Evaluation of the SUN UltraSparc T2+ Processor for Computational Science

Martin Sandrieser, Sabri Pllana, and Siegfried Benkner

University of Vienna, Department of Scientific Computing,
Nordbergstrasse 15, 1090 Vienna, Austria
{ms,pllana,sigi}@par.univie.ac.at

Abstract. The Sun UltraSparc T2+ processor was designed for throughput computing and thread level parallelism. In this paper we evaluate its suitability for computational science. A set of benchmarks representing typical building blocks of scientific applications and a real-world hybrid MPI/OpenMP code for ocean simulation are used for performance evaluation. Additionally we apply micro benchmarks to evaluate the performance of certain components (such as the memory subsystem). To recognise the capabilities of the T2+ processor we compare its performance with the IBM POWER6 processor. While the UltraSparc T2+ is targeted on server workloads with high throughput requirements via low-frequency core design and massive chip multithreading capabilities, the ultra-high frequency core design of the IBM POWER6 optimised for instruction-level parallelism follows a contrary approach. The intention of this evaluation is to investigate whether the current generation of massive chip multithreading processors is capable of providing competitive performance for non-server workloads in scientific applications.

Keywords: Sun UltraSparc T2+, Niagara2, Evaluation, Computational Science.

1 Introduction

The improvement of single-core processor performance has reached technological limits known as power, memory, and instruction-level parallelism (ILP) walls. Chip multithreading technologies (CMT) [15] are considered as a viable approach to overcome most of these limitations. CMT is based on the employment of multiple cores per chip usually in combination with simultaneous multithreading (SMT) capabilities. CMT designs adhere to the principle that with lower frequencies and a higher number of SMT enabled cores, performance gains can be achieved while keeping power consumption at modest levels. These CMT/SMT design principles are incorporated in the Sun UltraSparc T2 processor (codenamed Niagara-2) [14].

The Sun UltraSparc T2 was designed as a scalable solution for fast growing datacenter applications, its main target being emerging web- and database markets. The massively multithreaded nature of the T2 chip reflects the market's need for a high throughput solution that can serve hundreds of clients on one single processor die. In April 2008 the T2+ processor [18] was introduced as an SMP extended version of the T2 processor allowing multiple processors to be used within a single system. The achievable savings in

G. Allen et al. (Eds.): ICCS 2009, Part I, LNCS 5544, pp. 964–973, 2009.

system costs per customer make massively multithreaded architectures very appealing to growing businesses, but for computationally intensive scientific computing applications it is unclear if these architectures can provide enough raw computational power.

In this paper we evaluate the suitability of the T2+ processor for computational science. A comprehensive measurement-based performance analysis study of the T2+ processor involves microbenchmarks, a collection of typical scientific kernels, and a real world ocean simulation application. We use microbenchmarks for the performance evaluation of certain system components. This includes an evaluation of the sustainable memory performance with the STREAM memory microbenchmark [8]. The NAS parallel benchmarks [1] serve as representatives of commonly used building blocks for scientific applications. To complement our performance evaluation study we use a real-world hybrid MPI/OpenMP Fortran90 code [4] that simulates the western intensification of wind-driven ocean currents.

To highlight the limitations and capabilities of the T2+ processor we provide a performance comparison with the IBM POWER6 [6] processor. While the UltraSparc T2+ is designed for server workloads with high throughput requirements via low-frequency core design and massive chip multithreading capabilities, the ultra-high frequency core design of the IBM POWER6 optimised for instruction-level parallelism follows a contrary architectural approach. Our aim is to find out if the current generation of massive chip multithreading processors is able to provide competitive performance for non-server workloads in scientific applications.

This paper is structured as follows. Section 2 gives an overview of the tested computing systems. The benchmarks are presented in Section 3. Section 4 presents and discusses the results of our performance analysis. Related work is discussed in Section 5. Finally, Section 6 concludes the paper and briefly describes future work.

2 Experimentation Platforms

In this section we describe the hardware and software environments of the computer systems used in our performance analysis study: (1) Sun T5140 Server and (2) IBM BladeCenter JS22. Table 1 summarises the main features of the analysed systems.

