Intel Woodcrest: An Evaluation for Scientific Computing[1]

Philip C. Roth and Jeffrey S. Vetter

Future Technologies Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory
One Bethel Valley Road
Oak Ridge, TN 37831
{rothpc,vetter}@ornl.gov

Intel recently began shipping its Xeon 5100 series processors, formerly known by their "Woodcrest" code name. To evaluate the suitability of the Woodcrest processor for high-end scientific computing, we obtained access to a Woodcrest-based system at Intel and measured its performance first using computation and memory micro-benchmarks, followed by full applications from the areas of climate modeling and molecular dynamics. For computational benchmarks, the Woodcrest showed excellent performance compared to a test system that uses Opteron processors from Advanced Micro Devices (AMD), though its performance advantage for full applications was less definitive. Nevertheless, our evaluation suggests the Woodcrest to be a compelling foundation for future leadership class systems for scientific computing.

Introduction

In late June 2006, Intel began shipping its Xeon 5100 series processors. These processors, code-named "Woodcrest," are dual-core processors using the company's Core architecture. To evaluate the suitability of the Woodcrest processor for scientific computing, we obtained access to a system at Intel with two Woodcrest processors through an early access program. We benchmarked the system following the methodology used in past early evaluations of high performance computing (HPC) systems at Oak Ridge National Laboratory (ORNL) [1, 2], adapted slightly to reflect differences in the scales of the systems under consideration. In our evaluation, we used not only computation and memory micro-benchmarks, but also three full scientific applications from the areas of climate modeling and molecular dynamics. Because the AMD Opteron processor features prominently in current systems from HPC vendors like Cray Inc., we compared the Woodcrest system's performance with that of an Opteron-based system with a similar socket organization. Our evaluation suggests the Wood-

[1] This research is sponsored by the Office of Advanced Scientific Computing Research; U.S. Department of Energy. The work was performed at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725.

crest to be a compelling foundation for building leadership class systems for scientific computing.

The Woodcrest Processor

The Intel Woodcrest processor contains two processor cores and a cache hierarchy. Each processor core employs Intel's next generation Core microarchitecture [3]. Cores using this microarchitecture can sustain execution of four instructions per clock cycle, compared to three instructions per cycle with previous Intel microarchitectures. Also, cores with this microarchitecture can execute 128-bit SIMD instructions at a rate of one per cycle; the previous generation microarchitecture could produce one 128-bit SIMD result every two cycles. Furthermore, the microarchitecture includes both a floating-point multiply unit and a floating-point add unit, each of which can operate on two packed double-precision values each cycle using 128-bit SIMD instructions. Thus, each Woodcrest core is capable of producing four double-precision floating-point results per clock cycle. In contrast, current generation AMD Opteron processors can produce at most two double-precision floating point results per clock cycle [4].

The Woodcrest processor memory hierarchy is implemented with a combination of on-chip memory caches, an off-chip memory controller, and Fully Buffered Dual Inline Memory Modules (FB-DIMMs). Each processor core has its own L1 caches (16KB for instructions and 16KB for data), but a 4MB unified L2 cache is shared by both cores. Memory accesses that are not satisfied in the processor caches are transferred via front-side bus (FSB) to a memory controller. The Woodcrest processor line uses FB-DIMM memory that provides better bandwidth than DDR and DDR2 memory technology. The Intel memory hierarchy design, especially for Woodcrest, differs substantially from the AMD approach. In current AMD dual-core processors, cache is not shared between cores. Furthermore, a memory controller is located on the processor chip with some system memory being local to each processor socket, a design that AMD argues will scale better than one with an off-chip memory controller as the number of processors increases. However, the AMD approach results in a Non-Uniform Memory Access (NUMA) architecture, where the time required to access a memory location differs depending on whether the location is local to the processor performing the access. On such architectures, good process placement can be critical for achieving good performance. In contrast, the Intel design results in a Uniform Memory Access (UMA) architecture that may be more forgiving to bad process placement than a NUMA design. However, process placement cannot be ignored on systems with Woodcrest processors because the Woodcrest cores share an L2 cache.

The Woodcrest processor line includes parts clocked from 1.6 GHz (Xeon 5110) to 3.0 GHz (Xeon 5160). The Thermal Design Point (TDP) for the entire line except the 5160 model is 65W; the TDP for that part is 80W.

Test System Configurations

We used two systems for most of this evaluation: one with Intel Woodcrest processors and one with AMD Opteron processors. At the time our evaluation, both systems had two dual-core processors for a total of four cores per system.

The Woodcrest system contained two Xeon 5160 dual-core processors. The processor clock rate was 3.0 GHz and the front-side bus rate was 1333 MHz. The system contained 8 GB of FB-DIMM memory. We obtained access to the evaluation Woodcrest system through the Intel Remote Access Service. This service provides access to the latest Intel technologies and tools via the Internet for evaluation, validation, and optimization activities. For one benchmark, we also used results obtained from an Intel Woodcrest system containing two Xeon 5150 dual-core processors, with clock rate 2.66 GHz and 8 GB memory.

The Opteron system contained two Opteron 2220 SE dual-core processors, running at 2.8 GHz. This test system contained 8 GB DDR-400 memory. We obtained access to the evaluation Opteron system through the AMD Developer Center remote access service. Like Intel's service, the AMD Developer Center provides access to the latest AMD technologies and tools for remote evaluation, validation, and optimization efforts.

