

Performance Characteristics of Dual-Processor HPC Cluster Nodes Based on 64-bit Commodity Processors

A. Purkayastha, Chona S. Guiang, K. Schulz,
T. Minyard, K. Milfeld, B. Barth, P. Hurley,
J. Boisseau

Texas Advanced Computing Center
The University of Texas at Austin



THE UNIVERSITY OF TEXAS AT AUSTIN
Texas Advanced Computing Center



Outline

1. Need for 64-bit computing
2. 64-bit systems evaluated in this study
3. Micro-benchmarks
4. Application benchmarks
5. Summary and observations



Why 64-bit computing?

- Larger memory addressability (for overcoming 2GB/process limit)
- Larger filespace addressability (although Linux has large-file support)
- Native support for 64-bit integer arithmetic



Potential advantages/disadvantages of 64-bit computing

- Larger memory addressability can benefit SIMD codes that are not scalable; jobs can be spread over a smaller number of processors
- x86-64 has twice the number and size of GPRs as x86

- 64-bit executables are bigger
- Bigger datatypes and instruction size occupy larger cache footprints
- Bigger datatypes and instruction size consume more bandwidth



32-bit to 64-bit migration issues

- In C, change from ILP32 to LP64 data model. Need to pay attention to
 - assignment of `pointers` or `long` to `int` (data truncation)
 - interchangeable use of `int`, `long` and `pointers` (performance degradation due to implicit casts)
 - `struct` size and alignment may change
- May need to maintain separate source trees
- Cannot mix 32-bit and 64-bit objects, need to recompile separate 32-bit and 64-bit versions
- Maintain two different sets of applications and libraries



64-bit Systems

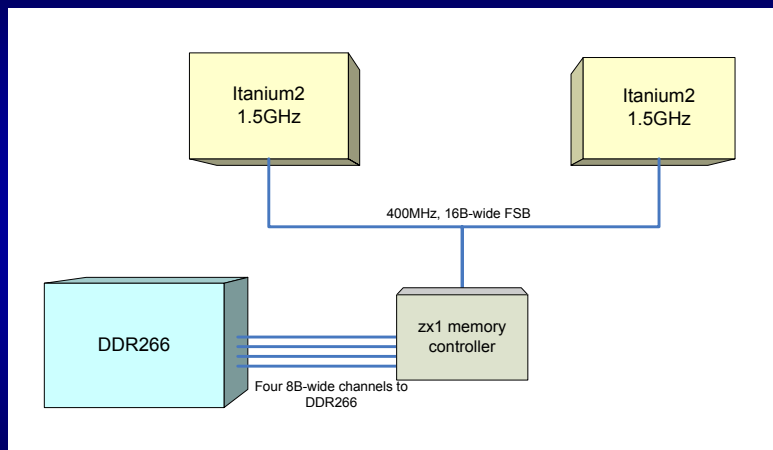
1. HP Integrity rx2600
2. IBM e325 Model 246
3. Apple Power Mac G5



