LA-UR-02-4850

Title: | Performance Evaluation of an EV7 AlphaServer Machine

Author(s): | Darren J. Kerbyson
Adolfy Hoisie
Scott Pakin
Fabrizio Petrini
Harvey J. Wasserman

Los Alamos
NATIONAL LABORATORY

Performance Evaluation of an EV7 AlphaServer Machine.

Darren Kerbyson, Adolfy Hoisie, Scott Pakin, Fabrizio Petrini, Harvey Wasserman
Los Alamos National Laboratory,
CCS-3 Modeling, Algorithms and Informatics Group,
Parallel Architectures and Performance Team
Los Alamos, NM 87545
Email: {djk,hoisie,pakin,fabrizio,hjw}@lanl.gov

Abstract

In this paper we detail the performance of a new Alphaserver machine consisting of 16 Alpha EV7 CPUs. This processor is based on the Alpha EV68 processor core as used in the existing Alphaserver ES45 that has been used to build tera-scale systems at Los Alamos and at Pittsburgh. The EV68 processor core is supplemented with an additional six-way router circuitry which enables a 2-D inter-processor torus network to be constructed as well as direct connections to I/O and local memory. A performance evaluation of this machine is reported here which considers memory hierarchy, intra-node MPI communication, and full application performance. Comparisons are also made to existing Alphaserver machines. It is clear from this analysis that this machine achieves an excellent main memory bandwidth of over 4 GB/s. This has a positive impact on application performance on larger problem sizes in comparison to a similar speed EV68 processor.

1. Introduction

This paper details a performance evaluation of a state-of-the-art Alphaserver machine. It represents one of the next generation Alphaserver machines which are designed to scale up to 64 processors within a node. The most significant changes relative to the current Compaq ES40 [1] and ES45 systems include: the upgrade to the EV7 CPU module, PCI-X I/O slots, and a NUMA memory architecture [3].

The EV7 CPU uses the same EV68 core as in the Alphaserver ES45 but also incorporates two on-chip Direct Rambus (RDRAM) memory controllers and a 1.75-MB L2 cache on the chip. The instruction set architecture is identical to that of the EV68; a maximum of two floating-point operations can be executed each cycle, so a 1.2-GHz CPU has a peak theoretical processing rate of 2.4-GFLOPS. However, certain improvements to the core have been made; for example, the EV7 CPU can accommodate 16 concurrent outstanding cache misses (versus 8 for the EV68).

The L1 and L2 cache latencies are the same as they were in the EV68: with an expected 2 cycle latency to the L1 and 12-cycle latency to the L2. The EV7's L2 cache is seven-way set associative and can transfer data to the CPU at 16 bytes/cycle (up to 19.2 GB/s at 1.2-GHz). Note that the previous EV68 L2 cache was much larger (up to 16 MB) but was off-chip and had a maximum transfer rate to the CPU of only 5.3 GB/s. The two EV7 on-chip RDRAM memory controllers support a maximum memory-to-L2 transfer rate of 12 GB/s; this is to be compared with only 2.6 GB/s maximum in the EV68.

The EV7 chip also includes a router with a total of six connections. Four connections go to neighboring processors arranged within a node as a 2-D torus topology. These are capable of running at 6.4-GB/s each. One is an I/O port and the remaining one connects to the local processor resources - the local processor core and its two memory controllers. (Note: this router is similar to the router chip in the SGI Origin2000, the main difference being that in the EV7 it is on-chip rather than as a separate ASIC.) This arrangement is shown in Figure 1.
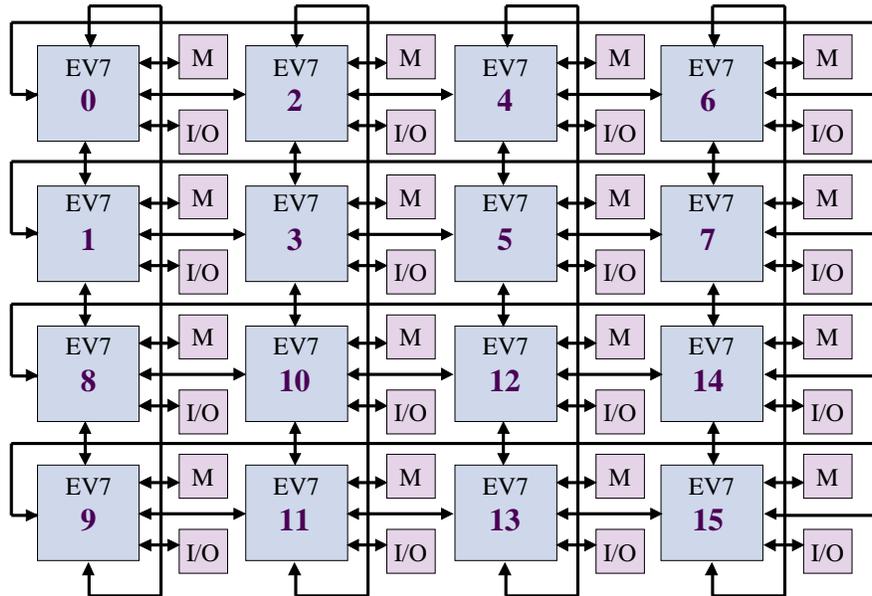
Figure 1 – an example 16 processor EV7 machine indicating processor number ordering

A node is composed of between 1 and 64 EV7 CPUs interconnected via the on-chip routers. The resulting system is a ccNUMA (cache coherent, non-uniform memory architecture) design in which any CPU can access all of the memory in the node but in which the memory access *time* differs depending on where the data are located (also similar to the SGI Origin2000). The latency to local memory was measured at 83ns. Each reference to non-local memory pays this same 83ns penalty, plus about 30ns of overhead getting in and out of the network, plus about 18ns per hop of mostly wire and router delay - a total delay of about 140ns to read the memory on a node that's one hop away. Each additional hop adds another 35-36ns of delay on average.

