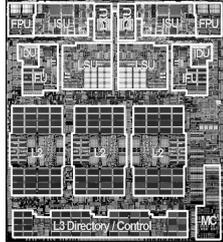




Università degli Studi
di Cassino



**Corso di
Calcolatori Elettronici II**

Misura delle prestazioni

Anno Accademico 2006/2007

Francesco Tortorella

Misura delle prestazioni

E' fondamentale definire una metrica corretta per misurare le prestazioni di un sistema di elaborazione.

- Prospettiva dell'acquirente
 - dato un insieme di macchine, quale ha
 - le migliori prestazioni ?
 - il minor costo ?
 - Il miglior rapporto prestazioni/costo ?
- Prospettiva del progettista
 - di fronte a possibili soluzioni progettuali alternative, quale presenta
 - il miglior incremento delle prestazioni ?
 - il minor costo ?
 - il miglior rapporto prestazioni/costo ?

Calcolatori Elettronici II
Lezione 1 - 1

F. Tortorella © 2007
Università degli Studi
di Cassino

Come si misurano le prestazioni ?

- Quali sono le grandezze tipicamente usate ?
 - frequenza di clock
 - dimensione della RAM
 - dimensione dei dischi
- Qual è il parametro che ci interessa realmente ?

Calcolatori Elettronici II
Lezione 1 - 2

F. Tortorella © 2007
Università degli Studi
di Cassino

Due concetti di prestazioni

Aereo	Capacità	Autonomia	Velocità di crociera	Portata
Boeing 777	375	4630	610	228750
Boeing 747	470	4150	610	286700
Douglas DC8	146	8720	544	79424

- Quale aereo ha le migliori prestazioni ?
- **Prospettiva del passeggero**
 - *Tempo impiegato per un singolo viaggio*
- **Prospettiva della compagnia aerea**
 - *Numero di passeggeri trasferiti in un dato intervallo di tempo*

Calcolatori Elettronici II
Lezione 1 - 3

F. Tortorella © 2007
Università degli Studi
di Cassino

Due misure possibili

- Tempo di risposta (o di esecuzione)
 - Quanto tempo devo aspettare per l'uscita del mio programma?
 - Quanto tempo per lanciare il mio programma ?
 - Quanto tempo per eseguire il mio programma ?
- Throughput
 - Quanto lavoro viene svolto?
 - Quanti programmi possono essere eseguiti insieme?
 - Qual è il rate di esecuzione medio ?

Prestazione = f(tempo di esecuzione)

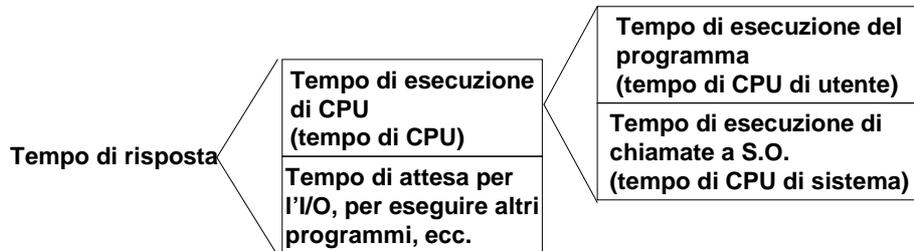
- Per un dato programma eseguito sulla macchina X,

$$\text{Prestazione}(X) = 1 / \text{tempo di esecuzione}(X)$$

- "X è n volte più veloce di Y" se:

$$\text{Prestazione}(X) / \text{Prestazione}(Y) = n$$

Che tempo fa ?