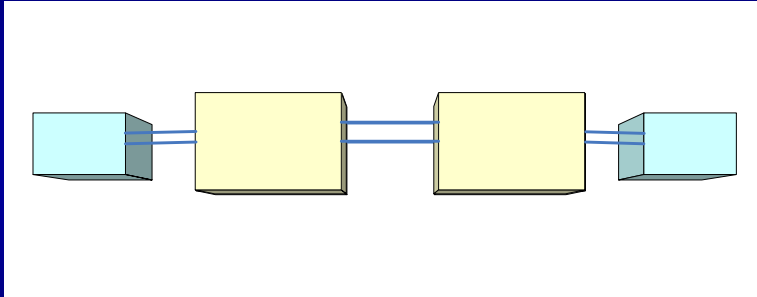
	HP Integrity rx2600	IBM e325 Model 246	Apple Power Mac G5
Processor	Intel Itanium 2	AMD Opteron	Apple G5
Processor Speed	1.5GHz	2.0GHz	2.0GHz
Floating Point	2 FMAs	2 Units (separate multiply and add units)	2 FMAs
SIMD		SSE2	AltiVec—128bit
Prefetch	Hardware prefetch	Hardware prefetch (8 streams)	Hardware prefetch (8 streams)
Architecture	EPIC (64-bit)	x86-64	PowerPC
Peak 64-bit FLOPS/CP (clock period)	4 FLOPS/CP	2 FLOPS/CP	4 FLOPS/CP
Chipset	HP zx1	8111/8131	
Cache	L3 6MB; L2 256K; L1 16K/16K data/instr.	L2 1.0MB L1 64KB	L2 512KB L1 32K/32K data/instr.
Peak aggregate bus bandwidth	6.4GB/s 400MHz 16B-wide shared frontside bus (FSB)	10.6GB/s Each processor has dual 8B-wide channels to memory	16GB/s Each processor has a dedicated FSB, each 1GHz FSB has two 4B-wide unidirectional links
Memory bandwidth	four channels to DDR266 at 8B = 8.5GB/s	dual channels to DDR333 at 8B = 5.3GB/s for each processor	400MHz SDRAM at 16B = 6.4GB/s
Configured Memory DIMMs	512MB DDR266 (PC2100) ECC CL2 Reg. (8 DIMMS)	512MB DDR 333, CL2.5, ECC, Reg. (4 DIMMS)	256MB DDR400 (PC3200) CL3, not ECC nor Reg. (2 DIMMS)
Processor-processor interconnect	Shared front-side bus (FSB)	HyperTransport links	PowerPC G5 system controller



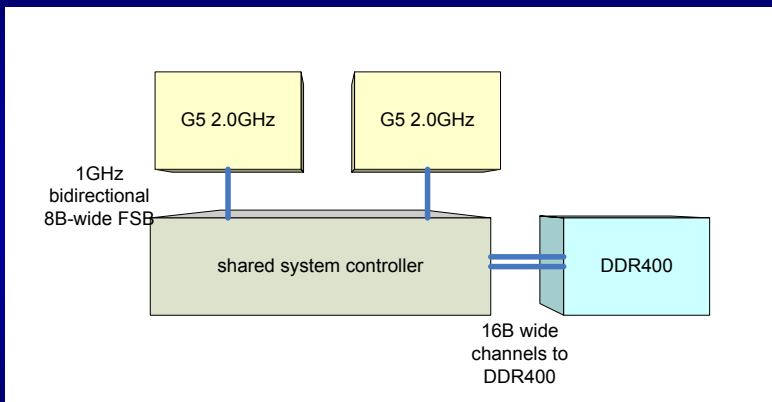
HP rx2600



IBM e325



Apple Power Mac G5



Micro-benchmarks

1. FLOPS

A serial C program that estimates system MFLOPS rating for FADD, FSUB, FMUL and FDIV operations.

This provides an estimate of peak MFLOPS performance for a single floating-point unit (FPU) by making maximal use of register variables with minimal interaction with main memory.

	Itanium2	Opteron	G5	Xeon
Compiler	icc 8.0	pgcc 5.1.3	gcc 3.3	icc 8.0
Compiler options	-O3 -tp p2	-O3 -fast	-O4	-O3 -tpp7 -xW
MFLOPS(1)	2.59	2.074	1.38	0.71
MFLOPS(2)	1.48	0.833	0.93	0.86
MFLOPS(3)	2.51	1.20	1.67	1.61
MFLOPS(4)	4.13	1.41	3.54	2.56



Micro-benchmarks

2. High Performance Linpack involves the solution of a dense linear algebra system. System performance is dependent on factors such as the quality of the BLAS library implementation of the DGEMM function and to a lesser extent the MPI implementation used and the interconnect network speed.