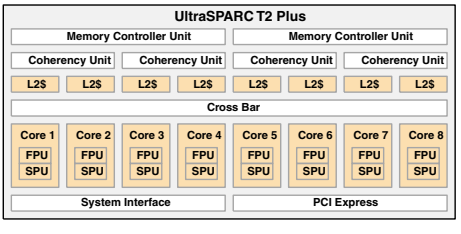


Fig. 1. T2+ processor block diagram[18]

SUN UltraSparc T2 Plus. The Sun T5140 Server [17] is comprised of two Sun UltraSPARC T2 Plus (codenamed Niagara-2) multi-core processors. The T2 Plus processor [14], which was introduced in April 2008, is an SMP extended version of the T2 processor allowing multiple Chip-level MultiThreading (CMT) processors to be used within a single system (shown in Figure 1). In the T2 architecture, massive chip multithreading is favoured over single thread performance. With concurrent execution possibilities of 64 threads per chip sustained processor performance and scalability are crucially dependent on efficient memory access mechanisms. Hence, the eight SPARC cores are

Table 1. Testing environments

System	Sun T5140 Server	IBM BladeCenter JS22
CPU's	2x 1.2 GHz Sun UltraSparc T2+	2x 4 GHz POWER6
Cores (System)	8(16)	2(4)
L1 instr. cache (core)	16KB	64KB
L1 data cache (core)	8KB	64KB
L2 cache (CPU)	4MB	2x 4MB
L3 cache (CPU)	-	32MB
HW Threads (CPU)	8x 8 (SMT)	2x 2 (SMT)
HW Threads (System)	128	8
Memory	8x 4GB 667MHz FBDIMM	8GB DDR2 DRAM
OS	Solaris 10 5/08	AIX 6.1
C Compiler	SUN Studio C 5.9	IBM XLC/C++ 10.1
Fortran Compiler	Sun Fortran 95 8.3	IBM XLF 12.1
Optimization	-O3 (max)	-O5 (max)
MPI	OpenMPI 1.2.5	-

connected via a high bandwidth crossbar to eight memory banks of a shared 4MB L2 cache. Each core can execute up to eight threads simultaneously. One core provides two integer execution units (EXU), one floating point and graphics unit (FPU) and a specialised stream processing unit (SPU) for cryptographic acceleration. The L2 cache banks are connected to two memory controllers (MCU).

IBM POWER6. The IBM BladeCenter JS22 is comprised of two IBM POWER6 multicore processors. In comparison to the UltraSparc T2 the IBM POWER6 microarchitecture [6] follows a contrary architectural approach. Ultra high frequency core design (up to 4.7Ghz) optimised for ILP is favoured over massive chip multithreading (that is TLP). Based on the POWER5 microprocessor, the POWER6 architecture implements two high-frequency *simultaneous multithreading* (*SMT*) cores per chip. SMT enables simultaneous execution of up to two threads per core. With its 4MB core-private L2 caches and a large shared 32MB L3 cache the POWER6 microarchitecture is optimised for performance and computational power. The SMP interconnect facilities enable system configurations of up to 32 POWER6 processors.

Operating system and environment. Our goal was to evaluate the systems under realistic conditions. To achieve this, we used typical vendor setups where applicable. This means, that we used the vendor's operating system and compiler tool-chain (see Table 1), and allowed maximum compiler optimisation. For OpenMP and MPI compilation the corresponding tools and flags in the compiler tool-chain were used.

Scheduling policies. Consistent with our goal to use realistic conditions, we did not manipulate the operating system scheduling policies. With the OpenMP scheduler we also didn't change the policy, except for comparison runs with the NAS Benchmarks on the T2+, where we were interested in a single chip performance evaluation. Since the systems were delivered with SMP dual- chip configurations we needed to delegate the execution of OpenMP [11] threads to a specific processor. This was achieved via the native task schedulers.

