



Quad Core AMD Opteron™ Processor Overview

Brian Waldecker, Ph.D.
Senior Member of Technical Staff
AMD, Austin
4/14/2008

Talk Outline

1. Multi-Core Processor Architecture
2. FPU enhancements
3. Core IPC enhancements
4. Cache Hierarchy and Structure
5. TLBs and large pages
6. NB enhancements
7. Prefetching
8. Multi-core CPUs - configuring for performance
9. Some Recommended Programming Practices

Introducing "Barcelona"...

Native quad-core, 3rd Gen. AMD Opteron

Native Quad-Core Processor

To increase performance-per-watt efficiencies using the same Thermal Design Power.

Advanced Process Technology

65nm Silicon-on-Insulator Process

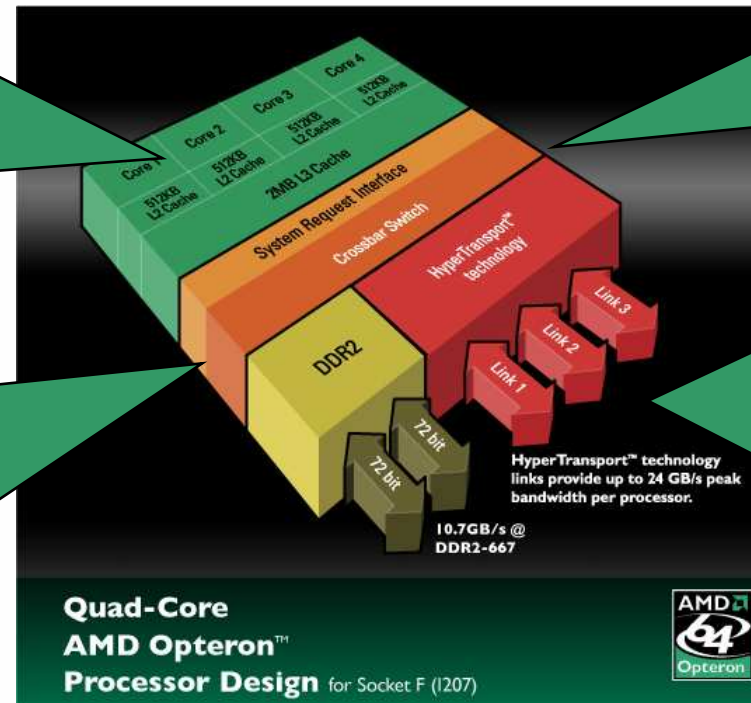
Fast transistors with low power leakage to reduce power and heat.

Platform Compatibility

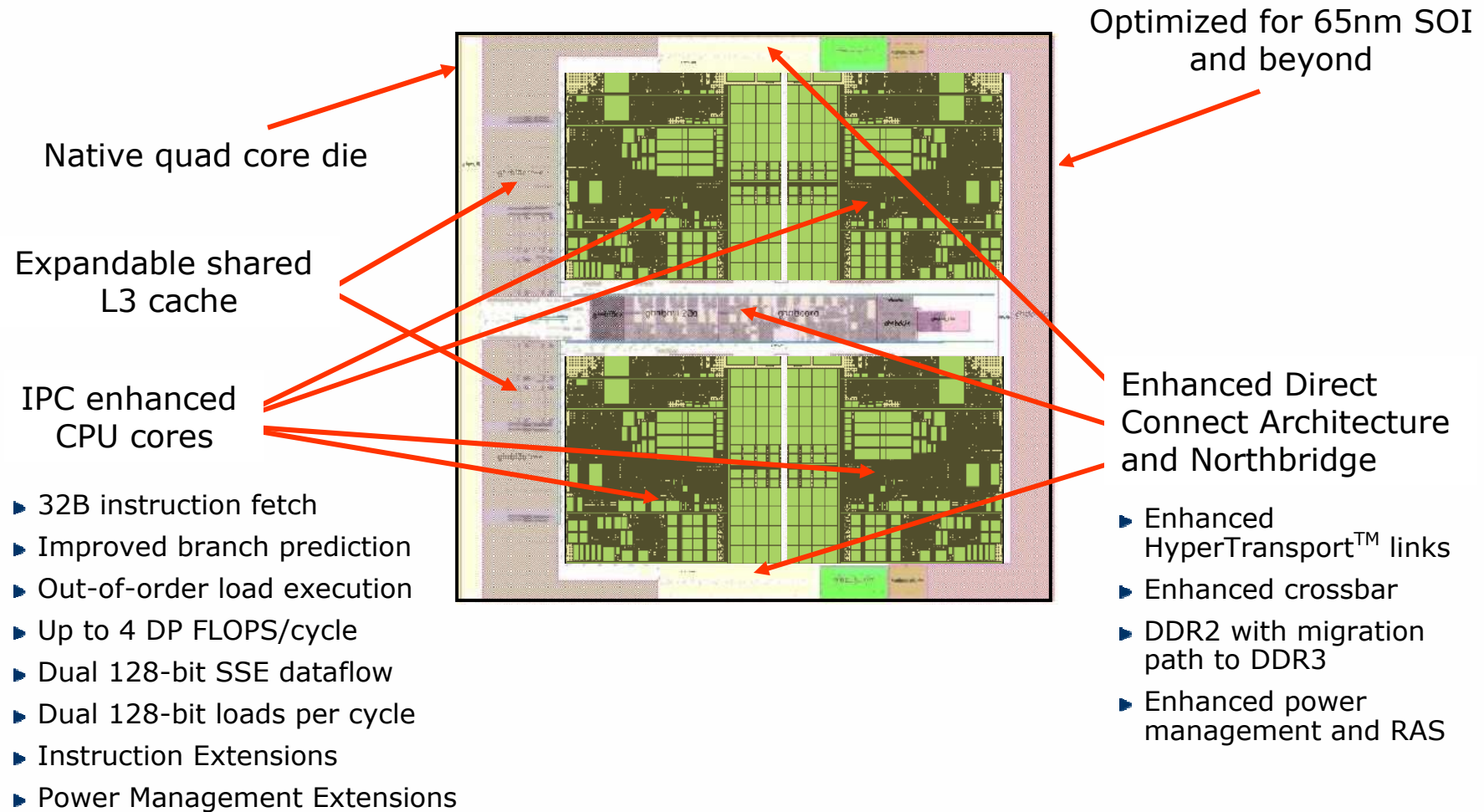
Socket and thermal compatible with "Socket F".

Direct Connect Architecture

- Integrated memory controller designed for reduced memory latency and increased performance
 - Memory directly connected
- Provides fast CPU-to-CPU communication
 - CPUs directly connected
- Glueless SMP up to 8 sockets



AMD's Next Generation Processor Technology



Comprehensive Upgrades for SSE128

128bit FPU



Parameter	Current Processor	"Barcelona"
SSE Exec Width	64	128 + SSE MOVs
Instruction Fetch Bandwidth	16 bytes/cycle	32 bytes/cycle + unaligned Ld-Ops
Data Cache Bandwidth	2 x 64bit loads/cycle	2 x 128bit loads/cycle
L2/NB Bandwidth	64 bits/cycle	128 bits/cycle
FP Scheduler Depth	36 Dedicated x 64-bit ops	36 Dedicated x 128-bit ops

Can perform SSE MOVs in the FP "store" pipe

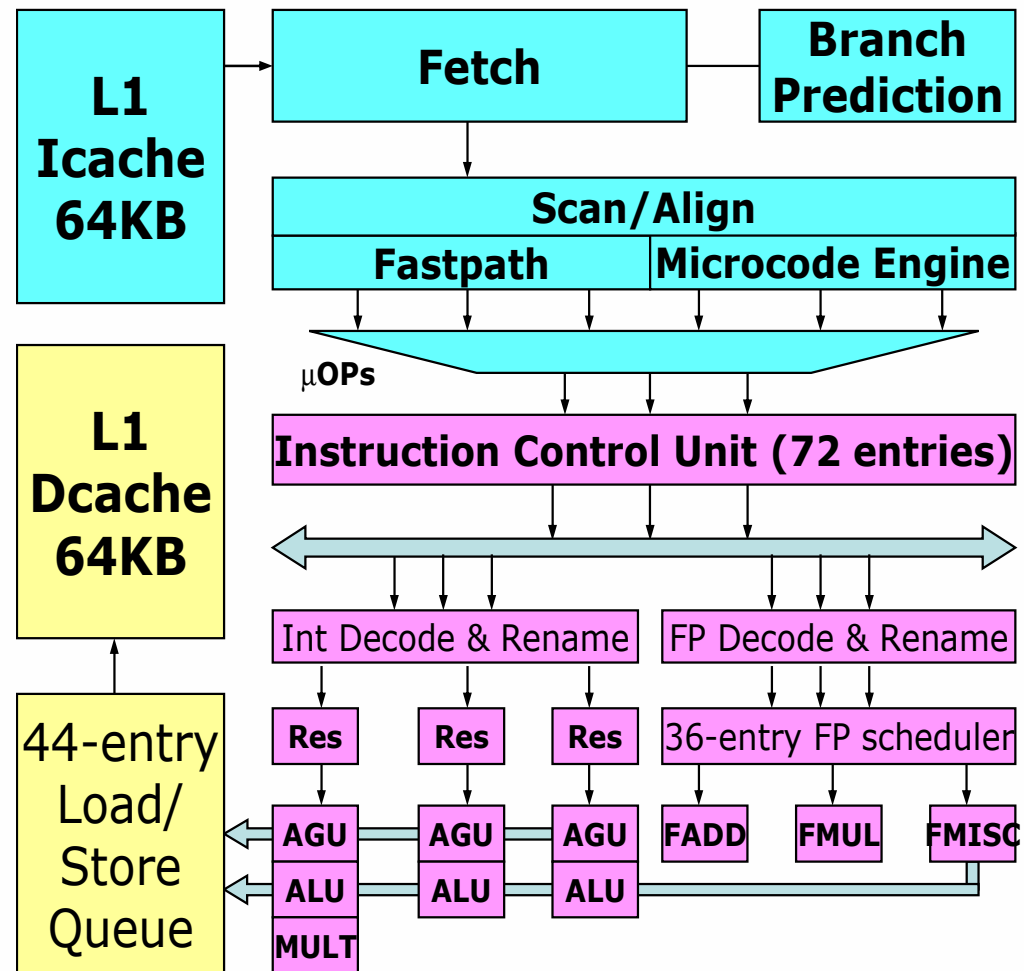
- Execute two generic SSE ops + SSE MOV each cycle (+ two 128-bit SSE loads)