	Itanium2	Opteron	G5	Xeon
Compiler	icc/fort 8.0	pgcc/pgf90 5.1.3 for x86-64	gcc 3.3	icc/icc 7.1
Compiler options	-O2 -tpp2 - nofor_main	-O3 -tp k8-64	-O3 -mtune=970 -mcpu=970 -W -Wall -Wl,framework,vecLib	-O3 -tpp7 -xW
MPI library	LAM 7.0.4	LAM 7.0.4	LAM 7.0.4	LAM 7.0.4
BLAS	Goto BLAS	Goto BLAS	Apple VecLib	Goto BLAS
Single node, two processors: PxQ=1x2 GFLOPS/s	10.52	5.815	7.949	8.96
Two nodes, two processors: PxQ=1x2 GFLOPS/s	10.28	5.869	8.225	9.21



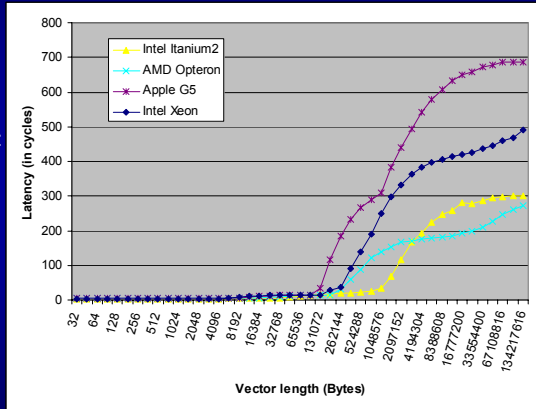
Micro-benchmarks

3. Memory Latency Measurement

Latency in accessing data from all levels of the memory hierarchy is calculated using the following loop:

```

ip1=ia(1)
do i=2,n
  ip2=ia(ip1)
  ip1=ip2
end do
    
```



Micro-benchmarks

4. STREAM

The STREAM benchmark measures the memory bandwidth for the following four kernels:

```

COPY:  a(i) = b(i)
SCALE: a(i) = q*b(i)
SUM:   a(i) = b(i) + c(i)
TRIAD: a(i) = b(i) + q*c(i)
    
```

	Itanium2	Opteron	G5	Xeon
Compiler	icc 8.0	pgcc 5.1.3	xlc	icc 7.1
Compiler options	-O3 -tpp2 -ftz -fno-alias	-O3 -Mvect=prefetch -Mnontemporal -Munroll -fastsse	-O5 -qarch=ppc970 -qtune=ppc970 -qaltivec -qunroll=auto -qcache=auto	-O3 -tpp7 -axW -fno-alias
Copy BW (MB/s)	3380.8	3657.2	3012.0	2977.0
Scale BW (MB/s)	3288.4	3604.8	2670.7	2952.2
Sum BW (MB/s)	3796.7	3602.5	2577.0	2921.4
Triad BW (MB/s)	3869.5	3621.8	2573.2	2881.2



Micro-benchmarks

4. STREAM

The STREAM benchmark measures the memory bandwidth for the following four kernels:

COPY: $a(i) = b(i)$

SCALE: $a(i) = q * b(i)$

SUM: $a(i) = b(i) + c(i)$

TRIAD: $a(i) = b(i) + q * c(i)$

	Itanium2	Opteron	G5	Xeon
Compiler	icc 8.0	pgcc 5.1.3	xlc_r	icc 7.1
Compiler options	-O3 -tpp2 -ftz -fno-alias -openmp	-O3 -mp -fastsse -Mnontemporal	-O5 -qarch=ppc970 -qarch=ppc970 -qcache=auto -qhot=vector	-O3 -tpp7 -ftz -fno-alias -openmp
Copy BW (MB/s)	3872.0	6394.2	3000.5	1935.2
Scale BW (MB/s)	3663.8	6735.5	2882.6	1775.9
Sum BW (MB/s)	3271.9	6622.0	2992.5	1731.0
Triad BW (MB/s)	3465.6	6650.7	2963.5	1834.2



Micro-benchmarks

5. On-node MPI

The PRESTA benchmarks measure, among other things, the latency along with the unidirectional and bidirectional bandwidths of point-to-point MPI communication.