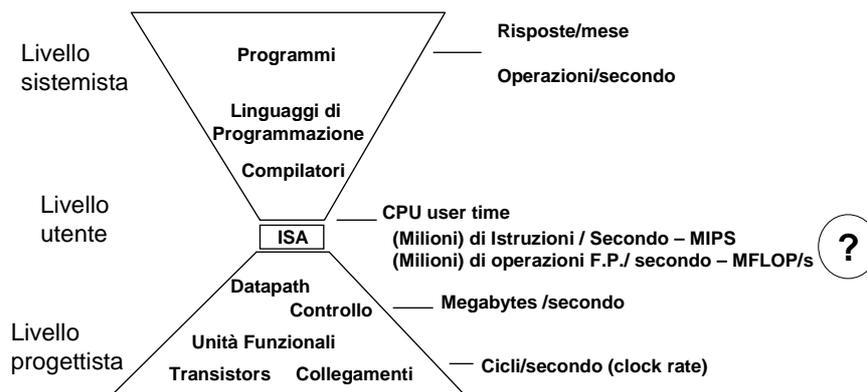


Parametro di riferimento:
tempo di CPU di utente (*user CPU time*)

Calcolatori Elettronici II
Lezione 1 - 6

F. Tortorella © 2007
Università degli Studi
di Cassino

Metriche per misurare le prestazioni



Diverse metriche per diversi aspetti (e per diversi abusi)
Come si relazionano le diverse metriche ?

Calcolatori Elettronici II
Lezione 1 - 7

F. Tortorella © 2007
Università degli Studi
di Cassino

Clock

- Nel valutare le prestazioni di un sistema di calcolo, è necessario tenere presente che la CPU è sincronizzata da un orologio interno (clock), segnale periodico definito da una propria frequenza $f=1/T$
- I "clock ticks" definiscono gli istanti possibili per la realizzazione di eventi in hardware (evoluzione dello stato della macchina) :



- tempo di ciclo= intervallo tra due ticks = secondi per ciclo
- clock rate (frequenza) = cicli al secondo (1 Hz. = 1 ciclo/sec)

Un clock da 1 Ghz ha un tempo di ciclo di $\frac{1}{10^9}=1$ nanosecondo

Da che cosa dipendono le prestazioni ?

- E' possibile esprimere il tempo di esecuzione in termini di cicli di clock
$$\frac{\text{secondi}}{\text{programma}} = \frac{\text{cicli}}{\text{programma}} \times \frac{\text{secondi}}{\text{ciclo}}$$
- A parità di altre condizioni, si ottiene un aumento delle prestazioni se
 - diminuisce il numero di cicli/programma, oppure
 - diminuisce il tempo di ciclo del clock o, equivalentemente,
 - aumenta il clock rate.
- Bisogna però tenere conto che, nelle architetture reali, i diversi parametri possono essere correlati

Esempio

Un programma viene eseguito in 10 secondi sul computer A, che ha un clock da 4 Ghz. E' però necessario che venga eseguito in 6 secondi e per questo motivo si intende costruire una nuova macchina B con una nuova tecnologia di realizzazione della CPU che permette un incremento notevole della frequenza di clock, ma a spese del numero di cicli per istruzione che porterebbe ad un aumento del 20% del numero di cicli richiesti per l'esecuzione del programma. Quale frequenza di clock deve assicurare la macchina B ?

Equazione fondamentale:

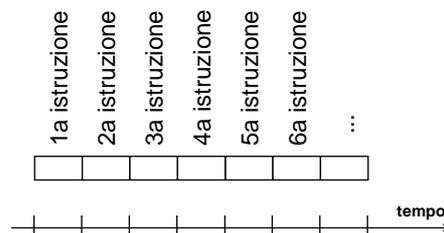
$$\frac{\text{secondi}}{\text{programma}} = \frac{\text{cicli}}{\text{programma}} \times \frac{\text{secondi}}{\text{ciclo}}$$

Calcolatori Elettronici II
Lezione 1 - 11

F. Tortorella © 2007
Università degli Studi
di Cassino

Quanti cicli per eseguire un programma ?

- Si può assumere numero di cicli = numero di istruzioni ?



No !

Istruzioni differenti comportano tempi diversi su macchine differenti.