SSE Unaligned Load-Execute mode

- Remove alignment requirements for SSE ld-op instructions
- Eliminate awkward pairs of separate load and compute instructions
- *To improve instruction packing and decoding efficiency*

Core IPC improvements

- Improve Branch Prediction.
- TLB enhancements.
- More out of order Ld/St capability.
- New Instructions
 - POPCNT / LZCNT
 - EXTRQ / INSERTQ
 - MOVNTSD / MOVNTSS
- Fastpath support for FP to Integer data movement.



Cache Hierarchy

Dedicated L1 cache

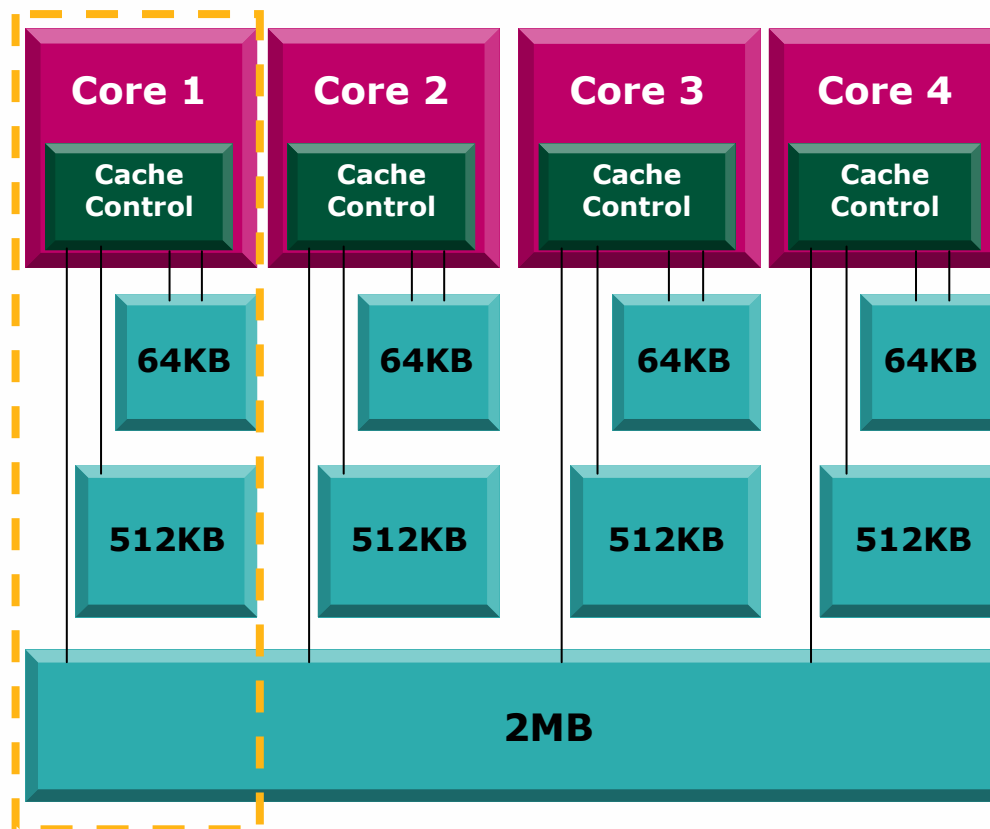
- 2 way associativity.
- 8 banks.
- 2 128bit loads per cycle.

Dedicated L2 cache

- 16 way associativity.

Shared L3 cache

- fills from L3 leave likely shared lines in L3.
- sharing aware replacement policy.



TLB Enhancements

- Support for 1GB pagesize (4k, 2M, 1G)
- 48 bit physical addresses = 256TB (increase from previous 40bits)
- Data TLB
 - L1 Data TLB
 - 48 entries, fully associative
 - all 48 entries support any pagesize
 - L2 TLB
 - 512 4k entries, or
 - 128 2M entries
- Instruction TLB
 - L1 Instruction TLB
 - fully associative
 - support for 4k or 2M pagesizes
 - L2 Instruction TLB

Memory Controller Enhancements

- Independent (unganged mode) DRAM controllers
 - allow more concurrent reads of needed data.
 - reduce bank conflicts.
 - longer burst length = better command efficiency.
- Optimized DRAM paging
 - adaptive closing of DRAM pages = more page hits.
- Re-architect Northbridge for higher BW
 - resize buffers, better command scheduling, DDR2 and beyond.
- Write bursting
 - reduce read-write turnarounds.

Data Prefetch

- **Hardware prefetching**

- DRAM prefetcher
 - tracks positive, negative, non-unit strides.
 - dedicated buffer (in NB) to hold prefetched data.
 - Aggressively use idle DRAM cycles.
- Core prefetchers
 - Does hardware prefetching into L1 Dcache.

- **Software prefetching instructions**

- MOV (prefetch via load / store)
- prefetcht0, prefetcht1, prefetcht2 (currently all treated the same)
- prefetchw = prefetch with intent to modify
- prefetchnta = prefetch non-temporal (favor for replacement)

A Few Programming Hints

Which Prefetch to use ?

Data	Less than ½ L1 size	Less than ½ L2 size or of unknown size		Greater than ½ L2 size
		Reused	Not Reused	
Read only	prefetch or prefetchnta	prefetch	prefetchnta	prefetchnta
Sequential read only	hwprefetcher + prefetch	hwprefetcher + prefetch	prefetchnta	prefetchnta
Read-write	prefetchw	prefetchw	prefetchnta	prefetchnta
Sequential read-write	prefetchw	prefetchw	prefetchnta	prefetchnta
Write only	prefetchw	prefetchw	movnt	movnt
Sequential write only	hwprefetcher + prefetchw	hwprefetcher + prefetchw	movnt	movnt

A Few Programming Hints

- Use SSE2 instructions that modify entire 128bit SSE register instead of preserving one half.
- Generally good to prefetch 6 to 8 cachelines ahead
 - Latency-Bandwidth product estimates how much data must be “in-flight”
 - 1P, DDR2-800 $\approx 53\text{ns} * 10\text{GB/s} = 530 \text{ Bytes} = \sim 8 \text{ cache lines in flight.}$
 - 2P, DDR2-667 $\approx 81\text{ns} * 17\text{GB/s} = 1377 \text{ Bytes} = \sim 21 \text{ cache lines in flight (combined across both Northbridges).}$
- Try to have 100 cycles of computation in loop body between successive prefetches
- Avoid issuing multiple software prefetches to the same cacheline
- Unroll loops enough times so each iteration works on 1 or more cachelines of data.

note: neither hw or sw prefetches will be allowed to generate page faults, but a TLB miss on a prefetch can initiate a TLB fill.

A Few Programming Hints

L3 Cache Data Sharing

Optimizing Producer Consumer ring buffers

1. Data gets into L3 by spilling from L1+L2.
2. Consumer must “lag” producer by at least L1+L2 bytes so data is in L3 when consumed. (artifact of victim cache design).
3. Producer must not get too far ahead or it will flood L3 and evict unconsumed lines.
4. Producer must lag consumer by at least L1+L2 bytes so data is written into L3 with minimal collision with Consumer. (MOESI artifact).
5. Consumer should use PrefetchW* to prefetch data.
6. Ring Buffer should be at least 2 x (L1+L2) in size, plus some cushion.
7. Remember, to account for L3 size per Producer-Consumer pair. (when running 1 pair vs. 2 pairs per chip).

* PrefetchW brings data into cache in M state. Otherwise data might come in as S state and L3 retains a copy in O state.