Both systems used a Linux 2.6 kernel. On the Woodcrest system, we used the Intel Fortran and C/C++ compilers (version 9.1), the Intel Math Kernel Library (version 8.1), and Intel Message Passing Interface (MPI) library (version 2.0). Unlike Intel, AMD does not develop its own compiler, relying on products from companies like Portland Group, Pathscale, and even Intel for commercial compiler support. On our Opteron test system, we used the Portland Group Fortran and C compilers (version 6.2-5), the AMD Core Math Library (version 3.6), and the OpenMPI MPI implementation (version 1.1.4). We chose to use the Portland Group compiler on our Opteron test system because of its position as the preferred compiler on Opteron-based systems from Cray Inc.

Clearly, there are substantial differences in both hardware and software between the systems we considered. For example, AMD has traditionally downplayed differences in processor clock speed, a position recently adopted by Intel, so we struggle to isolate the performance differences in our benchmarking observations from both the perspective of raw performance and the relative to differences in test system hardware. Likewise, we felt it best to use the software suite (compilers and math libraries) that were likely to be used for scientific computing on each test system rather than arbitrarily choosing commercial compiler for all tests or using a free alternative like the GNU Compiler Collection. Whenever possible, we tried to use comparable optimization flags between compilers, but we cannot guarantee that we found the best platform- and compiler-specific optimization flags or that we spent exactly the same amount of time optimizing for each platform.

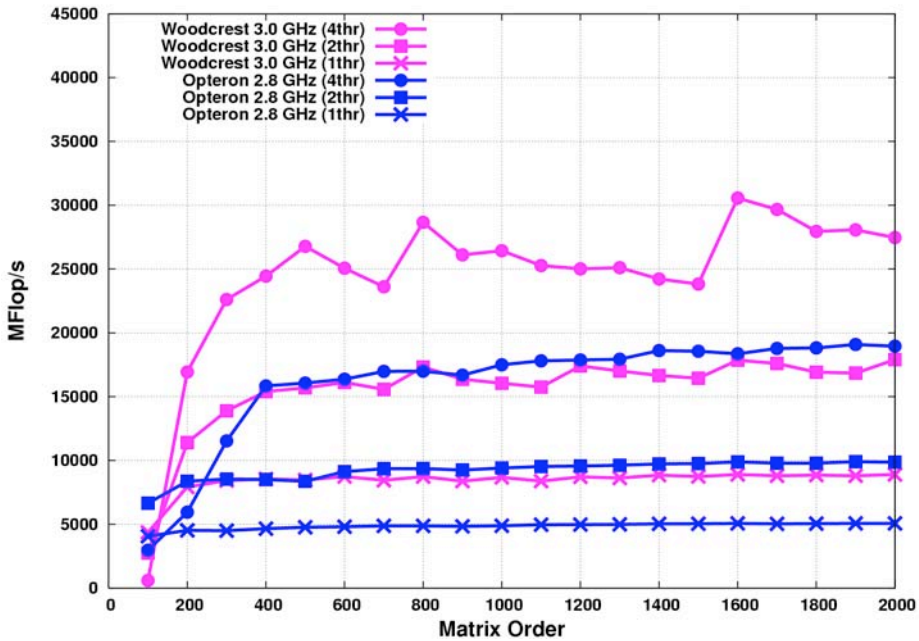

Fig. 1. Double precision matrix-matrix multiply (DGEMM) performance

Computation Performance

To evaluate Woodcrest's floating point capabilities, we measured the performance of the Woodcrest system when executing three important floating point operations: double-precision matrix-matrix multiply (DGEMM), computing the LU factorization of a matrix (DGETRF), and forming the Fast Fourier Transform (FFT) of a vector. We measured the performance of these operations as implemented by the Intel Math Kernel Library (MKL) and the AMD Core Math Library (ACML) on the Woodcrest and Opteron, respectively. We used OpenMP-enabled versions of both libraries.

Matrix-Matrix Multiply

DGEMM, the double-precision matrix-matrix multiply routine from the Linear Algebra Package (LAPACK), is often used as a measure of a system's maximum computational capability for scientific computing. To evaluate DGEMM performance of our test systems, we measured the elapsed time (i.e., wall clock time) required to complete a DGEMM operation on square matrices with order ranging from 100 to 2000 using one, two, and four OpenMP threads. We converted these elapsed times into throughput values and charted them in Fig. 1. Each data point represents the average of three runs.

In absolute terms, the Woodcrest DGEMM performance is exceptional. On one core, the Woodcrest system achieved just over 8.9 GFlop/s; with four OpenMP threads on all four cores, we observed 30.8 GFlop/s. In comparison, the Opteron system achieved a maximum of just over 19 GFlop/s with four OpenMP threads. That each Woodcrest core is theoretically capable of producing twice as many floating point results per clock tick as the Opteron is a significant factor in this performance difference. However, based on the clock speeds of the two systems and the theoretical computational rate, we observed the Opteron to run at higher floating-point efficiency than the Woodcrest (85% versus 64% with four threads). Although the Woodcrest's microarchitecture is capable of producing four floating point results per cycle per core, achieving this throughput requires careful scheduling of floating point operations to keep the available functional units busy. The observed DGEMM efficiency results show that the MKL DGEMM implementation does not (perhaps cannot, due to the constraints of implementing a DGEMM operation) keep the functional units available in the Woodcrest core as busy as the Opteron's fewer functional units with the ACML implementation. At least with the current processor hardware and math library implementations, those deploying a system must make a trade-off depending on what is most important to them: higher absolute performance or higher computational efficiency.

LU Factorization

Factorization of a matrix into lower and upper triangular matrices is another common operation, useful when solving linear systems. We measured the system throughput when executing DGETRF, the double precision LU factorization routine for general matrices from the LAPACK implementations of the MKL and ACML libraries. Our measurements are charted in Fig. 2. The performance difference between the Woodcrest and Opteron is not as pronounced as it was for DGEMM, but the Woodcrest does show a performance advantage at each thread count.