	Itanium2	Opteron	G5	Xeon
MPI library	LAM-MPI 7.0.4	LAM-MPI 7.0.4	LAM-MPI 7.0.4	LAM-MPI 7.0.4
Max. unidirectional BW (GB/s)	3.06	0.781	1.23	0.810
Max. bidirectional BW (GB/s)	2.93	1.25	1.32	0.838
Latency (microseconds)	2.9	0.61	0.71	0.60



Application Benchmarks

- Unless otherwise stated, all applications run on the Itanium2 and Opteron are 64-bit.
- All apps run on the G5 are 32-bit.
- Various compiler options were tried, we only report those that gave the best performance
- Applications with different performance characteristics were chosen for this study.
- All tests were run on non-busy systems.

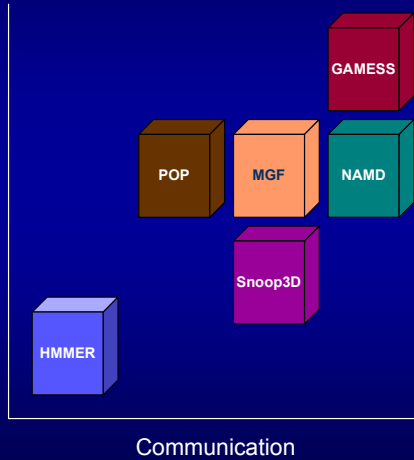


Application Benchmarks

Code	Application area	Memory BW	CPU	Interconnect BW
Snoop3D	CFD	low	high	moderate
MGF	Finite element	high	high	moderate
POP	Ocean modeling	high	high	low
GAMESS	Chemistry/Physics	high	high	high
NAMD	Chemistry/Biochemistry/ Biology	moderate	high	high
HMMER	Bioinformatics	low	high	none



Applications



Application Benchmarks – Snoop3D

- CFD Application
- Snoop3D is a 3-D unsteady, incompressible flow solver using unstructured meshes.
- High CPU intensive, moderate interconnect B/W and low memory B/W intensive code requirements.
- Timings are average of four simulation runs with each run simulating 1s of fluid flow corresponding to 50 iterations of the solver.

	Itanium2	Opteron	G5	Xeon
Compiler	icc 8.0	gcc 3.0	xlc	icc 7.1
Compiler options	-O3 -ip -fno-alias -ftz	-O3 -ffast-math -fargument- noalias	-O4 -qnoipa qmaxmem=-1 -qstrict	-O3 -ip -fno-alias -axW -tpp7
MPI library	MPICH	MPICH	MPICH	MPICH
Execution Time (s)	212.91	154.24	157.12	117.85

Application Benchmarks -- MGF

- Finite Element Application
- MGF is a parallel 3-D multi-physics problem for high performance simulation of thermo-capillary flows in microgravity.
- High CPU and memory B/W intensive, moderate interconnect B/W intensive code requirements.
- Matrix-vector product is the main computational kernel.
- Timings are based on serial and parallel execution on two processors.

	Itanium2	Opteron	G5	Xeon
Compiler	icc 8.0	gcc/g++ 3.3	gcc/g++ 3.3	icc 7.1
Compiler options	-O2	-O2	-O2 -faltivec -framework	-O2
MPI library	MPICH	MPICH	MPICH	MPICH
BLAS library	Goto BLAS	Goto BLAS	Veclib	Goto BLAS
Serial (s)	74.01	105.65	151.57	141.23
Two processors, on-node (s)	45.50	56.90	94.69	74.64
Two processors, across two nodes (s)	38.61	57.08	76.75	78.73



Application Benchmarks -- POP

- Climate-Weather Application
- Parallel Ocean Program is an ocean circulation model that solves 3-D primitive equations for fluid motions on the sphere under hydrostatic and Boussinesq approximations.
- High CPU and memory B/W intensive, moderate interconnect B/W intensive code requirements.
- Timings are based on serial and parallel execution on two processors.

