-XX,
  - BigBench,
  - ...

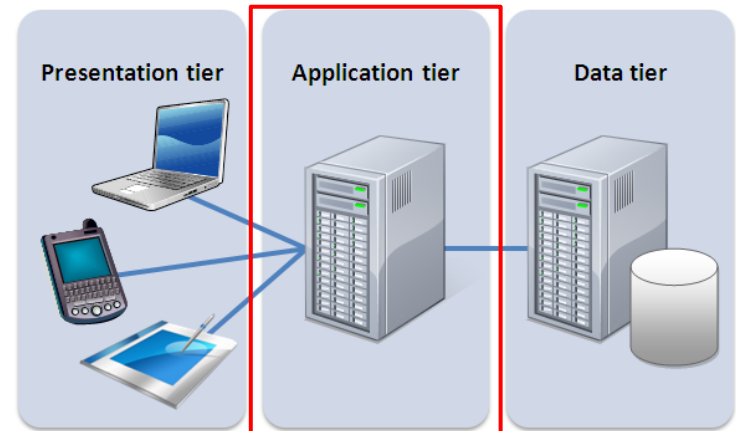
# Real “Life” use (Level 2)

- Use SpecWeb2005 as guinea pig
  - Now is obsolete... but still useful
- Three mayor components
  - Banking Workload
    - Heavy dynamic content and fully secure
  - Ecommerce workload
    - Mixed static/dynamic content, some portic uses SSL
  - Support workload
    - Heavy static download, no SSL
- Performance measure is not easy (but not that hard either)
  - 3 pieces= Web server (jsp or php) + Back End Simulator + Clients
  - Although designed to run ins separate machines, we will use the same machine for the 3 pieces
- Hard to deploy
  - Many crosscutting issues: Apache, php, JSP, SSL, ...



# Real “Life” use (Level 1)

- Use SPECjbb2005
  - Representative benchmark for commercial servers
  - Evaluates the server side java in a three-tier client/server system (with emphasis in the middle tier)
  - Partially emulates an OLTP system
  - 1<sup>st</sup> degree approximation to other more complex approaches (TPC-C)
- Evaluates
  - Runtime: JVM, JIT, OS
  - CPU, Mem
  - Little I/O
- Much easier to deploy than SPECweb2005
  - Even easier than SPECcpu
  - The whole thing (setup and run) is encapsulated in a jar file



# Task 2

---

- Download code
  - <http://www.atc.unican.es/%7evpuente/SOA/SPECweb2005.tar.bz2>
  - <http://www.atc.unican.es/%7evpuente/SOA/SPECjbb2005.tar.gz>
- Software stack influence in complex workloads:
  - Effects of web-server in SPECweb2005 (apache vs nginx)
  - Effects of JVM in SPECjbb2005(Oracle JVM vs *gpl* JVMs)
- What is the effect of I/O system and runtime in performance? Use the tool u want

# Grades

---

- Task 1
  - Straightforward
  - Will require read documentation and “fix” some C/C++ code
- Task 2
  - Level 1
    - Read more advanced documentation
    - Even easier to deploy than task 1
  - Level 2
    - Hard, especially SSL portions
    - Will require configure apache/nginx (php, mod-cgi, SSL)
    - I will help
- We will use this in next lab
  - Use profuse documentation, specially in task2.2