Fast Fourier Transform

Fast Fourier Transform is the third computational micro-benchmark we considered in our Woodcrest evaluation. As with DGEMM and LU, we timed the implementation provided in the vendor's processor-specific math library, and converted them to floating point throughput values. Our one- and two-dimensional complex FFT measurements are shown in Fig. 3 and Fig. 4, respectively. Note that due to problems with the results collection from the 3.0 GHz Intel remote access system, our Woodcrest FFT results were collected from a system with 2.66 GHz Woodcrest processors in our group's Experimental Computing Laboratory at ORNL.

The performance profiles for the FFT operation are dramatically different between the one-dimensional and two-dimensional FFTs. For the one-dimensional FFTs, the Opteron system showed good scalability to slightly longer vector lengths than the Woodcrest system, but the Woodcrest system dramatically outperformed the Opteron system at all vector lengths despite its lower clock rate. OpenMP threading provided

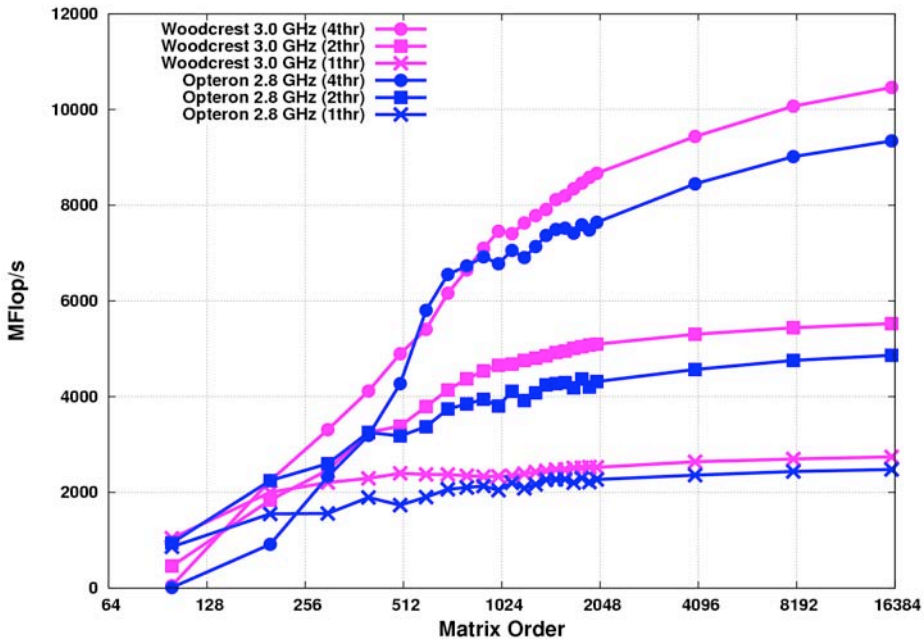

Fig. 2. Double precision LU factorization (DGETRF) performance

no benefit for either FFT implementation. In contrast, threading showed a substantial benefit for the two-dimensional complex FFTs on both systems, and when there are enough threads to saturate the cores the Opteron outperforms the Woodcrest system at all but the smaller problem sizes.

Memory Hierarchy Performance

In contrast to the Opteron dual-core processors that do not share cache between cores, Woodcrest processors have a 4MB shared L2 cache. Also, Woodcrest systems use FB-DIMM memory, whereas the Opteron uses DDR memory. These differences in memory hierarchy contribute to different memory access profiles for the two systems.

Fig. 5, Fig. 6, and Fig. 7 show the bandwidth observed when accessing vectors of increasing length, as measured by the CacheBench memory benchmark [5]. CacheBench supports measurement of memory hierarchy bandwidth under three memory access patterns: sequential reads to the elements of a single vector, sequential writes to the elements of a single vector, and sequential increments to the elements of a single vector (a pattern called "read-modify-write" in the CacheBench literature). Although CacheBench is not multi-threaded, measurements were taken when one, two, and four copies of the CacheBench program were running to try to capture the effect of memory contention. For Woodcrest, an error in the script that ran the experiment overwrote the two-thread results, hence Woodcrest results are shown only for one and four threads.