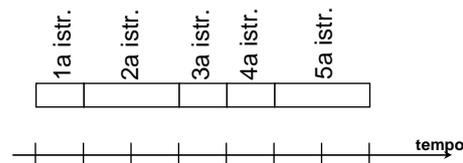
Perché ?

Calcolatori Elettronici II
Lezione 1 - 12

F. Tortorella © 2007
Università degli Studi
di Cassino

Istruzioni differenti richiedono numeri differenti di cicli

- Un'operazione di moltiplicazione richiede un tempo maggiore rispetto ad un'addizione



- Un'operazione tra floating point richiede un tempo maggiore rispetto ad una tra interi
- Un accesso in memoria richiede un tempo maggiore rispetto ad un accesso a registri interni

Calcolatori Elettronici II
Lezione 1 - 13

F. Tortorella © 2007
Università degli Studi
di Cassino

Dove sono le istruzioni ?

$$\frac{\text{secondi}}{\text{programma}} = \frac{\text{cicli}}{\text{programma}} \times \frac{\text{secondi}}{\text{ciclo}}$$

- Nell'equazione considerata non c'è riferimento (esplicito) al numero di istruzioni che formano il programma; chiaramente, questo è un parametro che influisce sul tempo di esecuzione.
- Come si può esplicitare il rapporto con il numero di istruzioni ? Definiamo il numero medio di cicli per istruzione: CPI (*clock cycles per instruction*).

$$\text{CPI} = \frac{\text{Numero di cicli di clock del programma}}{\text{Numero di istruzioni del programma}}$$

Calcolatori Elettronici II
Lezione 1 - 14

F. Tortorella © 2007
Università degli Studi
di Cassino

- In questo modo diventa chiaro la dipendenza del tempo di esecuzione dal numero di istruzioni:

$$\frac{\text{secondi}}{\text{programma}} = \text{numero istruzioni} \times \text{CPI} \times \frac{\text{secondi}}{\text{ciclo}}$$

- Quali sono gli aspetti che influenzano i parametri evidenziati ?

Come si dividono le responsabilità ?

	Numero istruzioni	CPI	Clock rate
Algoritmo	X		
Compilatore	X	X	
ISA	X	X	X
Organizzazione		X	X
Tecnologia			X

Istruzioni e CPI

- Le istruzioni che contribuiscono a formare il parametro CPI sono di diverso tipo e caratterizzate da differenti numeri di cicli per l'esecuzione.
- E' quindi possibile raggruppare le diverse istruzioni in classi caratterizzate dallo stesso numero di cicli di clock, rendendo esplicito il contributo di ciascuna classe al valore di CPI

$$\text{cicli totali} = \sum_{i=1}^n \text{CPI}_i \times n_i$$

- CPI_i = CPI per le istruzioni della classe i -ma
- n_i = numero di istruzioni della classe i -ma

Istruzioni e CPI

- Alla fine il CPI è

$$\text{CPI} = \frac{\sum_{i=1}^n \text{CPI}_i \times n_i}{N} = \sum_{i=1}^n \text{CPI}_i \times \frac{n_i}{N} = \sum_{i=1}^n \text{CPI}_i \times f_i$$

- f_i = frequenza della classe i -ma
- N = numero totale istruzioni

- In quale direzione andare per migliorare il CPI ?

Esempio

- Una macchina è caratterizzata dalla seguente distribuzione di CPI su tre classi:

Classe	CPI
A	1
B	2
C	3

- Una particolare istruzione in HLL può essere tradotta da un compilatore tramite due sequenze possibili, che usano combinazioni diverse di istruzioni delle tre classi, secondo la tabella seguente:

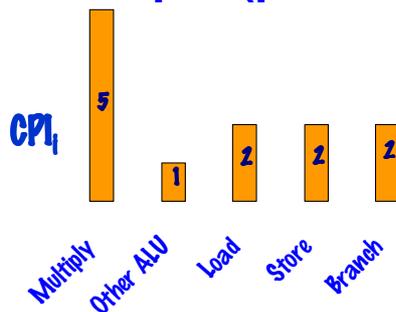
	A	B	C
S1	2	1	2
S2	4	1	1

- Qual è la soluzione migliore ?