MULTI-CORE PERFORMANCE EXAMPLES

Multi-Core Performance Examples 1&2



Opteron™ (Barcelona) System

Asus KFSN4-DRE

Two Opteron(R) 8356 CPUs @ 2.3GHz

8 x 2GB DDR2-667

SLES10 SP1 X86_64

PGI Fortran compiler for Linux 7.2

PGI C/C++ compiler for Linux 7.2

Compiler flags (all cases): -O3 -fastsse -tp=barcelona-64 -mcmmodel=medium

Intel® Xeon® (Harpertown) System 1

Supermicro X7DWN+ server board

Two Intel(R) Xeon(R) E5472 CPUs @ 3.00GHz

8 x 2GB DDR2-800 FBDIMM

SLES10 SP1 X86_64

Intel Fortran compiler for Linux 10.1

Intel C/C++ compiler for Linux 10.1

Compiler flags (all cases): -fast

Intel® Xeon® (Harpertown) System 2

Supermicro X7DWN+ server board

Two Intel(R) Xeon(R) E5410 CPUs @ 2.33GHz

8 x 2GB DDR2-800 FBDIMM

SLES10 SP1 X86_64

Intel Fortran compiler for Linux 10.1

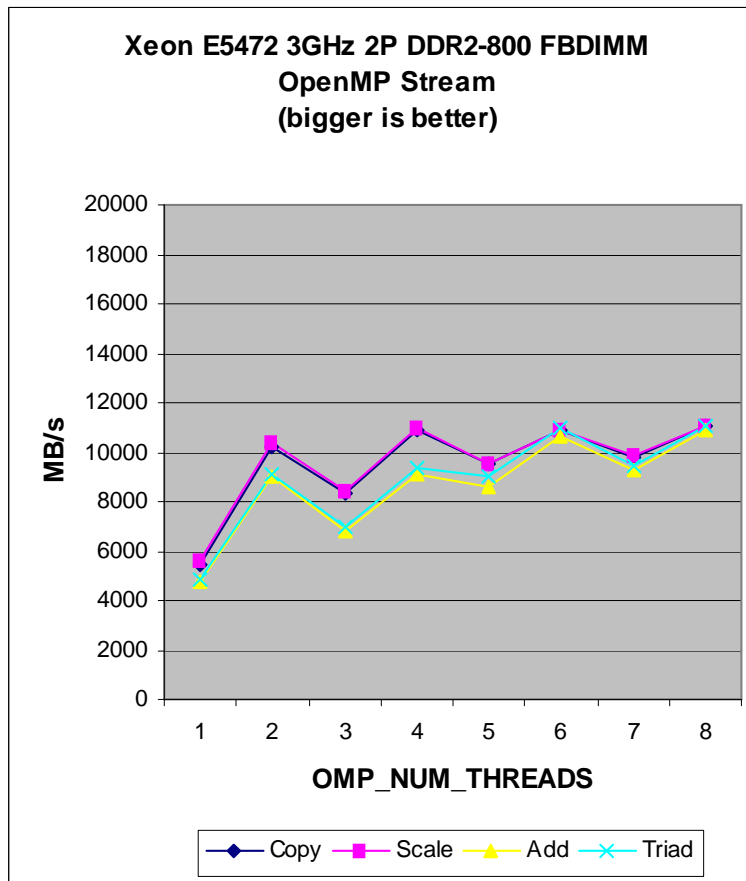
Intel C/C++ compiler for Linux 10.1

Compiler flags (all cases): -fast

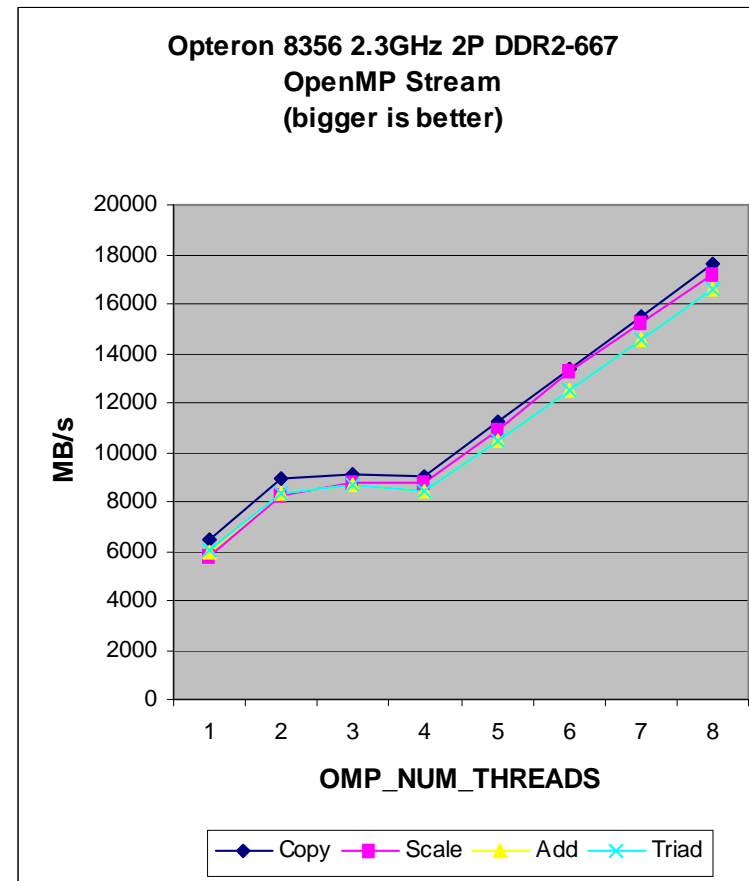
Multi-Core Performance Example1

OpenMP STREAM

(note logical cpu to physical core mapping behavior)



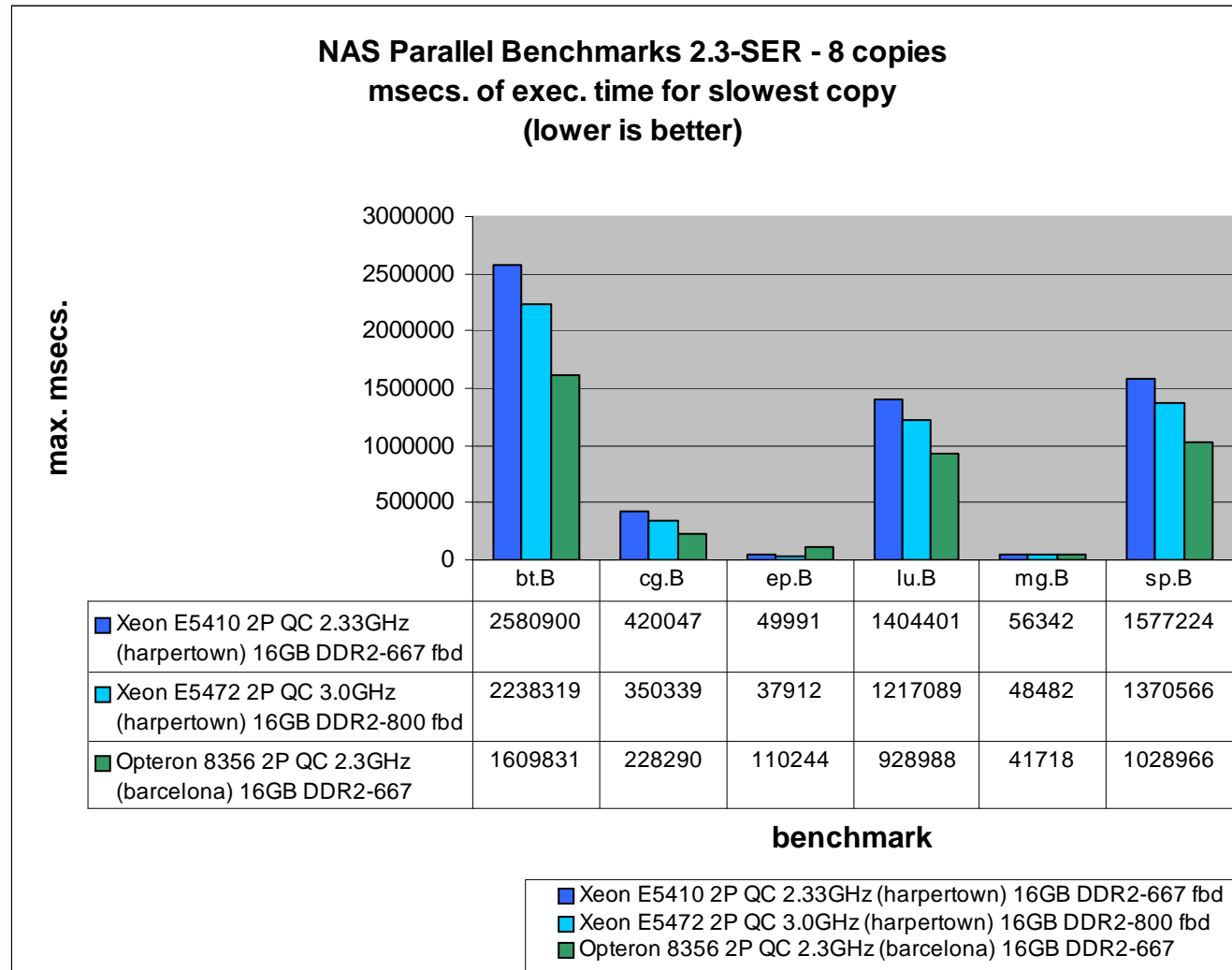
“socket-major”



“core-major”