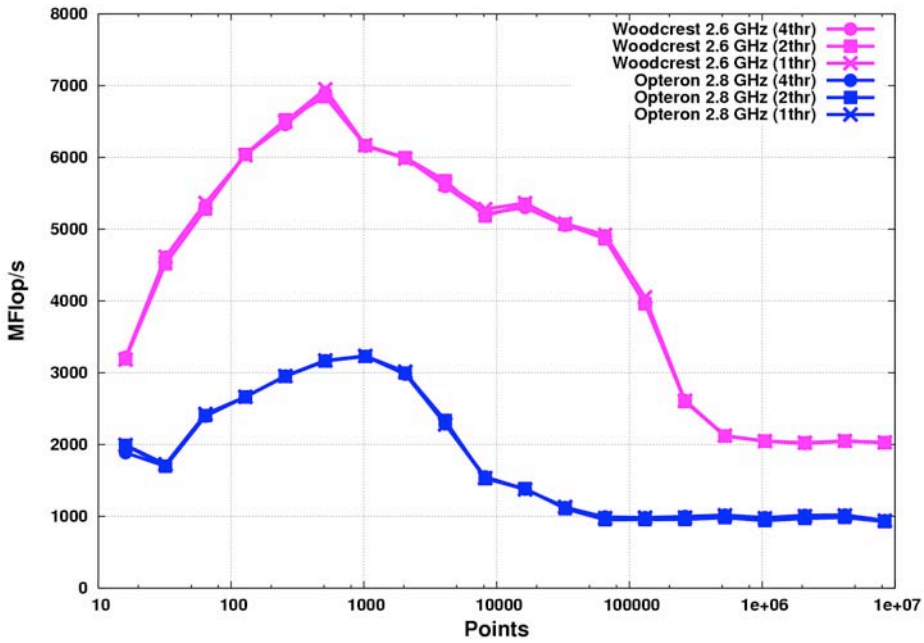

Fig. 3. One dimensional complex Fast Fourier Transform performance

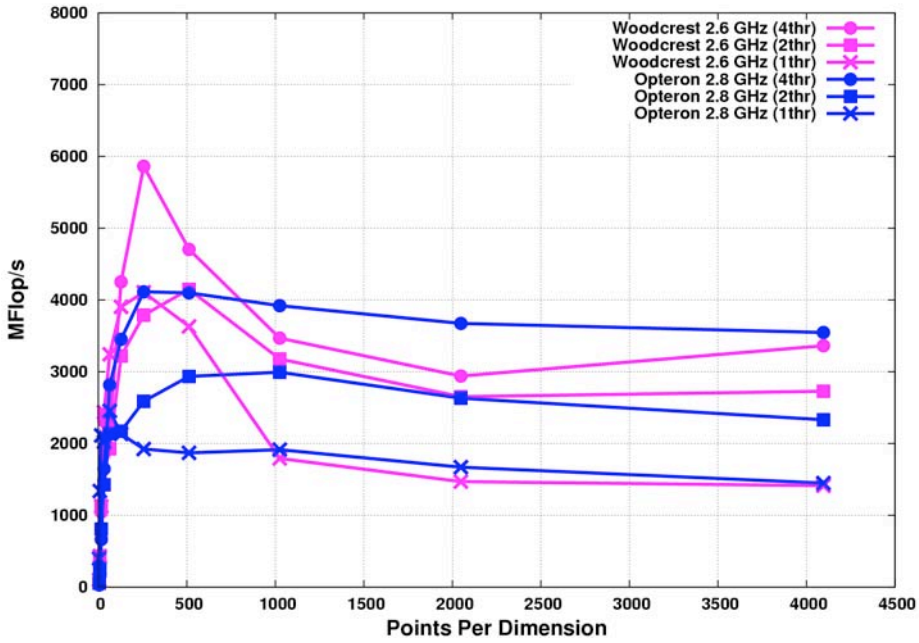

Fig. 4. Two dimensional complex Fast Fourier Transform performance

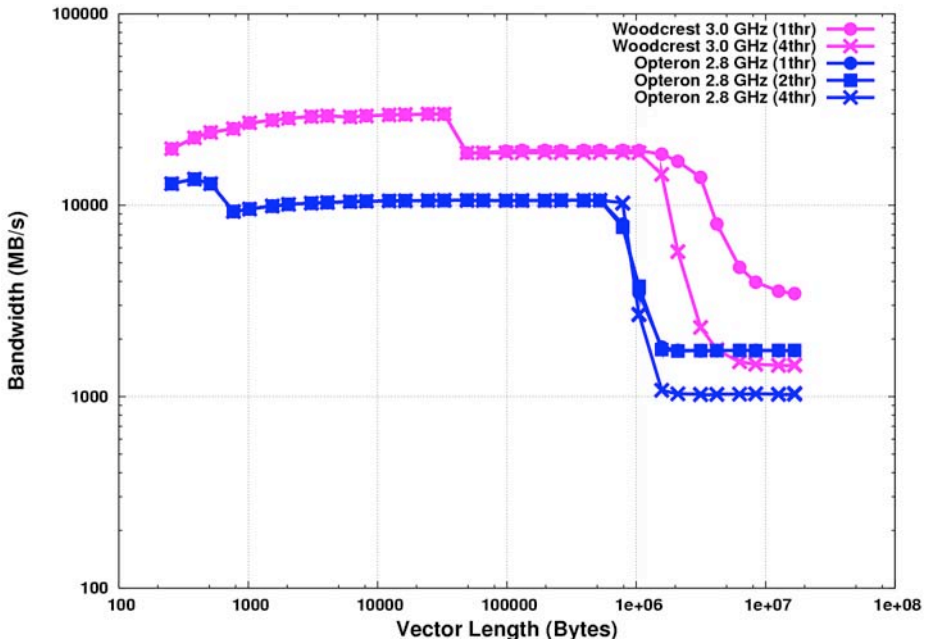

Fig. 5. CacheBench read performance

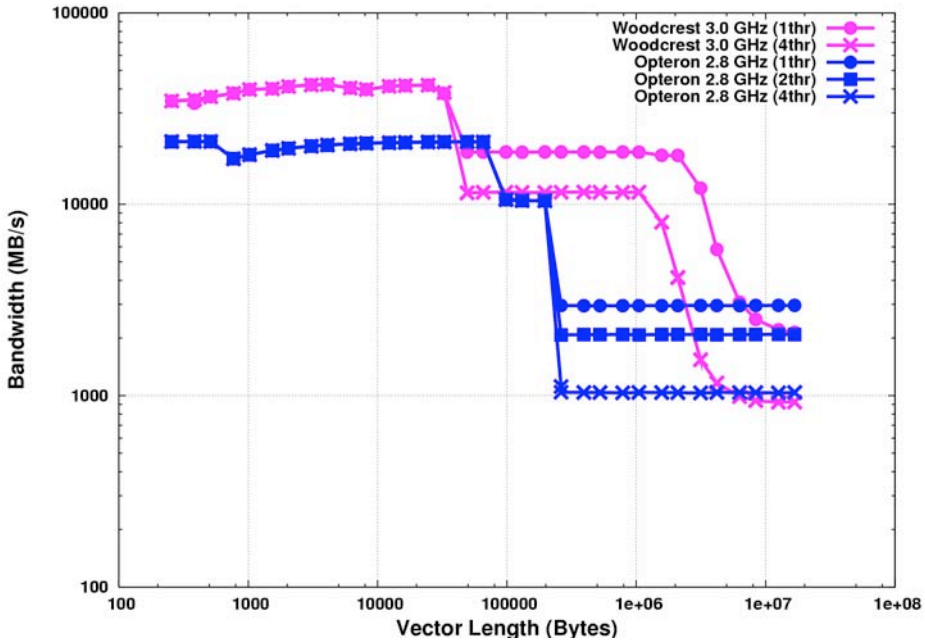

Fig. 6. CacheBench write performance

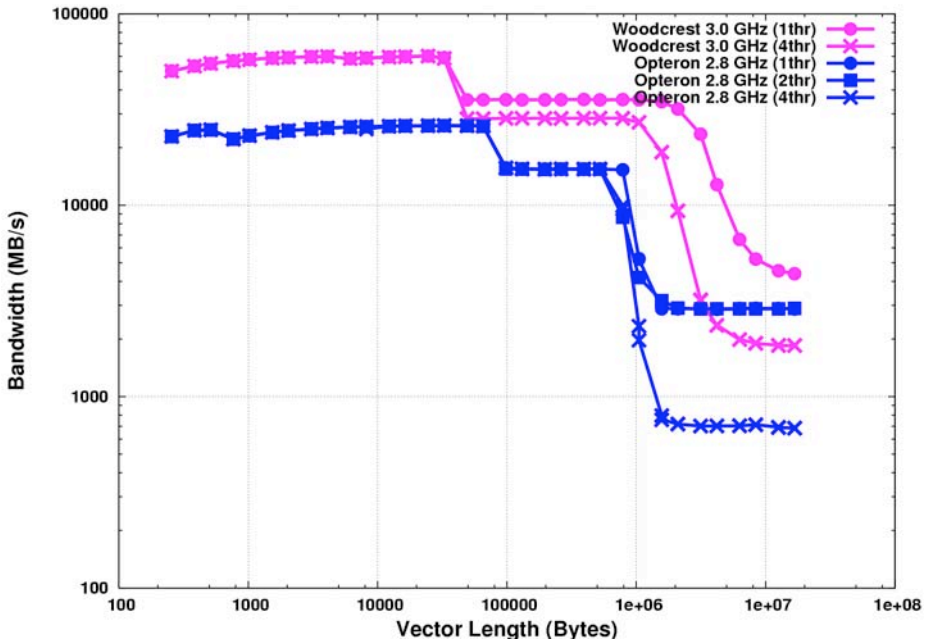

Fig. 7. CacheBench read-modify-write performance

Table 1. STREAM memory bandwidth benchmark performance
(All measurements in MB/s.)

OpenMP Threads	Woodcrest 5160 (3.0 GHz, 1333 MHz FSB)	Opteron 2220 SE (2.8 GHz)
1	4217	5782
2	7355	11183
4	7494	12222

The observed memory bandwidth for the Woodcrest processors is superior compared to the Opteron test system in most cases, with the notable exception for large vectors accessed from memory when all processor cores were used (i.e., with four threads). We believe this result shows the benefit of the use of FB-DIMM memory in the Woodcrest system when there is little contention for the system's front side bus, and the advantage of the AMD approach when contention for the links to memory is expected in the workload. For both Opteron and Woodcrest, the effects of contention are apparent in our CacheBench results, but note that the Woodcrest shows the effects of contention in accesses to the L2 cache with writes, whereas the Opteron results show no sign of contention until vector lengths cause accesses to be satisfied from main memory.

The STREAM benchmark [6] provides another perspective on the memory performance of the Woodcrest processor. The STREAM micro-benchmark measures memory bandwidth when performing simple operations (copy, add, multiply, and a combination of the three called 'Triad') on long vectors. The vectors must be long

enough that the benchmark measures bandwidth to main memory instead of to any level of data cache. Unlike the CacheBench benchmark, each STREAM operation involves elements from at least two vectors and the vectors' relative location in memory can affect observed memory bandwidth. In particular, poorly placed vectors can result in cache conflicts when accessing corresponding elements of the vector, dramatically decreasing the observed memory hierarchy bandwidth.