3 Benchmarks

In this section we describe the software used for performance analyses. We use a collection of *microbenchmarks* for performance evaluation of individual system components (such as the memory subsystem). The *NAS Parallel Benchmarks* serve to evaluate the performance of typical building blocks that frequently appear in computational science applications. As an example of scientific applications, we use an ocean simulation code with a hybrid MPI/OpenMP programming model.

Microbenchmarks. The *STREAM Memory Benchmark* [8] is a synthetic benchmark that measures the performance of four operations on large vectors: (1) *copy*, (2) *add*, (3) *scale* (involves multiplication operations), and (4) *triad* (combines *copy*, *add* and *scale*). The main goal of the STREAM benchmark is a realistic evaluation of sustainable memory performance. *Cache Bench* is a specialised memory benchmark designed for evaluation of memory hierarchy performance. Different tests are used including *read*, *write*, *modify* operations and the *memset()* and *memcpy()* functions from the C library. Each test is executed using several vector sizes [9]. *DGEMM* [3] tests the floating point performance of a double precision Matrix-Matrix multiplication via hardware optimised Level-3 Blas routines [2].

NAS Parallel Benchmarks. This benchmark suite was originally developed by the Numerical Aerodynamic Simulation (NAS) program at the NASA Ames Research Center for effective evaluation of platforms for computational fluid dynamics (CFD) applications. The NAS parallel benchmark (NPB) suite [1] has proven to be a credible solution for measuring computational performance of parallel computer systems. In our performance evaluation study we use a C version of NPB 2.3 parallelised with OpenMP [11] that was developed by the Omni OpenMP Compiler Project [10]. We have selected four NPB benchmarks: FT, CG, LU, and EP. The FT benchmark executes a 3-D fast Fourier Transform (FFT) based kernel. In the CG benchmark a Conjugate Gradient method is used to approximate the smallest eigenvalue of a sparse matrix. The EP (Embarrassingly Parallel) kernel generates pseudo-random numbers and can be used as an estimation of the systems scalability. In the LU application benchmark a block lower triangular-block upper triangular system of equations is solved [1]. In our performance experiments all benchmarks have been executed with the problem size indicated as *Class B*.

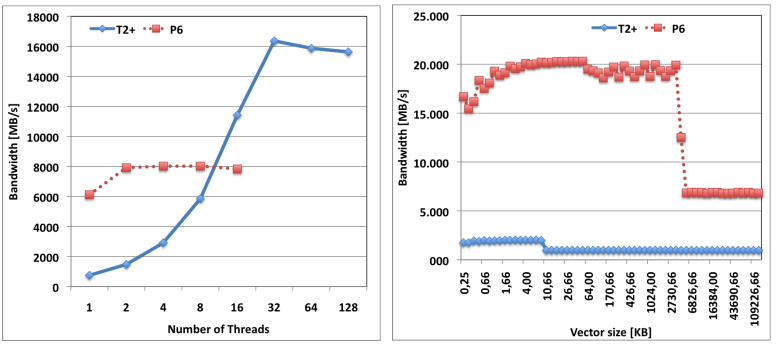
Ocean Simulation. This is a real-world hybrid MPI/OpenMP Fortran90 code [4] that simulates the western intensification of wind-driven ocean currents [16]. A 2D Stommel Model of Ocean Circulation (2D-SMOC) is solved by using a five-point stencil and Jacobi iteration. We use this application to complement the performance evaluation results of Micro- and NAS benchmarks.

4 Results

In this section we present the performance evaluation results obtained using micro benchmarks, NPBs, and the ocean simulation application. Our primary aim is to evaluate the suitability of the T2+ processor for computational science. We use the performance results of the POWER6 processor for comparative purposes only.

4.1 Microbenchmarks

Memory access is a limiting factor of the performance of many scientific computing applications. For an estimation of sustainable memory performance we used the STREAM memory benchmark (see Section 3) with a variable number of threads. All operations were executed on large vectors. The performance results obtained are depicted in Figure 2(a). With increased thread count the T2+ based system is able to constantly increase overall memory bandwidth until up to 16 threads. On the POWER6 system the memory performance gain in multithread runs is less intense. We may observe that for single thread runs the memory bandwidth of the T2+ based system is about five times lower compared to the POWER6.