Multi-Core Performance Example2

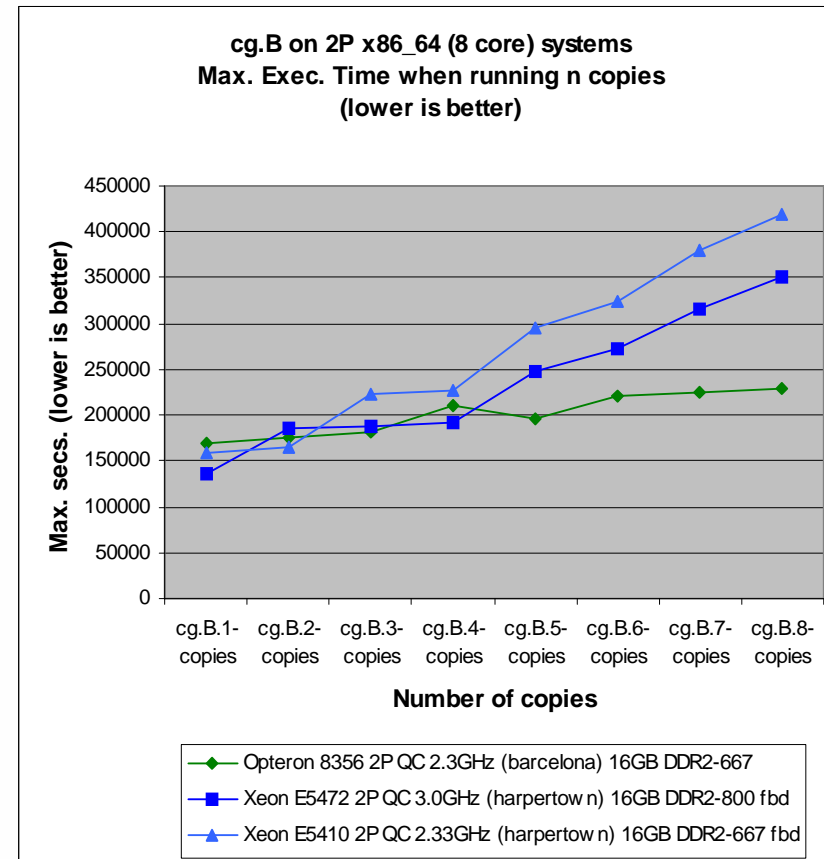
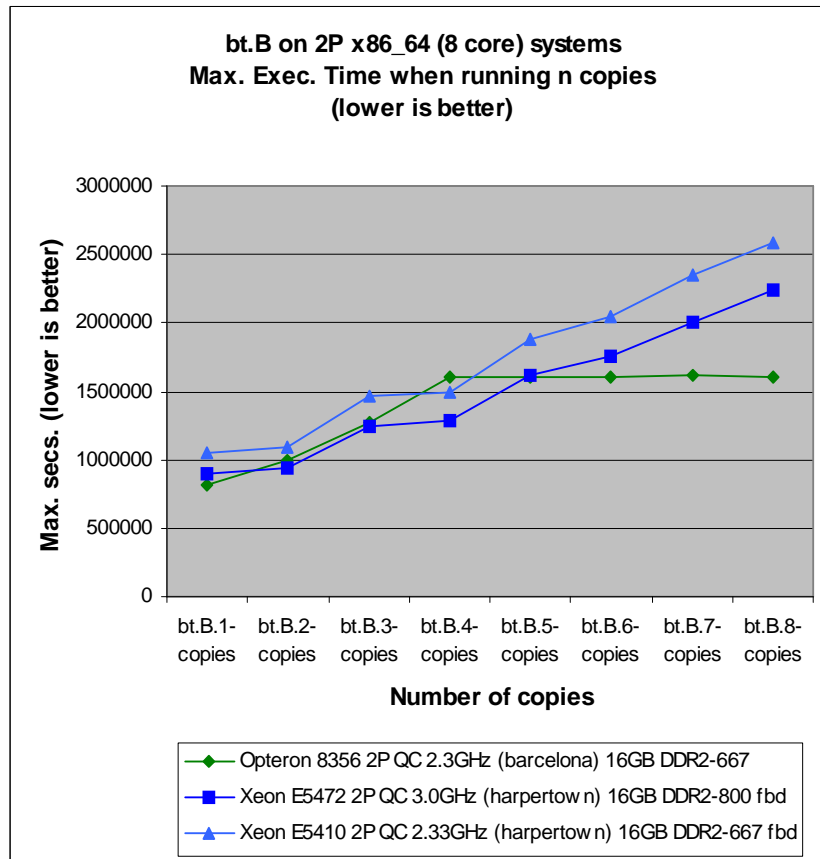
NPB2.3-SER Multiple Copies



Note: ep.B is cache contained with 3MB L2 per core, but not for 1MB L2+L3 per core

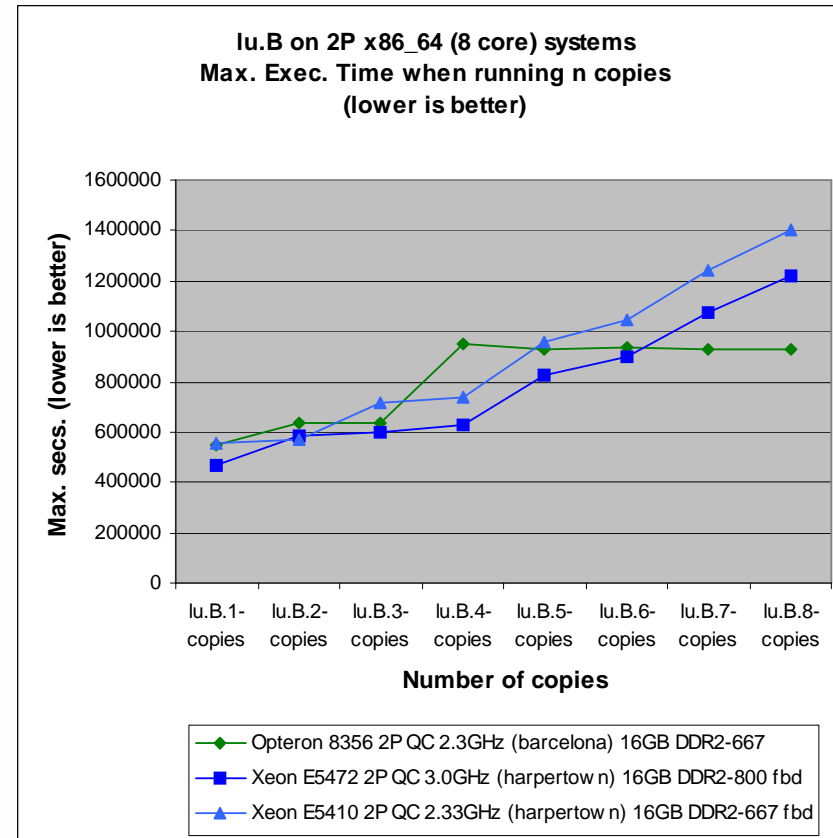
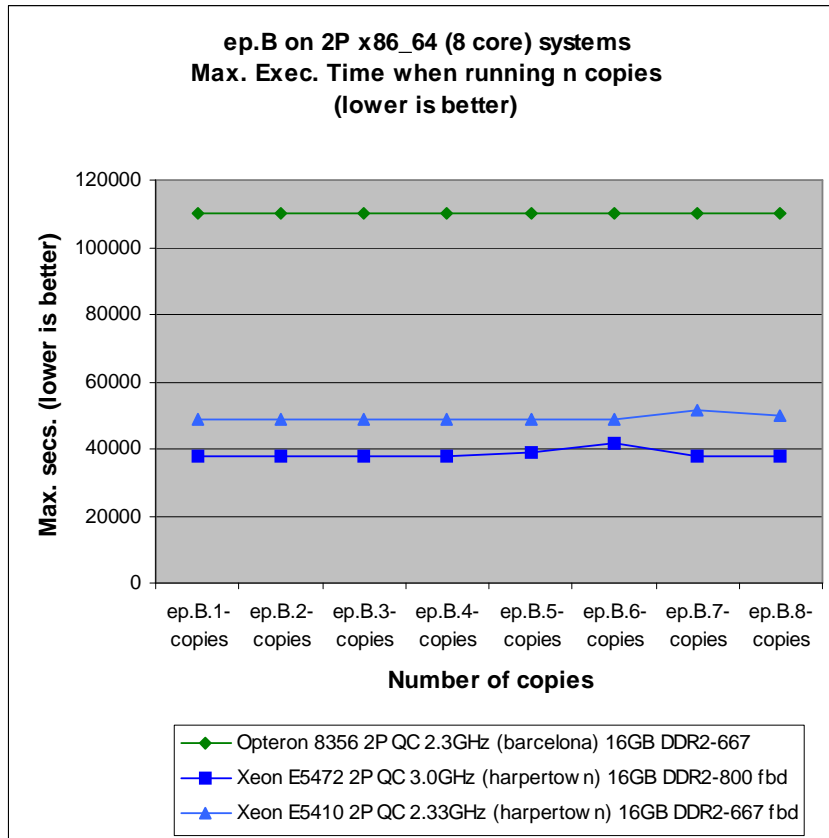
Multi-Core Performance Example2