Calcolatori Elettronici II
Lezione 1 - 19

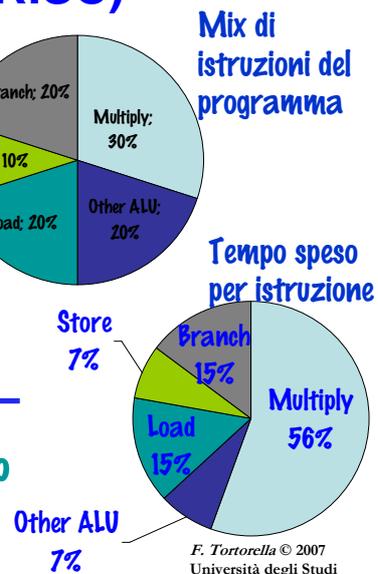
F. Tortorella © 2007
Università degli Studi
di Cassino

Esempio (processore RISC)



$$\frac{5 \times 30 + 1 \times 20 + 2 \times 20 + 2 \times 10 + 2 \times 20}{100} = 2.7 \text{ cicli/istruzione } \text{CPI complessivo}$$

Calcolatori Elettronici II
Lezione 1 - 21



F. Tortorella © 2007
Università degli Studi
di Cassino

La legge di Amdahl

- Il miglioramento di prestazione (accelerazione o speedup) ottenibile mediante l'uso di alcune modalità di esecuzione più veloci è limitato dalla frazione di tempo in cui queste modalità possono essere impiegate.
- Supponiamo di aver apportato una modifica E ad una certa macchina. Lo speedup che si ottiene si valuta come:

$$\text{Speedup}(E) = \frac{\text{Prestazione(dopo di E)}}{\text{Prestazione(prima di E)}} = \frac{\text{Tempo(prima di E)}}{\text{Tempo(dopo di E)}}$$

La legge di Amdahl

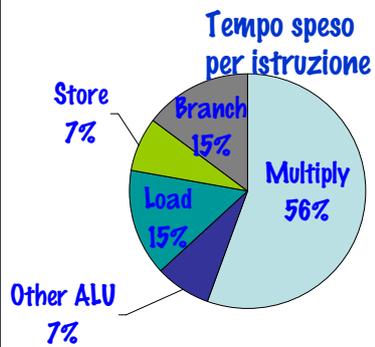
- Supponiamo che la modifica E porti ad un'accelerazione pari ad un fattore S di una frazione F dell'intero task, mentre il resto rimane inalterato:

$$\text{ExTime(dopo di E)} = ((1-F) + F/S) \times \text{ExTime(prima di E)}$$

- Per cui lo speedup è

$$\text{Speedup}(E) = \frac{1}{(1-F) + F/S}$$

La legge di Amdahl



Che cosa succede se si rende il Multiply infinitamente veloce lasciando inalterate le altre istruzioni ?