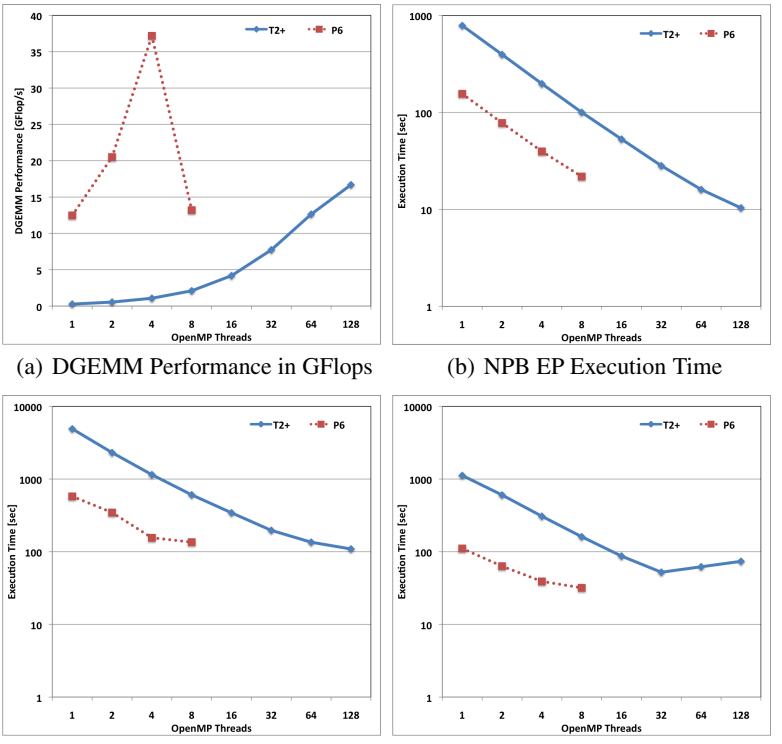


(a) STREAM (Triad) (b) Cachebench

Fig. 2. Performance evaluation of the memory subsystem using STREAM and Cachebench

Figure 2(b) depicts the evaluation of memory bandwidth using Cachebench for various vector sizes. Data was acquired in single thread runs of the read/modify/write test with vector lengths between 256 byte and 100MB. A bandwidth drop can be seen between the L1 (8KB) and the L2 cache on the T2+. The change between the L2 cache and the main memory is less distinctly observable via this test for the T2+. On the POWER6 based system we see drops between L1 and L2 cache at 64KB and between L2 and the main memory at the L2 cache size of 4MB. We may observe that Cachebench performance results of T2+ are significantly lower than POWER6 for all tested vector sizes.

Figure 3(a) depicts the evaluation of sustainable floating point performance using *DGEMM* benchmark. This benchmark uses system vendor SMP optimised Level3-Blas subroutines with a fixed matrix dimension and a variable number of threads. The benchmark results confirm the design principles of the T2+ system with low single thread performance but excellent scaling capabilities. The maximum performance of 37.18 GFlop/s is achieved on the POWER6 system in a 4-thread run. For the *DGEMM* benchmark the use of SMT on POWER6 system shows a steep performance drop when the number of SMT threads is larger than the number of physical cores. This is contrary to the T2+ system where a maximum performance of 16.69 GFlop/s is observable at the maximum number of 128 available SMT threads (please note that our evaluated T2+ based system has 16 physical cores).