NPB2.3-SER Multiple Copies



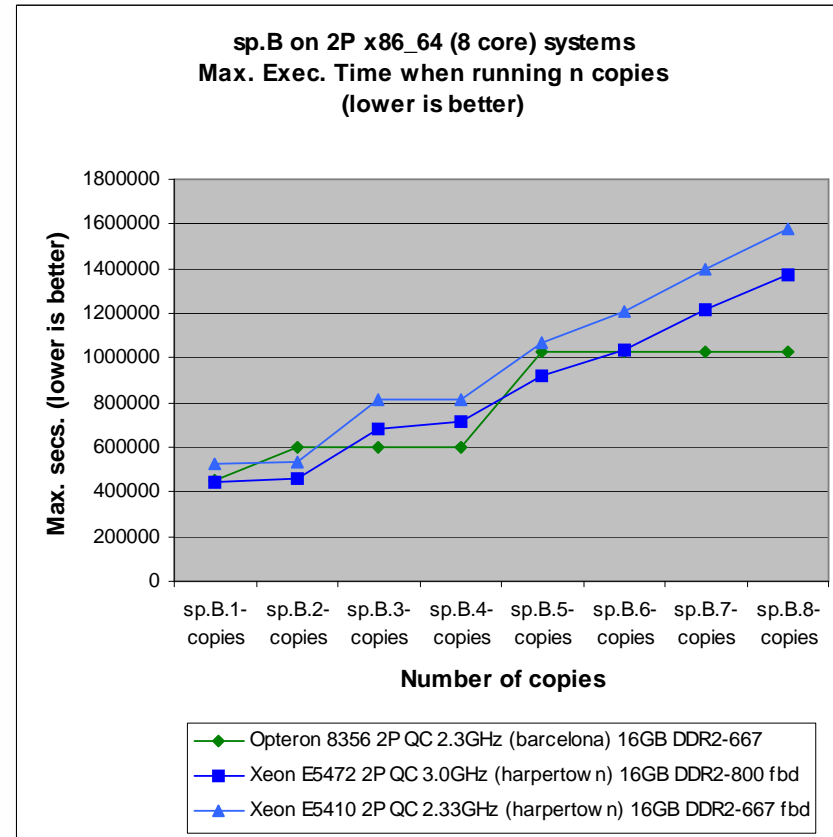
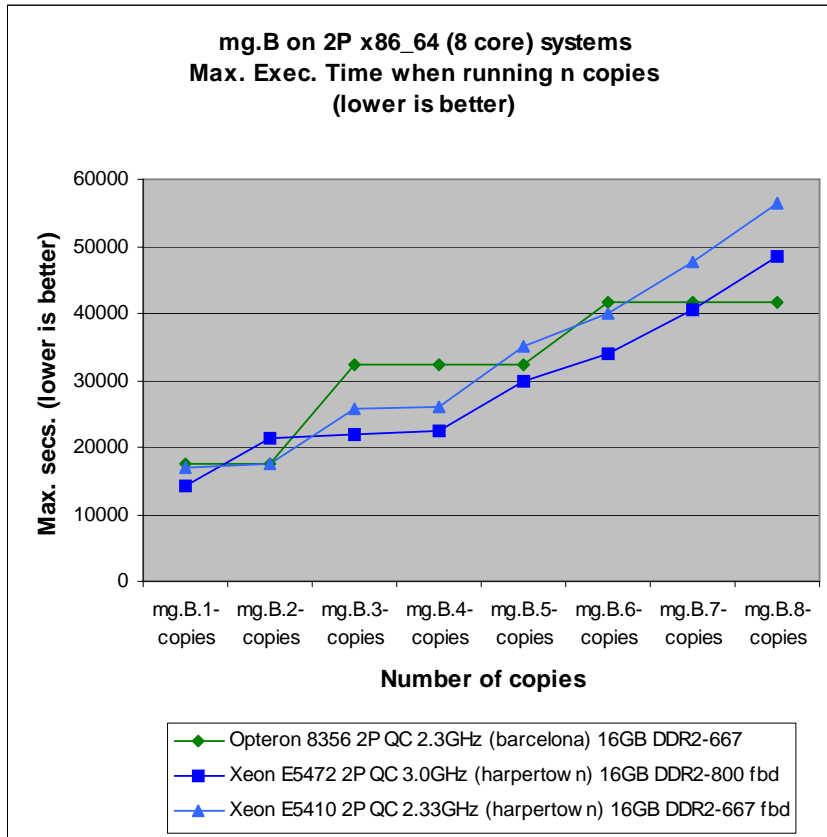
Multi-Core Performance Example2

NPB2.3-SER Multiple Copies



Multi-Core Performance Example2

NPB2.3-SER Multiple Copies



Multi-core Performance Example3

SPEC OMPL2001 (SPEC-HPG OpenMP benchmark)
313.swim_I (shallow water ocean model)

Opteron™ (Barcelona) System

Tyan Thunder n425QE (S4985E)
Four Opteron 8356 CPUs @ 2.3GHz
16 x 2GB DDR2-667
SLES10 SP1 X86_64
PathScale Compiler Suite 3.1

Multi-core Performance Example3

313.swim_l (shallow water modeling, large dataset)



Three sets of compiler* flags used:

"Ofast"

-mp -Ofast -mcpu=barcelona -OPT:early_mp=on -mcmmodel=medium

"Ofast_simd0"

-mp -Ofast -mcpu=barcelona -OPT:early_mp=on **-LNO:simd=0** -mcmmodel=medium

"Ofast_movnti2500"

-mp -Ofast -mcpu=barcelona -OPT:early_mp=on **-CG:movnti=2500** -mcmmodel=medium

Flags	Runtime in secs. (mins.)
Ofast	7194s (120m)
Ofast_simd0	1736s (29m)
Ofast_movnti2500	1785s (30m)

Too much of a good thing? (streaming stores or vectorization)

* Pathscale™ Compiler Suite, Version 3.1, SLES10 SP1

Multi-core Performance Example3

313.swim_l (shallow water modeling, large dataset)



Profiling shows

Problem Size:
 7701 x 7701 grid, REAL*8
 452MB per array (5.8GB total)

Program Structure:
 10 NCYCLE=NCYCLE+1
 Calc1 (writes 4 arrays)
 ...
 Calc2 (writes 3 arrays)
 ...
 Calc3 (writes 6 arrays) ←
 ...
 GOTO 10

Thrashing WCPU!

Ofast

samples	%	symbol name
598797565	79.1052	__ompdo_calc3_1
90931114	12.0126	__ompreregion_calc2_1
48912887	6.4617	__ompreregion_calc1_1
17952271	2.3716	__ompdo_MAIN__1
153331	0.0203	__ompdo_calc3z_1

Ofast simd0

samples	%	symbol name
68233066	34.4688	__ompreregion_calc2_1
61859772	31.2492	__ompdo_calc3_1
48931003	24.7181	__ompreregion_calc1_1
18617176	9.4047	__ompdo_MAIN__1
132326	0.0668	__ompdo_calc3_2

Ofast movnti2500

samples	%	symbol name
68314854	34.4881	__ompreregion_calc2_1
61749164	31.1735	__ompdo_calc3_1
48932509	24.7031	__ompreregion_calc1_1
18770527	9.4761	__ompdo_MAIN__1
132286	0.0668	__ompdo_calc3_2

Oprofile: Counted CPU_CLK_UNHALTED events (Cycles outside of halt state)

Multi-core Performance Example3

313.swim_l (shallow water modeling, large dataset)



Performance Counters

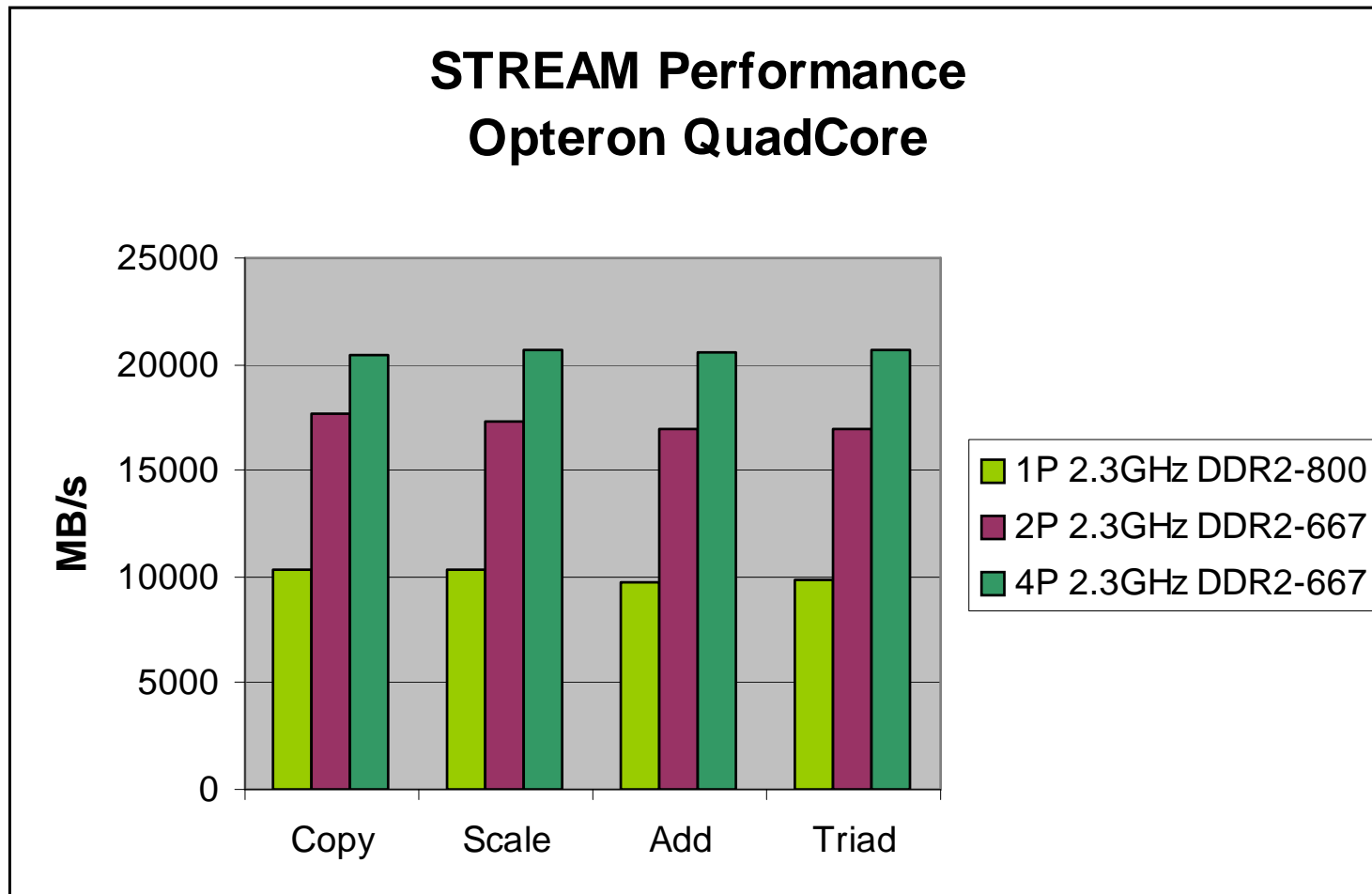
Ofast = default
Ofast_simd0 = -LNO:simd=0
Ofast_movnti2500 = -CG:movnti=2500