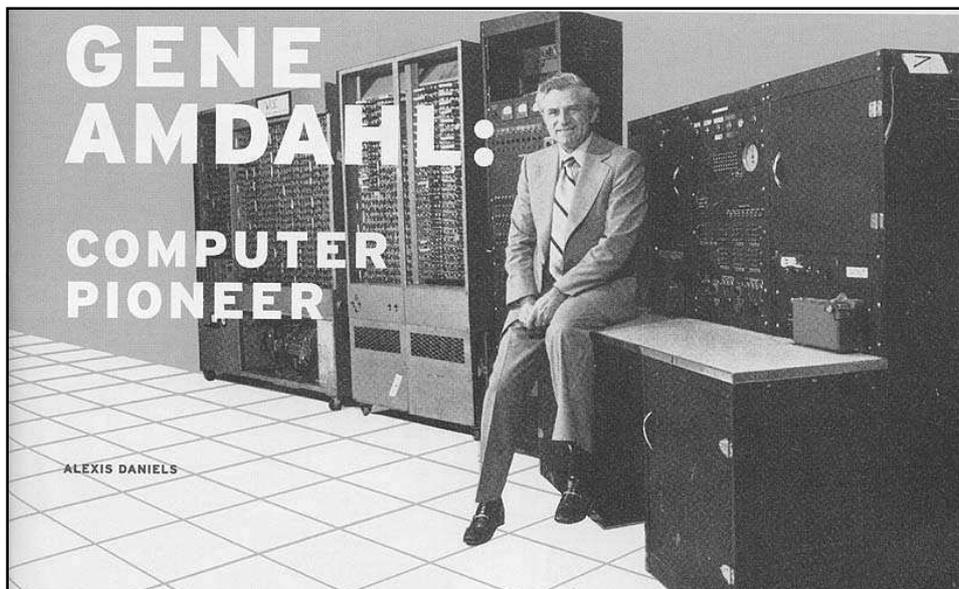
$$\text{Speedup} = \frac{1}{\frac{56}{\infty} + 0.44} = 2.08$$

Lezione ?

Necessario migliorare in modo bilanciato

Calcolatori Elettronici II
Lezione 1 - 24

F. Tortorella © 2007
Università degli Studi
di Cassino



Calcolatori Elettronici II
Lezione 1 - 25

F. Tortorella © 2007
Università degli Studi
di Cassino

Programmi per valutare le prestazioni

- Quale programma usare per valutare le prestazioni?
- L'ideale sarebbe di usare l'insieme di programmi che si sa di dover eseguire sulla macchina (**workload**), ma non è sempre possibile.
- In alternativa, si potrebbero usare dei programmi campione (**benchmark**).
- Difficoltà:
 - le caratteristiche del benchmark devono essere simili a quelle del workload
 - il benchmark deve essere standard
- Soluzioni:
 - benchmark sintetici (Whetstone, Dhrystone, kernel benchmark)
 - mix di applicazioni reali (**SPEC System Performance Evaluation Cooperative**)

Calcolatori Elettronici II
Lezione 1 - 26

F. Tortorella © 2007
Università degli Studi
di Cassino

SPEC CPU2006 Benchmark Descriptions

The benchmarks are described in order by category - first the integer benchmarks, then the floating point benchmarks.

Part 1: Integer Benchmarks

- 400.perlbench
- 401.bzip2
- 403.gcc
- 429.mcf
- 445.gobmk
- 456.hmmer
- 458.sjeng
- 462.libquantum
- 464.h264ref
- 471.omnetpp
- 473.astar
- 483.xalancbmk

Part 2: Floating Point Benchmarks

- 410.bwaves
- 416.gamess
- 433.milc
- 434.zeusmp
- 435.gromacs
- 436.cactusADM
- 437.leslie3d
- 444.namd
- 447.dealII
- 450.soplex
- 453.povray
- 454.calculix
- 459.GemsFDTD
- 465.tonto
- 470.lbm
- 481.wrf
- 482.sphinx3
- 999.specrand

*“An ounce of honest
data is worth a
pound of marketing
hype”*

Calcolatori Elettronici II
Lezione 1 - 27

F. Tortorella © 2007
Università degli Studi
di Cassino

CINT2006 (Integer Component of SPEC CPU2006):