	Itanium2	Opteron	G5	Xeon
Compiler/MPI library	ifort 8.0	pgf90 5.1.3	xlf90	ifc 7.1
Compiler options	-O3 -ip -fno-alias	-O3	-O4 -qnoipa -qmaxmem=-1 -qstrict	-O3 -ip -fno-alias -xW -tpp7
MPI library	MPICH	MPICH	MPICH	MPICH
Serial (s)	107.65	63.82	41.67	40.12
Two processors, on-node (s)	55.24	35.40	30.41	26.04
Two processors, across two nodes (s)	56.04	33.27	25.64	19.92



Application Benchmarks -- GAMESS

- Quantum chemistry application
- The General Atomic and Molecular Electronic Structure System (GAMESS) is a general *ab-initio* quantum chemistry package.
- Supports parallel execution, sockets implementation used
- Numerically-, memory- and BW-intensive
- Timings are based on serial and parallel execution on two processors.

	Itanium2	Opteron (32-bit mode)	G5	Xeon
Compiler	ifort/icc 8.0	pgf90/pgcc 5.1.3	xlf90/xlc	ifc/icc 7.1
Compiler options	-O2	-O3 -fastsse -Mnontemporal	-O3 -qhot -qtune=ppc970 -qarch=ppc970	-O3 -axW
Serial (s)	289	178.7	136.0	135.8
Two processors, on-node (s)	145	102.5	75.5	92.7
Two processors, across two nodes (s)	122	95.2	---	79.1



Application Benchmarks -- NAMD

- Molecular modeling application
- NAMD is a parallel molecular dynamics code targeted towards the high-performance simulation of large, biomolecular systems
- Numerically-intensive, moderate memory use, communication demand is high
- Timings are based on serial and parallel execution on two processors.

	Itanium2	Opteron	G5	Xeon
Compiler	icc/icpc 8.0	gcc/g++ 3.3	gcc/g++ 3.3	icc/icpc 7.1
Serial (s)	1104	1978	2583	1901
Two processors, on-node (s)	578	1029	1372	1010
Two processors, across two nodes (s)	576.2	1101	1332	974



Application Benchmarks -- HMMER

- Bioinformatics application
- HMMER is an implementation of profile hidden Markov models (HMMs) for biological sequence analysis
- pthreads implementation used here
- Numerically intensive, low communication demand

	Itanium2	Opteron	G5	Xeon
Compiler	icc 8.0	pgcc 5.1.3	gcc 3.3	icc 7.1
Compiler options	-O3 -mcpu=itanium2 -fno-alias	-O3 -fastsse -Mnontemporal	-O3 -mcpu=G5 -mtune=G5	-O3 -tpp7 -axW
Serial (s)	136.6	191	178	198
Dual-threaded (s)	71.2	85.7	115	111



Summary

- 64-bit computing require both hardware (e.g., wider data and address registers) and software support. Itanium2 and Opteron are fully-64 bit.
- Micro-benchmark results consistent with what is expected from the system architecture:
 - Itanium and G5 performed very well on FLOPS, Linpack
 - Opteron shows great scalability on STREAM
- Application benchmark results are harder to rationalize
 - All apps benchmarked are FP-intensive, but Itanium2 perform best on only three (MGF, NAMD, HMMER); G5 performs best on GAMESS and POP
 - Itanium2 does comparatively poorly on GAMESS, POP and Snoop3D
 - Itanium2 outperforms Opteron and G5 on HMMER, even though latter two have SIMD units
- Application results that do make sense
 - Itanium2 great for BLAS and Linpack
 - Opteron exhibits best scalability overall



Summary

- If memory addressability is not an issue, Xeon is on par with (and sometimes significantly outperforms) other three systems
- Based on pricing:

HP rx2600: \$8570

IBM e325 : \$2796

Apple Power Mac G5: \$2999

Price/performance leader among 64-bit systems for the applications benchmarked here is Opteron system

- 64-bit hardware, OS, compilers, libraries
- easy migration path from 32-bit to 64-bit mode
- math libraries available



Acknowledgments

Thanks to

HP, Apple, IBM for loaner systems

LCI committee