(a) DGEMM Performance in GFlops (b) NPB EP Execution Time

(c) NPB LU Execution Time (d) NPB FT Execution Time

Fig. 3. Performance results for the DGEMM, NPB EP, NPB LU, and NPB FT benchmarks

4.2 NAS Parallel Benchmarks

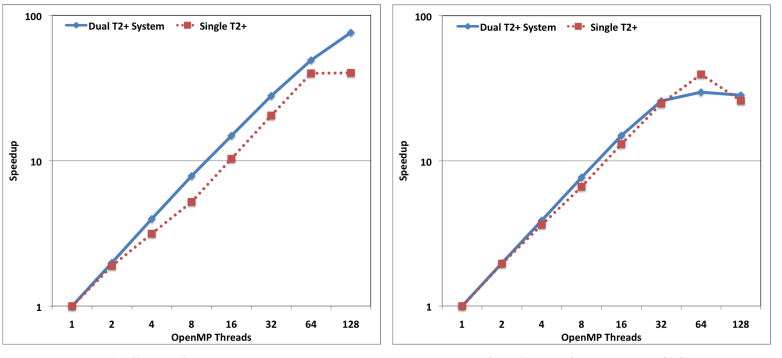
In this section we investigate the performance of a subset of the NPB benchmarks on the T2+ and POWER6 based systems. The aim is to identify NPBs that perform well on T2+ based systems. We used the problem size that in NPB is indicated as *Class B* in all presented experiments. On the T2+ based system the maximum number of SMT threads was set equal to the number of available logical CPUs (128). On the POWER6 based systems we performed experiments up to the limit of 8 SMT threads. With these settings we could investigate performance of the systems under demand for heavy resource sharing and per core context switching.

Figure 3(b) depicts performance evaluation results obtained using the NPB EP benchmark. The NPB EP benchmark generates pairs of Gaussian random deviates. This *embarrassingly* parallel task is obviously very well suited for the massively multithreaded T2+. The T2+ based system exhibits a nearly linear speedup. We may observe that the highest achievable performance of the NPB EP benchmark on T2+ based system is higher than on POWER6. This is due to the larger number of cores and better SMT capabilities of T2+. But, the POWER6 system clearly outperforms the T2+ in single thread performance and computational core power.

The performance evaluation results obtained using the NPB LU benchmark are depicted in Figure 3(c). NPB LU benchmark uses a symmetric successive over-relaxation (SSOR) method to solve a seven-block-diagonal system. A pipelined algorithm is used to solve the triangular systems [5]. We may observe in Figure 3(c) that for large numbers of threads the T2+ based system outperforms the POWER6. However, due to the LU synchronisation overhead, the increase of performance with the number of threads is not as high as for EP.

Figure 3(d) depicts performance evaluation results obtained using the NPB FT benchmark. The FT benchmark performs a 3-D FFT by execution of three 1-D FFT routines. Therefore the 3-D data is copied into 1-D work arrays for each dimension. Before the final 1-D FFTs are performed, array transposes take place [13]. With elevated communication demand the scaling abilities of the T2+ are limited. For FT benchmark the T2+ is unable to reach the POWER6 system's performance in any of our test cases. We may observe that on the T2+ system almost no speedup can be achieved with more than 32 threads.

Figure 4 depicts the speedup of the EP and CG benchmarks on the T2+ based system. For this evaluation we compared the speedup of the system when both T2+ chips are used with the case when only one of the available two chips is used. Although we ran these tests with all problem sizes, only *Class B* tests will be discussed here. The *Class B* provided the most interesting results, by being able to utilise the available resources.



(a) Speedup NPB EP (b) Speedup NPB CG

Fig. 4. The speedup of EP and CG benchmarks on the T2+ based system. Speedup when both available T2+ processors are used is compared to speedup when a single T2+ processor is used.

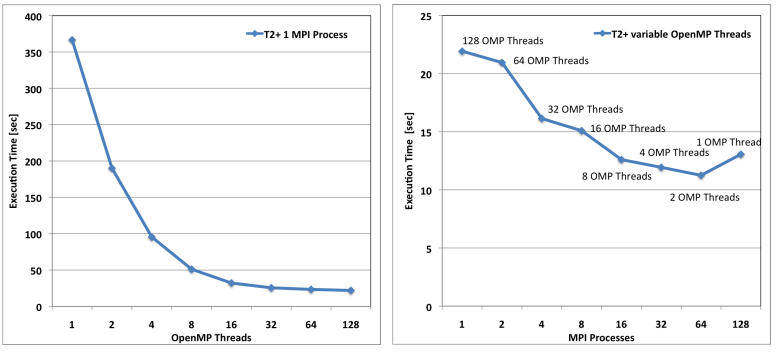
The speedup of EP benchmark is depicted in Figure 4(a). If both T2+ chips are used, the EP benchmark shows the best speedup of 76.15 when executed with 128 threads. Up to 32 threads the speedup is nearly linear. If only one of the available two T2+ chips is used, the speedup is slightly lower and reaches its maximum with 64 threads for both benchmarks. For more than 64 threads there is no performance improvement if a single T2+ chip is used, but also no dramatic performance drop. This is due to the embarrassingly parallel nature of the EP benchmark.