	default	"-LNO:simd=0"	"-CG:movnti=2500"
CPI	14	2.55	3.7
Clocks(B)	15373	4107	4146
Insts(B)	1100	1618	1096
L3Req	440.000	161.800	142.480
L3Miss	330.000	134.294	134.808
totSSE	792.000	1164.960	789.120
FPadd pipe	583.000	388.320	252.080
absolute(B) FPmult pipe	242.000	210.340	151.248
FPstore pipe	291.500	142.384	113.984
PgOpen	253.000	37.214	36.168
PgClose	233.200	114.878	111.792
PgCflct	114.400	63.102	61.376

* Pathscale™ Compiler Suite, Version 3.1, SLES10 SP1

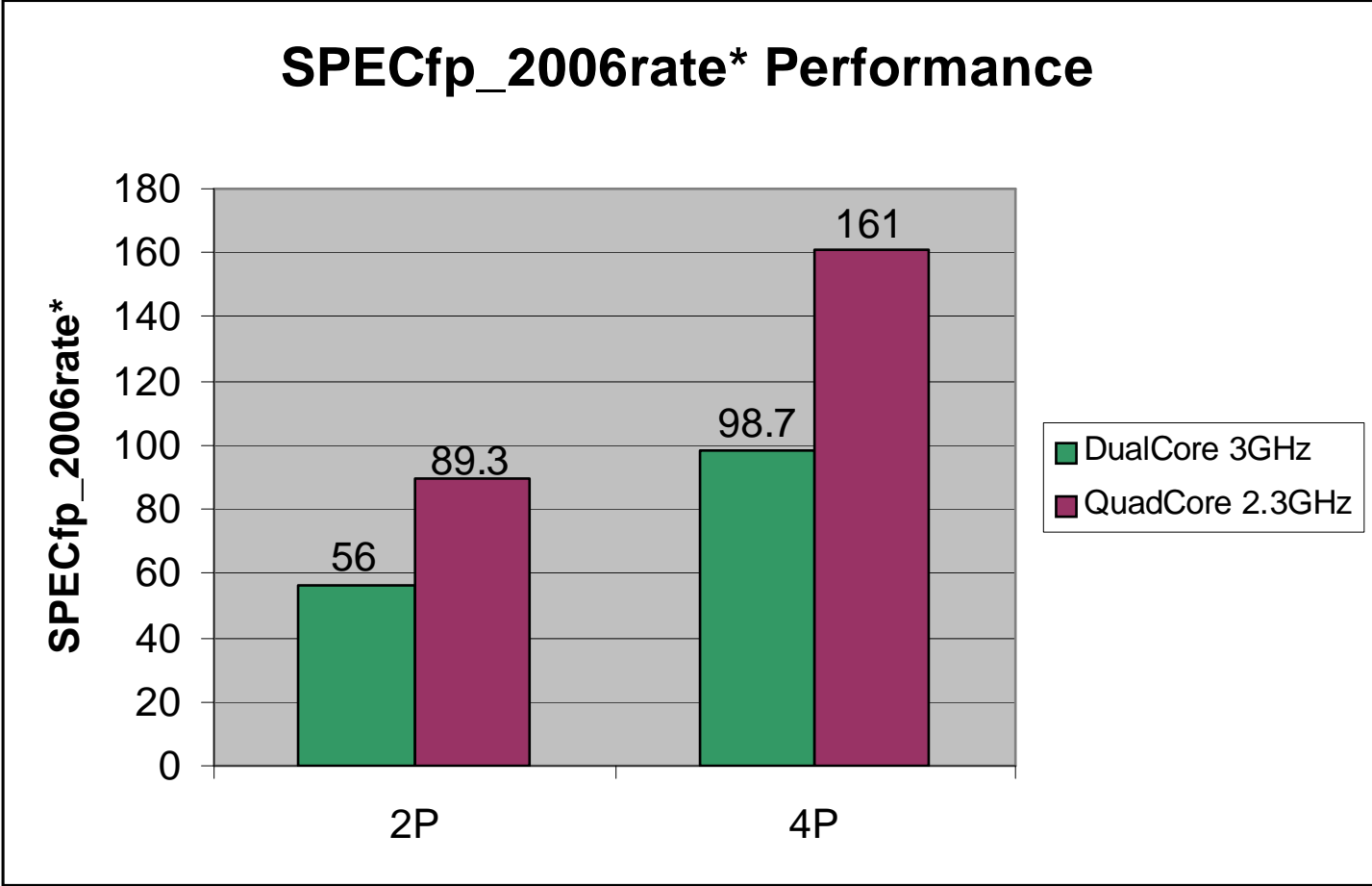
PERFORMANCE RESULTS

Performance of Multi-core: STREAM



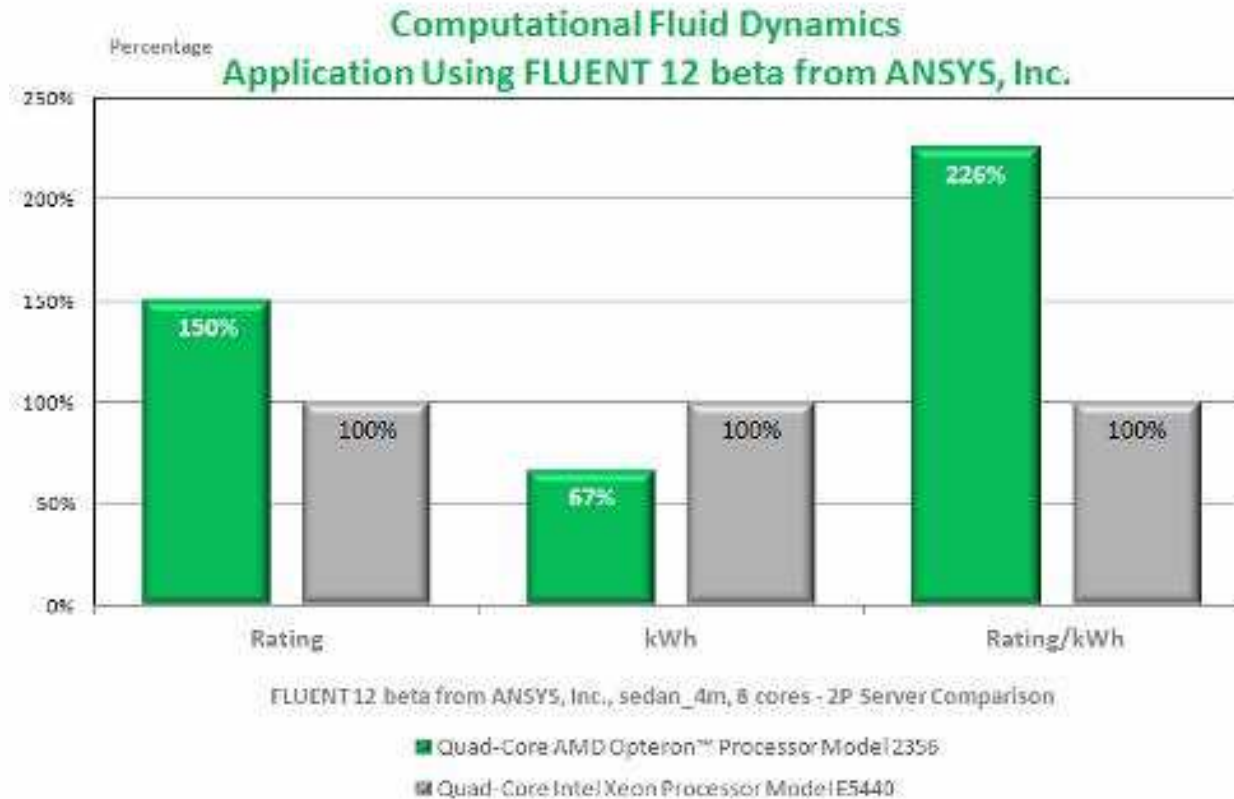
Pathscale™ Compiler Suite, Version 3.1, SLES10 SP1