Our observed STREAM Triad results for one, two, and four OpenMP threads on the Woodcrest are shown in Table 1. We suspect that slightly higher bandwidths may be possible with further optimization; we performed a minimal amount of tuning of vector lengths and used a collection of compiler flags that had proven to give good performance in our other experiments. For comparison, we also include STREAM Triad results from our 2.8 GHz Opteron system. As expected, the Opteron system memory bandwidth is quite high compared to that of our test Woodcrest system, due to the AMD memory design including its on-chip memory controllers and memory local to a processor socket.

Our memory micro-benchmarks measure performance of memory accesses satisfied at all levels of the memory hierarchy. We emphasize that neither CacheBench nor STREAM gives a complete picture regarding memory hierarchy performance, especially since both use synthetic memory access patterns that are rarely encountered in full applications. Nevertheless, they provide valuable information regarding the maximum achievable performance to each level of the memory hierarchy.

Application Performance

Our micro-benchmark results suggest the Woodcrest is well suited for systems designed for computational science. To further evaluate its suitability, we ran three full scientific applications on the Woodcrest system: the Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) [7], the Climate Atmosphere Model (CAM) [8], and the Parallel Ocean Program (POP) [9]. LAMMPS and POP use MPI for communication between parallel tasks, whereas CAM was built to use only OpenMP. Although the test systems were not distributed memory systems, we considered MPI-only applications because of the likelihood that MPI performance will remain important in the near future, due to portability concerns and development team inertia.