Figure 4(b) depicts the speedup of the CG benchmark. The CG benchmark uses a Conjugate Gradient method to approximate the smallest eigenvalue of a sparse matrix.

Due to the communication demand of the CG benchmark the single chip run can take advantage of data locality. For 64 threads the execution of the CG benchmark on a single chip shows better speedup and performance (measured in absolute execution time) than the case when both chips are used. For more than 64 threads there is a clear performance drop for both cases. In this experiment the SMP interconnect seems to be a bottleneck when a high number of threads is used.

4.3 Ocean Simulation

In this section we investigate the performance of a real-world Ocean Simulation application on our T2+ based system. This application is implemented in Fortran90 and parallelised using MPI and OpenMP (see Section 3). While the initial data partitioning is done via message passing, the Jacobi iterations are parallelised with OpenMP [11]. The nature of this application allows an investigation on the impact of combination of these programming models. Therefore, we conducted multiple performance experiments for various numbers of MPI processes and OpenMP-threads. Figure 5 depicts the performance results obtained on the T2+ based system.



(a) OpenMP (b) OpenMP/MPI

Fig. 5. Performance results for the ocean simulation application

In Figure 5(a) are presented performance results obtained when the number of MPI processes is constant (a single MPI process is specified) and the number of OpenMP threads is varied from one to 128. The results reveal that this application scales well on the T2+ based system when the number of OpenMP threads is varied. The best result (21.92 seconds) is achieved when all 128 OpenMP threads are used.

Figure 5(b) depicts the performance results that are obtained for various combinations of the number of MPI processes and OpenMP threads, where the sum of MPI processes and OpenMP threads is kept constant at 128. The best result (11.25 seconds) is obtained for 64 MPI processes each having two OpenMP threads.

5 Related Work

There exist many different techniques for performance evaluation of highly parallel computer systems. Measurement-based performance evaluation approaches are

commonly used for the evaluation of existing computer architectures, while the model-based approaches are typically used to answer what-if questions for future architectures that are under the development. In [12], a measurement-based approach is used for the evaluation of the Intel Woodcrest processor for scientific computing using a set of benchmarks. An evaluation of the STI Cell/B.E. processor using scientific computing kernels is presented in [20]. In [7], a model-based approach is used for performance evaluation of bioinformatics applications on GPU-accelerated architectures. In [19], a sparse matrix-vector multiply kernel is optimised for different multicore platforms. This approach may have the benefit of discovery of algorithmic optimisations for a specific hardware.

In this paper we present a comprehensive measurement-based performance analysis study of the T2+ processor, which aims at investigating the suitability of this processor for computational science. For this purpose we use, alongside the NAS parallel benchmarks, a collection of microbenchmarks and a real world ocean simulation application.

6 Conclusions

CMT systems like the SUN UltraSparc T2+ are especially well suited for server workloads with a high level of thread parallelism. In our experimental evaluation for the scientific computing domain we observed excellent scaling capabilities on workloads with little or no inter-thread communication such as the NPB EP benchmark. However, for problems with higher communication demand (such as NPB CG benchmark) or unpredictable memory access patterns the T2+ shows performance drawbacks. Due to the high number of threads, increased communication overhead becomes more severe on the T2+ than on high frequency multi-core designs like the IBM POWER6. This issue is further aggravated in SMP configurations where multiple T2+ processors are used. We have observed that for communication-intensive codes such as NPB CG, better performance may be achieved using only one of the two available T2+ processors. Using a real-world ocean simulation hybrid code we conducted multiple performance experiments for various numbers of MPI processes and OpenMP-threads. The best result was obtained for a combination of a rather larger number of MPI processes with a small number of OpenMP threads.