Performance of Multi-core SPECfp 2006rate*



* SPEC and SPECfp are registered trademarks of the Standard Performance Evaluation Corporation. For the latest SPECfp_rate2006 results, visit <http://www.spec.org/cpu2006/results/>

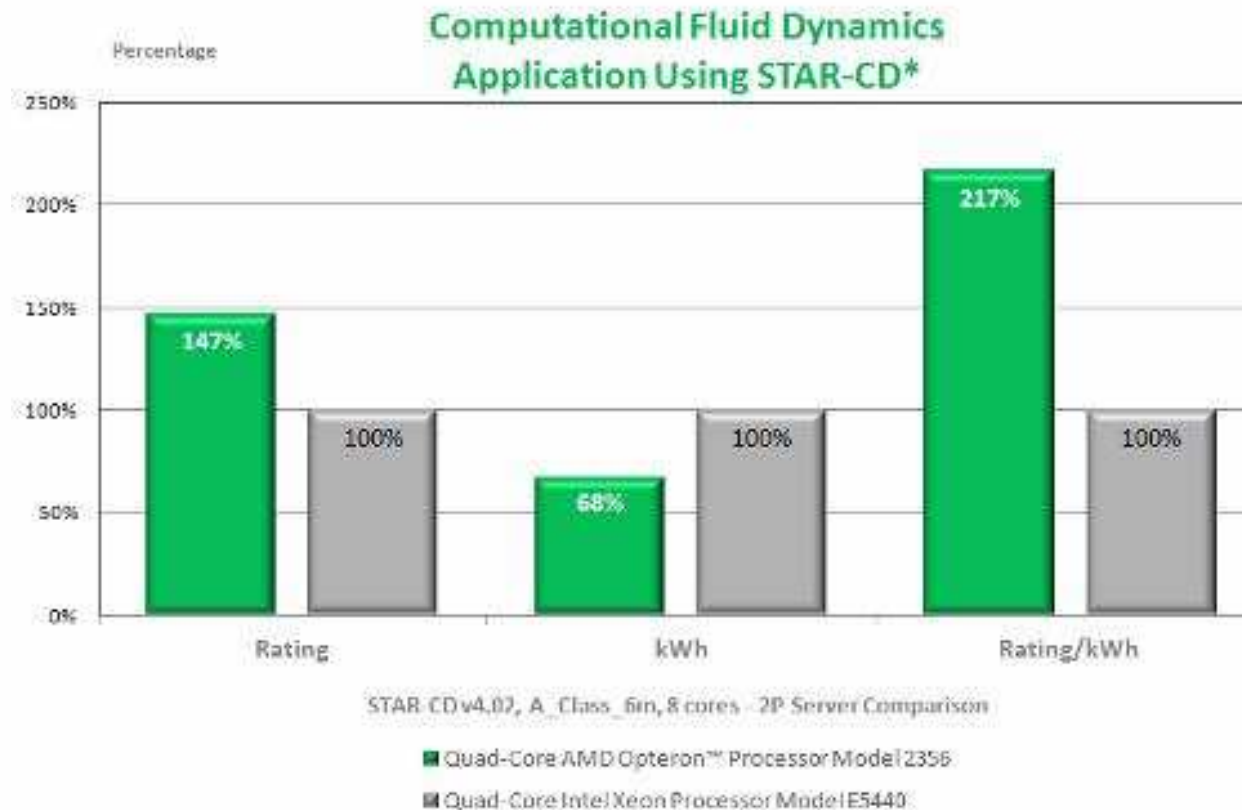
Performance of Multi-core: CFD, 2P Servers



Configuration Info:

1. 2 x Quad-Core AMD Opteron™ processors Model 2356 in Supermicro H8DMU+ motherboard, 16GB (8x2GB DDR2 memory), 150GB IDE disk drive, SuSE Linux® Enterprise Server 10 SP1 64-bit
2. 2 x Quad-Core Intel Xeon processors Model E5440 in Supermicro X7DWN+ motherboard, 16GB (8x2GB FBDimm memory), 150GB IDE disk drive, SuSE Linux Enterprise Server 10 SP1 64-bit

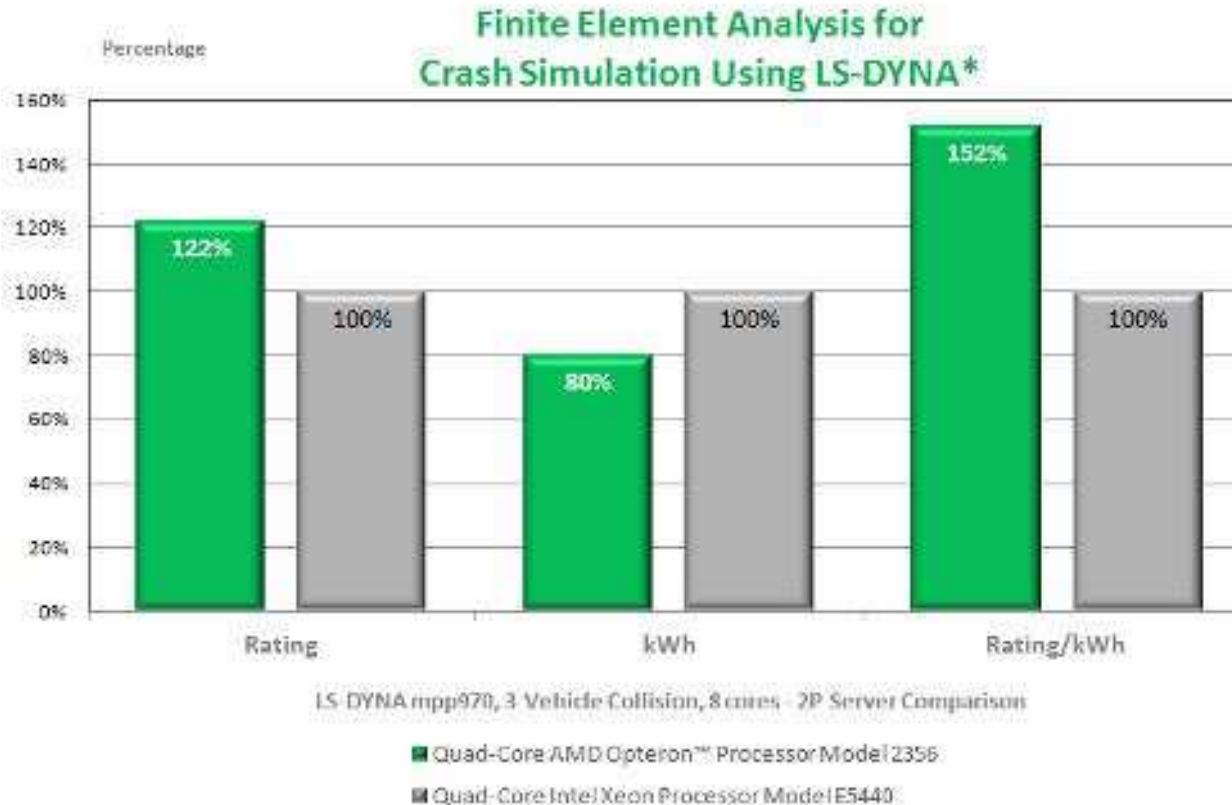
Performance of Multi-core: CFD, 2P Servers



Configuration Info:

1. 2 x Quad-Core AMD Opteron™ processors Model 2356 in Supermicro H8DMU+ motherboard, 16GB (8x2GB DDR2 memory), 150GB IDE disk drive, SuSE Linux® Enterprise Server 10 SP1 64-bit
2. 2 x Quad-Core Intel Xeon processors Model E5440 in Supermicro X7DWN+ motherboard, 16GB (8x2GB FBDimm memory), 150GB IDE disk drive, SuSE Linux Enterprise Server 10 SP1 64-bit