LAMMPS

LAMMPS is a C++ program that uses MPI for communication. We used the "12Apr06" version of LAMMPS for our timings. The LAMMPS distribution includes several benchmark problems; we chose three for our timings. We ran each in weak scaling mode. We used the Intel MPI library on the Woodcrest system, and used the OpenMPI library version 1.1.4 built using the PGI compilers on our test Opteron system. The timings from our LAMMPS experiments with one, two, and four MPI tasks are shown in Fig. 8, Fig. 9, and Fig. 10 (Eam benchmark, Lj benchmark, and Rhodo benchmark, respectively). Each chart plots the elapsed time required to compute 100 LAMMPS timesteps, so lower values are better than higher values. The Eam and Lj problems were run with 1,024,000 atoms, whereas the Rhodo problem was run with 128,000 atoms.

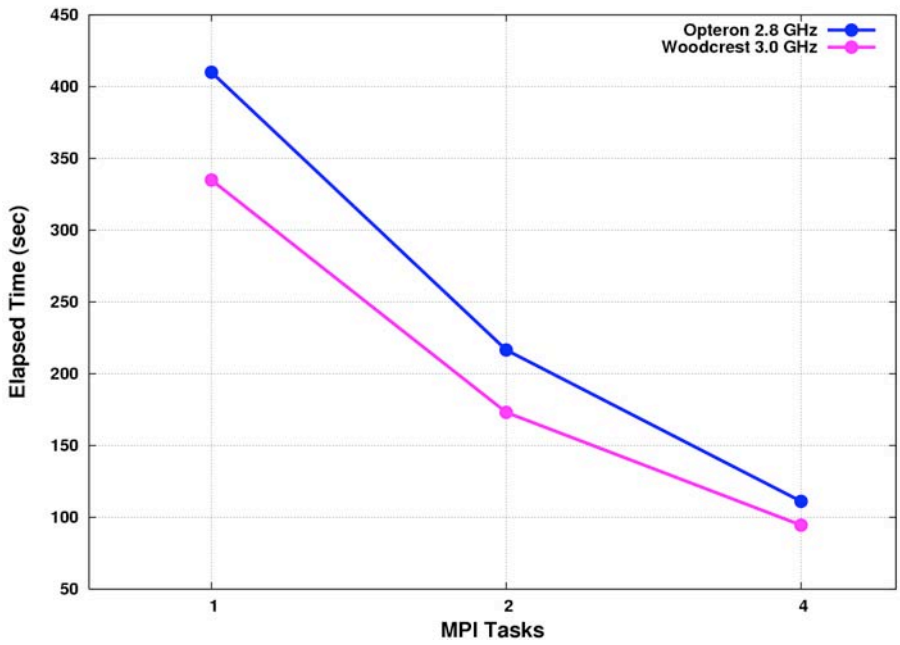

Fig. 8. LAMMPS Eam benchmark problem performance

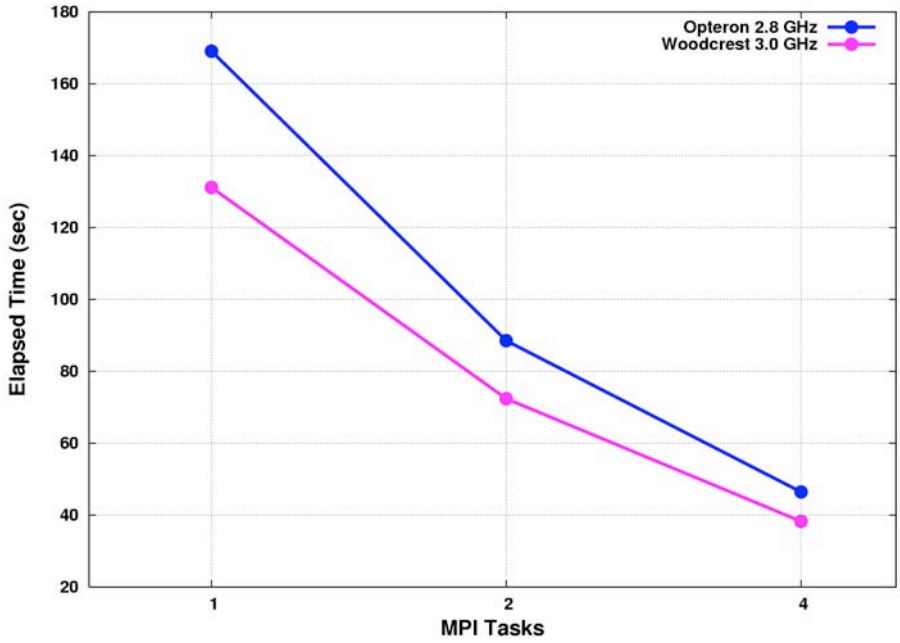

Fig. 9. LAMMPS Lj benchmark problem performance

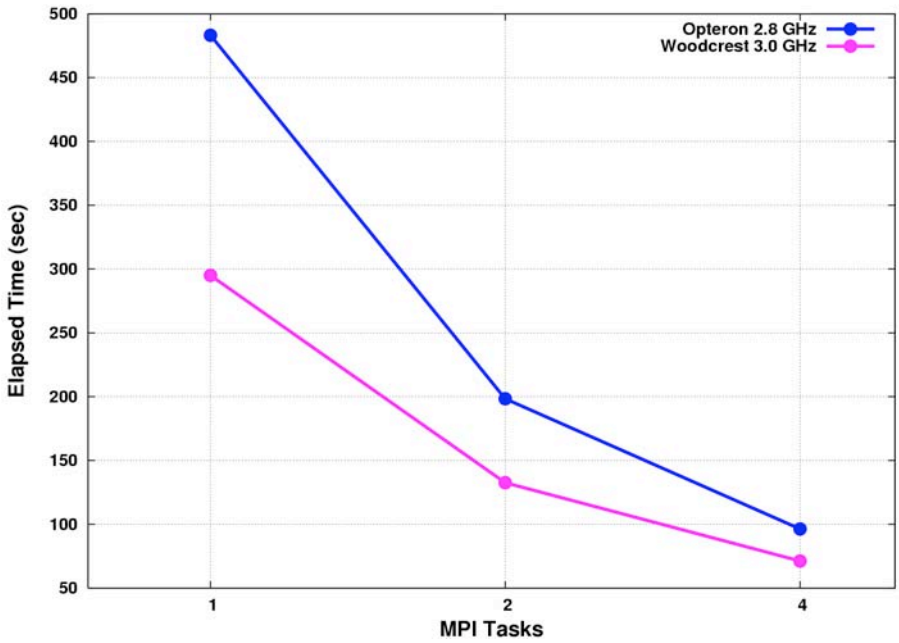

Fig. 10. LAMMPS Rhodo benchmark problem performance

For all LAMMPS problems we considered, the Woodcrest showed better performance than the Opteron, although the performance gap narrowed as the number of MPI tasks was increased. Furthermore, the performance gap was more pronounced for the Rhodo problem than the other problems.

CAM

CAM consists mainly of Fortran code, and can be built to use MPI, OpenMP, or both for expressing parallelism. To ensure that we considered one example of a "pure" OpenMP application, we built CAM to use OpenMP only. We used CAM version 3.0p1 for our timings. After determining that CAM is extremely sensitive to the OpenMP per-thread stack size, we were able to run it on both the Woodcrest and Opteron test systems and measured the time required to complete two simulation days of the T42 benchmark problem (144 timesteps plus one partial timestep). The results of our CAM timings are shown in Fig. 11. The trend shown in the chart is similar to that shown by the LAMMPS timings: the Woodcrest system provided superior performance for the CAM code over the Opteron system, but the differences were not as pronounced as for the LAMMPS benchmark problems.