Benchmark	Language	Application Area	Brief Description
400.perlbench	C	Programming Language	Derived from Perl V5.8.7. The workload includes SpamAssassin, MHonArc (an email indexer), and speediff (SPEC's tool that checks benchmark outputs).
401.bzip2	C	Compression	Julian Seward's bzip2 version 1.0.3, modified to do most work in memory, rather than doing I/O.
403.gcc	C	C Compiler	Based on gcc Version 3.2, generates code for Opteron.
429.mcf	C	Combinatorial Optimization	Vehicle scheduling. Uses a network simplex algorithm (which is also used in commercial products) to schedule public transport.
445.gobmk	C	Artificial Intelligence: Go	Plays the game of Go, a simply described but deeply complex game.
456.hammer	C	Search Gene Sequence	Protein sequence analysis using profile hidden Markov models (profile HMMs)
458.sjeng	C	Artificial Intelligence: chess	A highly-ranked chess program that also plays several chess variants.
462.libquantum	C	Physics / Quantum Computing	Simulates a quantum computer, running Shor's polynomial-time factorization algorithm.
464.h264ref	C	Video Compression	A reference implementation of H.264/AVC, encodes a videostream using 2 parameter sets. The H.264/AVC standard is expected to replace MPEG2
471.omnetpp	C++	Discrete Event Simulation	Uses the OMNet++ discrete event simulator to model a large Ethernet campus network.
473.astar	C++	Path-finding Algorithms	Pathfinding library for 2D maps, including the well known A* algorithm.
483.xalanbmk	C++	XML Processing	A modified version of Xalan-C++, which transforms XML documents to other document types.

*Calcolatori Elettronici II
Lezione 1 - 28*

F. Tortorella © 2007
Università degli Studi di Cassino

SPEC [®] CINT2006 Result																																								
Tyan	SPECint [®] 2006 = --																																							
Tyan Thunder K8E Tomcat (S2865) (AMD Opteron 146)	SPECint_base2006 = 9.02																																							
Test sponsor: Advanced Micro Devices Tested by: Advanced Micro Devices																																								
CPU2006 license #: 49 Test date: Apr-2006 Hardware Availability: Aug-2005 Software Availability: Apr-2006																																								
<table border="1"> <thead> <tr> <th>Benchmark</th> <th>Rate</th> <th>Time</th> </tr> </thead> <tbody> <tr><td>400.perlbench</td><td>1.0</td><td>1.0</td></tr> <tr><td>401.bzip2</td><td>1.0</td><td>1.0</td></tr> <tr><td>403.gcc</td><td>1.0</td><td>1.0</td></tr> <tr><td>429.mcf</td><td>1.0</td><td>1.0</td></tr> <tr><td>445.gobmk</td><td>1.0</td><td>1.0</td></tr> <tr><td>456.hammer</td><td>1.0</td><td>1.0</td></tr> <tr><td>458.sjeng</td><td>1.0</td><td>1.0</td></tr> <tr><td>462.libquantum</td><td>1.0</td><td>1.0</td></tr> <tr><td>464.h264ref</td><td>1.0</td><td>1.0</td></tr> <tr><td>471.omnetpp</td><td>1.0</td><td>1.0</td></tr> <tr><td>473.astar</td><td>1.0</td><td>1.0</td></tr> <tr><td>483.xalanbmk</td><td>1.0</td><td>1.0</td></tr> </tbody> </table>		Benchmark	Rate	Time	400.perlbench	1.0	1.0	401.bzip2	1.0	1.0	403.gcc	1.0	1.0	429.mcf	1.0	1.0	445.gobmk	1.0	1.0	456.hammer	1.0	1.0	458.sjeng	1.0	1.0	462.libquantum	1.0	1.0	464.h264ref	1.0	1.0	471.omnetpp	1.0	1.0	473.astar	1.0	1.0	483.xalanbmk	1.0	1.0
Benchmark	Rate	Time																																						
400.perlbench	1.0	1.0																																						
401.bzip2	1.0	1.0																																						
403.gcc	1.0	1.0																																						
429.mcf	1.0	1.0																																						
445.gobmk	1.0	1.0																																						
456.hammer	1.0	1.0																																						
458.sjeng	1.0	1.0																																						
462.libquantum	1.0	1.0																																						
464.h264ref	1.0	1.0																																						
471.omnetpp	1.0	1.0																																						
473.astar	1.0	1.0																																						
483.xalanbmk	1.0	1.0																																						
SPECint_base2006 = 9.02																																								
Hardware CPU Name: AMD Opteron 146 CPU Characteristics: 2000 CPU MHz: 2600 FPU: Integrated CPU(s) enabled: 1 core, 1 chip, 1 core/chip CPU(s) on-line: 1 chip Primary Cache: 64 KB I + 64 KB D on chip per chip Secondary Cache: 2 MB I+D on chip per chip L2 Cache: None Other Cache: None Memory: 2 GB (2 x 1 GB DDR400 CL2) Disk Subsystem: 250GB SATA Other Hardware: None	Software Operating System: SUSE Linux 10.1 (for AMD64) Compiler: gcc 4.1.1 (for AMD64) Auto Parallel: No File System: ext3 System State: manual 3 Base Program: None Peak Program: Not Applicable Other Software: None																																							
Standard Performance Evaluation Corporation http://www.spec.org Page 1																																								