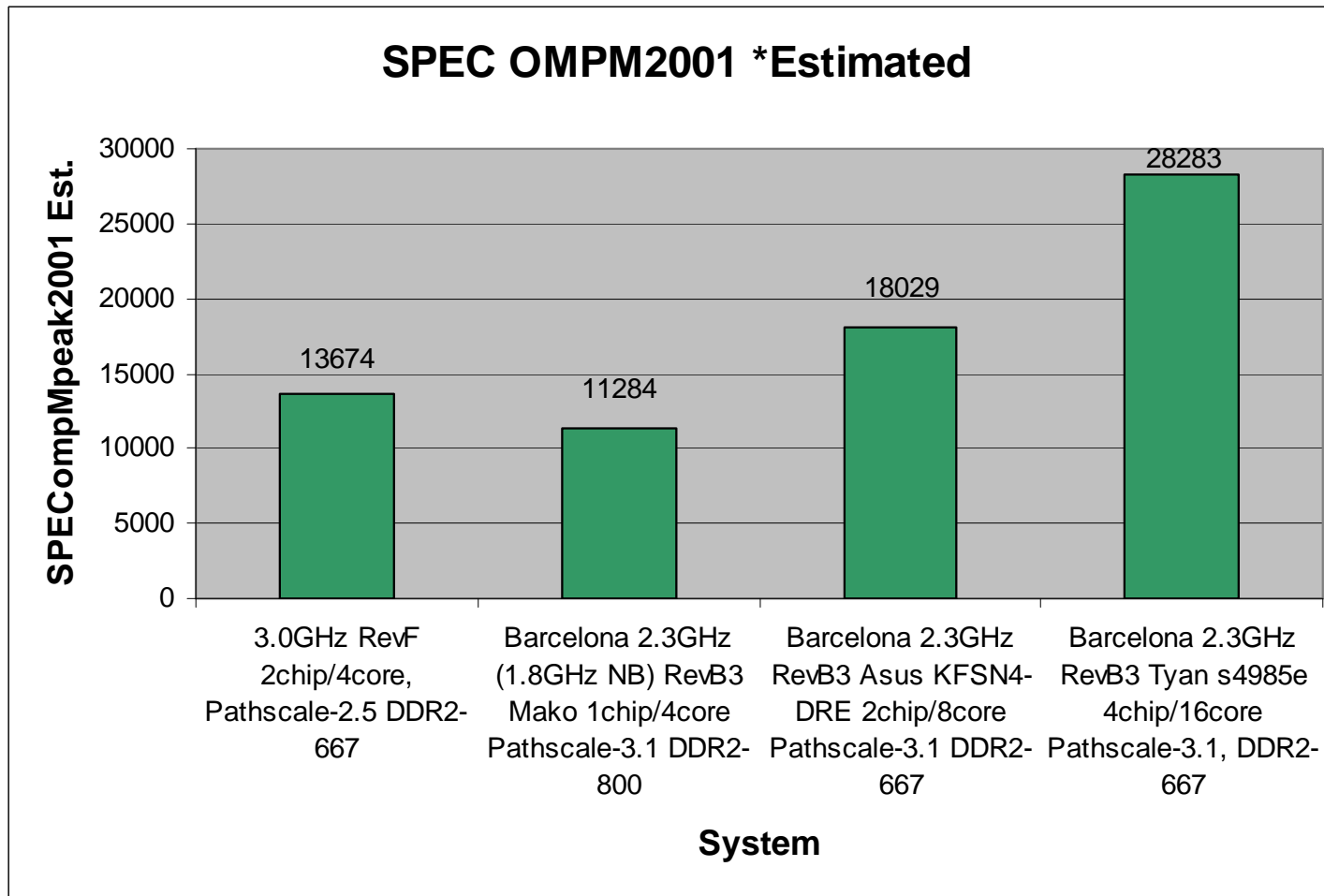
Performance of Multi-core: Finite Element Code, 2P Servers



Configuration Info:

1. 2 x Quad-Core AMD Opteron™ processors Model 2356 in Supermicro H8DMU+ motherboard, 16GB (8x2GB DDR2 memory), 150GB IDE disk drive, SuSE Linux® Enterprise Server 10 SP1 64-bit
2. 2 x Quad-Core Intel Xeon processors Model E5440 in Supermicro X7DWN+ motherboard, 16GB (8x2GB FBDimm memory), 150GB IDE disk drive, SuSE Linux Enterprise Server 10 SP1 64-bit

Performance of Multi-core SPEC OMPM2001 (SPEC-HPG OpenMP benchmark)

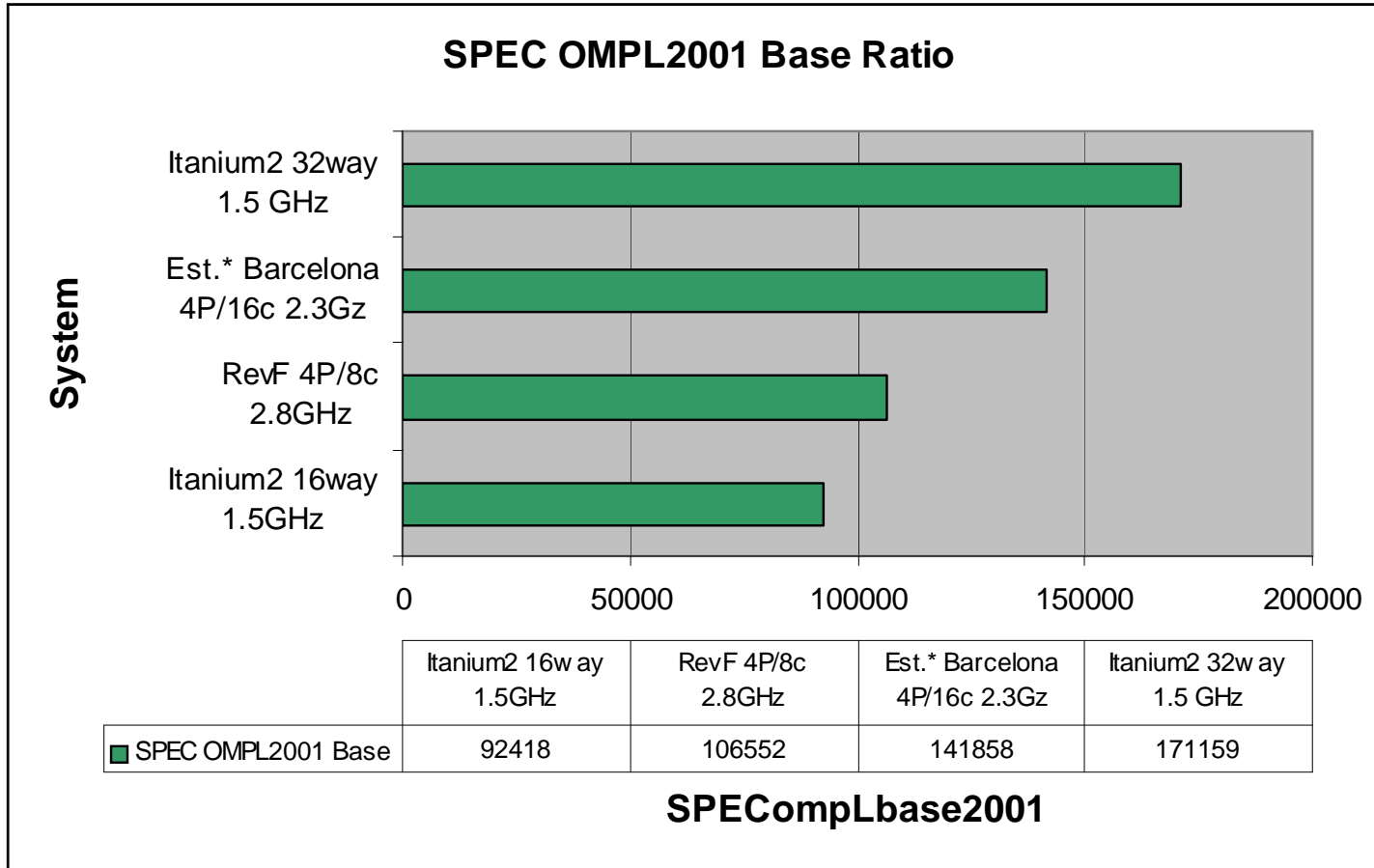


Pathscale™ Compiler Suite, Version 3.1, SLES10 SP1

C: pathcc -gnu3 -mp -Ofast -march=barcelona -mcmmodel=medium

Fortran: pathf90 -mp -Ofast -march=barcelona -OPT:early_mp=on -mcmmodel=medium

Performance of Multi-core SPEC OMPL2001 (SPEC-HPG OpenMP benchmark)



5 "CPU" years
of changes

Trivia:

- 1st Gen. Opteron launched in Spring 2003.
- Itanium2 systems are circa Fall 2003.

Pathscale™ Compiler Suite, Version 3.1, SLES10 SP1
 C: pathcc -gnu3 -mp -Ofast -march=barcelona -mcmmodel=medium
 Fortran: pathf90 -mp -Ofast -mcpu=barcelona -OPT:early_mp=on -CG:movnti=2500 -mcmmodel=medium

Summary

Attention paid throughout to improving:

- Core IPC.
- Caches and TLB.
- FPU performance.
- Memory performance.
- Compatibility, Usability, and Programmability.

Multi-Core Implications:

- Flop rich environment.
- Process to Core Mappings – Shared L3 data sharing.

References

- AMD Opteron Processor Families Technical Documentation
 - http://www.amd.com/us-en/Processors/TechnicalResources/0,,30_182_739_9003,00.html
- *Bios and Kernel Developers Guide (BKDG)* on AMD website.
 - RevF cpus are denoted as “Family 0Fh”.
 - Quadcore BKDG is “Family 10h”.
 - Portions useful to others than just Bios and Kernel developers.
- *Software Optimization Guide for AMD64 Processors*
 - Quadcore version available on AMD website.

Trademark Attribution

AMD, the AMD Arrow logo, AMD Opteron, and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. HyperTransport is a licensed trademark of the HyperTransport Technology Consortium. Linux is a registered trademark of Linus Torvalds. Other names used in this presentation are for identification purposes only and may be trademarks of their respective owners.

©2008 Advanced Micro Devices, Inc. All rights reserved.