POP

POP is a Fortran 90 code that, like CAM, can be built using MPI, OpenMP, or both. Because we used the MPI-only mode for previous evaluations, and because we used

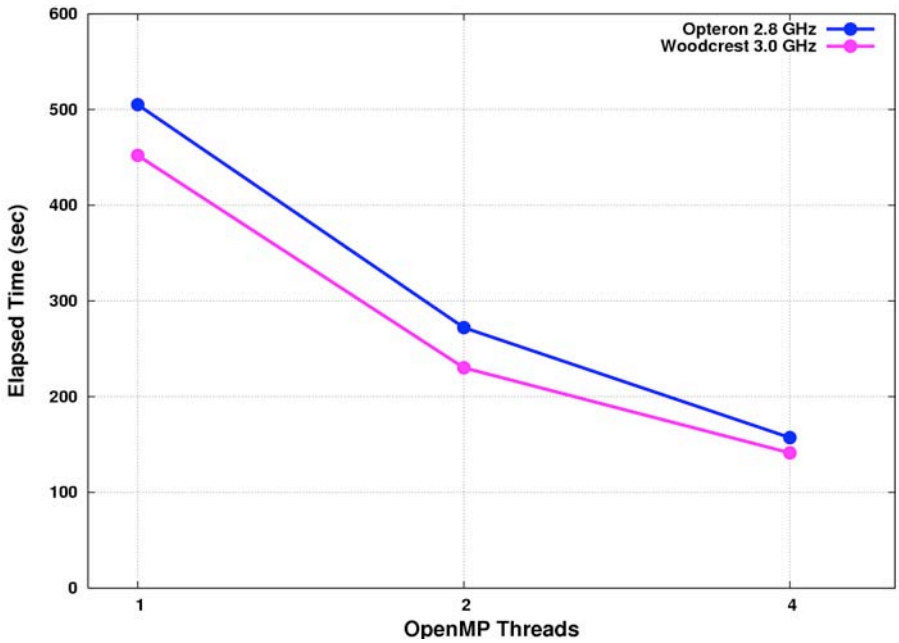

Fig. 11. CAM performance for ten simulation timesteps

CAM in OpenMP-only mode, we built and ran the MPI-only version of POP for this evaluation. We used POP version 1.4.3 for our timings. On the Woodcrest system, we used the Intel MPI library, and on the Opteron system we used OpenMPI 1.1.4 built using the PGI compilers. We collected timings for runs of ten simulation timesteps of the x1 (i.e., one degree) problem. The timings are shown in Fig. 12; each data point in the figure represents the average of three runs. Curves are shown for the total time spent in timesteps (labeled "Steps") and the two main phases of POP execution (labeled "Baroclinic" and "Barotropic").

As with our other test applications, the Woodcrest system showed a clear performance advantage over the Opteron-based system for our benchmark POP problem with two processes (or two threads in the case of CAM), but unlike the other applications the Opteron outperformed the Woodcrest system with four processes. Although further experimentation is needed to be certain, we hypothesize that the POP benchmark problem we used results in a memory access pattern that cannot be satisfied from processor caches and thus places enough pressure on the Woodcrest system's front side bus with four MPI processes to fall into the regime where the Opteron's memory hierarchy design provides better bandwidth (as exposed in both our CacheBench and STREAM Triad benchmark results).