In the future we plan to investigate the performance of emerging multi-core systems using model-based evaluation techniques.

Acknowledgements. The authors are grateful to Martin Wimmer for numerous discussions and helpful comments regarding the work presented in this paper.

References

1. Bailey, D., Harris, T., Saphir, W., Van der Wijngaart, R., Woo, A., Yarrow, M.: The NAS Parallel Benchmarks 2.0. NAS Technical Report NAS-95-020 (December 1995)
2. Dongarra, J., Du Croz, J., Duff, I.S., Hammarling, S.: A set of Level 3 Basic Linear Algebra Subprograms. ACM Transactions on Mathematical Software (March 1990)

3. DGEMM Benchmark, https://computecanada.org/?pageId=138
4. Kaiser, T.H., http://www.sdsc.edu/~tkaiser/stommel.html
5. Jin, H., Frumkin, M., Yan, J.: The OpenMP Implementation of NAS Parallel Benchmarks and its Performance (1999)
6. Le, H.Q., Starke, W.J., Fields, J.S., O'Connel, F.P., Nguyen, D.Q., Ronchetti, B.J., Sauer, W.M., Schwarz, E.M., Vaden, M.T.: IBM POWER6 microarchitecture. IBM J. Res. and Dev. 51(6) (November 2007)
7. Liu, W., Schmidt, B., Mueller-Wittig, W.: Performance Analysis of General-Purpose Computation on Commodity Graphics Hardware: A Case Study Using Bioinformatics. Journal of VLSI Signal Processing Systems 48(3), 209–221 (2007)
8. McCalpin, J.D.: Memory Bandwidth and Machine Balance in Current High Performance Computers. IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter (December 1995)
9. Mucci, B., London, K., Thurman, J.: The CacheBench Report (November 1998)
10. NAS Parallel Benchmarks in OpenMP, http://phase.hpcc.jp/Omni/benchmarks/NPB/index.html
11. OpenMP Application Program Interface. Version 3.0 (May 2008), http://www.openmp.org/mp-documents/spec30.pdf
12. Roth, P.C., Vetter, J.S.: Intel Woodcrest: An Evaluation for Scientific Computing. In: Proc. 8th LCI International Conference High-Performance Clustered Computing (2007)
13. Saphir, W., Van der Wijngaart, R., Woo, A., Yarrow, M.: New Implementations and Results for the NAS Parallel Benchmarks 2. In: 8th SIAM Conference on Parallel Processing for Scientific Computing (1997)
14. Shah, M., Barreh, J., Brooks, J., Golla, R., Grohoski, G., Gura, N., Hetherington, R., Jordan, P., Luttrell, M., Olson, C., Saha, B., Sheahan, D., Spracklen, L., Wynn, A.: UltraSPARC T2: A Highly-Threaded, Power-Efficient, SPARC SOC. Sun Microsystems (2007)
15. Spracklen, L., Abraham, S.G.: Chip Multithreading: Opportunities and Challenges. In: 11th International Symposium on High-Performance Computer Architecture (2005)
16. Stommel, H.: The western intensification of wind-driven ocean currents. Transactions American Geophysical Union. 29, 202–206 (1948)
17. Sun SPARC Enterprise T5140 Server, http://www.sun.com/servers/coolthreads/t5140/
18. SUN SPARC Enterprise T5120, T5220, T5140 and T5240 Server Architecture. White Paper. Sun Microsystems (April 2008)
19. Williams, S., Oliker, L., Vuduc, R., Shalf, J., Yelick, K., Demmel, J.: Optimization of Sparse Matrix-Vector Multiplication on Emerging Multicore Platforms. In: Proceedings of the 2007 ACM/IEEE conference on Supercomputing (2007)
20. Williams, S., Shalf, J., Oliker, L., Kamil, S., Husbands, P., Yelick, A.K.: Scientific Computing Kernels on the Cell Processor. International Journal of Parallel Programming 35(3), 263–298 (2007)