*Calcolatori Elettronici II
Lezione 1 - 30*

F. Tortorella © 2007
Università degli Studi di Cassino

SPEC CINT2006 Result
Copyright ©2006 Standard Performance Evaluation Corporation

Tyan SPECint2006 = --

Tyan Thunder K8E Tomcat (S2865) (AMD Opteron 146) SPECint_base2006 = 9.02

Test sponsor: Advanced Micro Devices | Tested by: Advanced Micro Devices
 CPU2006 license #: 491 | Test date: Apr-2006 | Hardware Availability: Aug-2005 | Software Availability: Apr-2006

Results Table

Benchmark	Base					Peak				
	Seconds	Ratio	Seconds	Ratio	Ratio	Seconds	Ratio	Seconds	Ratio	Ratio
400.perfbench	396	10.4	<u>82</u>	<u>10.4</u>	939	10.4				
401.bg2	<u>1185</u>	<u>8.32</u>	1182	8.16	1186	8.13				
403.gcc	890	8.12	<u>878</u>	<u>8.22</u>	878	8.23				
423.nat	1271	7.17	1269	7.19	<u>1228</u>	<u>7.28</u>				
445.golub	<u>807</u>	<u>11.6</u>	807	11.6	808	11.6				
456.kannr	<u>1184</u>	<u>7.88</u>	1181	7.90	1183	7.87				
478.sjeng	<u>1185</u>	<u>10.2</u>	1209	10.1	1183	10.2				
462.libquantum	<u>1482</u>	<u>14.4</u>	1433	14.5	1436	14.4				
464.spectral	1451	15.4	<u>1482</u>	<u>15.8</u>	1453	15.7				
471.mumps	821	7.10	<u>808</u>	<u>7.28</u>	831	7.02				
473.su3	<u>888</u>	<u>7.31</u>	883	7.11	890	7.09				
483.xalancbmk	<u>1312</u>	<u>2.34</u>	1315	2.27	1331	2.18				

Results appear in the order in which they were run. Bold underlined text indicates a medium measurement.

General Notes

Tested by Advanced Micro Devices
 The system can be assembled using an ATX power supply such as a Sparkle P8P310-60PN and a case such as a Chentec PML15

Base Compiler Invocation

C benchmarks:
gcc
 C++ benchmarks:
g++

Base Portability Flags

C benchmarks (except as noted below):
-DFPC_CPU_LP4
 400.perfbench -DFPC_CPU_LINUX_X64 -DFPC_CPU_LP4
 403.gcc -DFPC_CPU_LP4
 462.libquantum -DFPC_CPU_LINUX -DFPC_CPU_LP4

Continued on next page
 Standard Performance Evaluation Corporation
 info@spec.org
 http://www.spec.org

Page 2

Calcolatori Elettronici II
 Lezione 1 - 31

F. Tortorella © 2007
 Università degli Studi
 di Cassino