In our initial experiments, we observed large variability in POP run times on the Woodcrest system with two processes. With help from Intel performance engineers, we determined that with the default MPI configuration on the system it was possible for both POP processes to contend for the same processor. When we used process affinity controls provided by the Intel MPI implementation, the variability in run times dropped to negligible levels. These updated results were taken from a slightly differ-

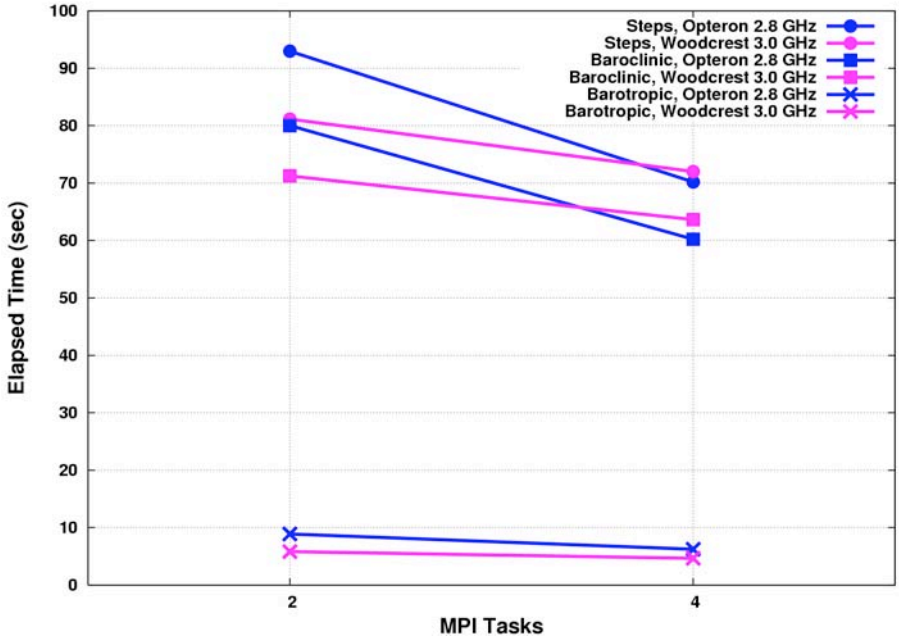

Fig. 12. POP x1 benchmark problem performance

ent Woodcrest system than our initial experiments (with newer processor version and different BIOS settings) but the observed run times on the newer system were approximately the same as the minimum run times from our initial experiments on the original test Woodcrest system.

Summary

We evaluated the dual-core Xeon 5160 processor, code-named Woodcrest, by considering the suitability of a Woodcrest-based system for scientific computing. With a limited time window for accessing the system, we performed a preliminary evaluation using computational and memory micro-benchmarks and several full applications. For computational benchmarks, the Woodcrest showed excellent performance compared to a test system that uses Opteron processors. DGEMM performance of the four-core test system was especially striking using the Intel Math Kernel Library optimized specifically for the Woodcrest processor. For memory benchmarks, the Woodcrest showed good performance compared to the Opteron except in situations with a large degree of contention for the links to memory. Also, the Woodcrest's performance advantage for full applications was less definitive than for the computational benchmarks. We caution that there were many hardware and software differences between the systems, and further comparison is needed to understand the reasons for the performance differences we observed. Also, the Woodcrest evaluation system was described by Intel representatives as an "Alpha level platform" and that the expected performance of shipping Woodcrest-based systems may differ from the performance

of the evaluation system due to processor and chipset revisions in shipping systems. Nevertheless, our evaluation suggests the Woodcrest to be a compelling foundation for future leadership class systems for scientific computing.

Acknowledgements

Thanks to the other members of the ORNL Future Technologies Group for their insightful discussion and critique of this work, with special thanks to Sadaf Alam, Richard Barrett, Jeff Kuehn, and Tammy Darland. Thanks also to David Barkai and Michael Greenfield of Intel and the staff of the Intel Remote Access Service, and Roger Wheatley and Geoff Sutton of AMD and the staff of the AMD Developer Center, for their help in securing access to cutting edge Woodcrest and Opteron systems and their help with technical questions about the systems and their software. Finally, thanks to the conference reviewers and conference shepherd, whose comments and questions helped us produce a stronger paper describing our work.

References

1. Dunigan, T.H., Jr., Vetter, J.S., White, J.B., III, Worley, P.H.: Performance Evaluation of the Cray X1 Distributed Shared Memory Architecture. IEEE Micro **25** (2005) 30-40
2. Vetter, J.S., Alam, S.R., Dunigan, T.H., Jr., Fahey, M.R., Roth, P.C., Worley, P.H.: Early Evaluation of the Cray XT3. IEEE International Parallel & Distributed Processing Symposium (IPDPS), Rhodes Island, Greece (2006)
3. Muchmore, S.: Inside the Intel Core Microarchitecture. Intel Developer Forum, Cairo, Egypt (2006)
4. Advanced Micro Devices, Inc.: Software Optimization Guide for AMD64 Processors. Vol. 2006 (2005)
5. Mucci, P.J., London, K., Thurman, J.: The CacheBench Report. University of Tennessee, Knoxville, TN (1998) 19
6. McCalpin, J.D.: Sustainable Memory Bandwidth in Current High Performance Computers. (1995)
7. Plimpton, S.J.: Fast Parallel Algorithms for Short-Range Molecular Dynamics. Journal of Computational Physics, Vol. 117 (1995)
8. Collins, W.D., Rasch, P.J., Boville, B.A., Hack, J.J., McCaa, J.R., Williamson, D.L., Briegleb, B.P., Bitz, C.M., Lin, S.-J., Zhang, M.: The Formulation and Atmospheric Simulation of the Community Atmosphere Model: CAM3. Journal of Climate (2006)
9. Jones, P.W., Worley, P.H., Yoshida, Y., White, J.B., III, Levesque, J.: Practical performance portability in the Parallel Ocean Program (POP). Concurrency and Computation: Experience and Practice (2004)