Therefore, the *worst-case* latency on a 64-CPU node is roughly 390ns (about five times worse than the best-case). The *average* memory latency across such a machine is approximately 260ns, which compares favorably with the (uniform) memory latency of 170ns on an Alpha (EV68) ES45 4-processor system. Smaller nodes will have smaller average latencies (e.g., about 225ns on a 32-processor system).

The EV7 machine analyzed here consisted of a single node containing 16 processors with a clock speed of 1.2-GHz. Three sets of tests were used to analyze its performance. These are:

      (1) memory hierarchy performance micro-benchmarks,
      (2) intra-node MPI communication kernels, and
   (3) full-application codes.

The performance of the memory hierarchy is detailed in Section 2, the performance of the intra-node MPI communication is detailed in Section 3, and the performance of several full application codes are detailed in Section 4. A comparison is made between the measured performance on this machine to that measured on existing Alphaserver machines in Section 5.

2. Memory Hierarchy Performance

The performance of the memory hierarchy is analyzed here both in terms of the latency of the various memory levels in the machine, and also the bandwidth possible to local processor memory and to remote memory within a node.

2.1 Memory Latency

The memory latency was measured by performing a read from a vector in which successive reads are from elements a cache-line length apart. This guarantees that each memory access will exhibit a latency cost as no spatial cache-reuse will occur. By increasing the size of the vector, the latency to different parts of the memory hierarchy can be observed. In addition the memory vector can be placed on a pre-determined PE (Processing Element) and read from on a further pre-determined PE – thus latency to memory on remote processors can also be observed.
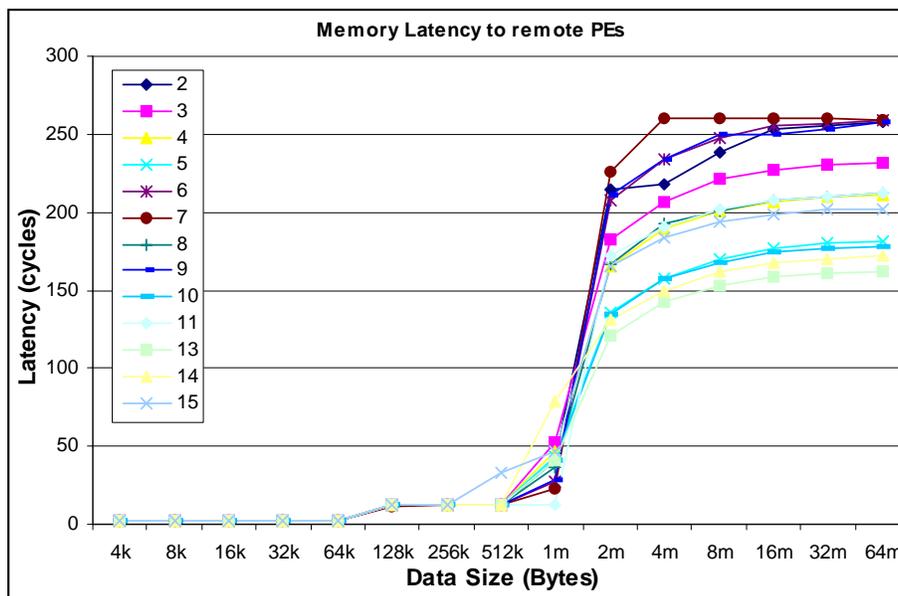
Figure 2 – Memory Latency to Remote Processors from Processor 12.

Figure 2 indicates the latency observed when the memory read is performed on processor 12 and memory is placed on processors 2-15 respectively. The latency for remote memory access can be seen on the larger data vector sizes. On smaller vector sizes, the data will reside in the local processor cache and thus appear the same for all remote processors.

The latency increases as the distance (processor hops increases). It also depends on the route taken, for instance two processors are contained on a single board from a vertical processor pair in within the node, and has less latency than to a processor on a different board. A summary of

3

the latency on the 64Mbytes data vector size is indicated in Figure 3 below. The processor hop distance is shown in Figure 4 for this study (distance from processor 12). The processor ID layout is as shown in Figure 1.

290	258	211	259
260	232	181	259
212	178	106	172
258	212	162	202

Figure 3 –memory latencies (clock cycles) for processor 12

| 4	3	2	3
3	2	1	2
2	1	0	1
3	2	1	2

Figure 4 – distance (hops) from processor 12

2.2 Memory Bandwidth

Cachebench [5] was used to measure the memory hierarchy bandwidth performance within a single node. Two sets of tests were performed. The first test measured the peak memory performance on a single processor for: read, write, read-modify-write, memset, and memcpy. These are shown in Figures 5 and 6. All results are measured for a vector of varying size.

A second test measured the bandwidth performance on a single processor while a number of other processors within the node performed background reads each to their local memory to measures the effective bandwidth in the presence of possible contention.
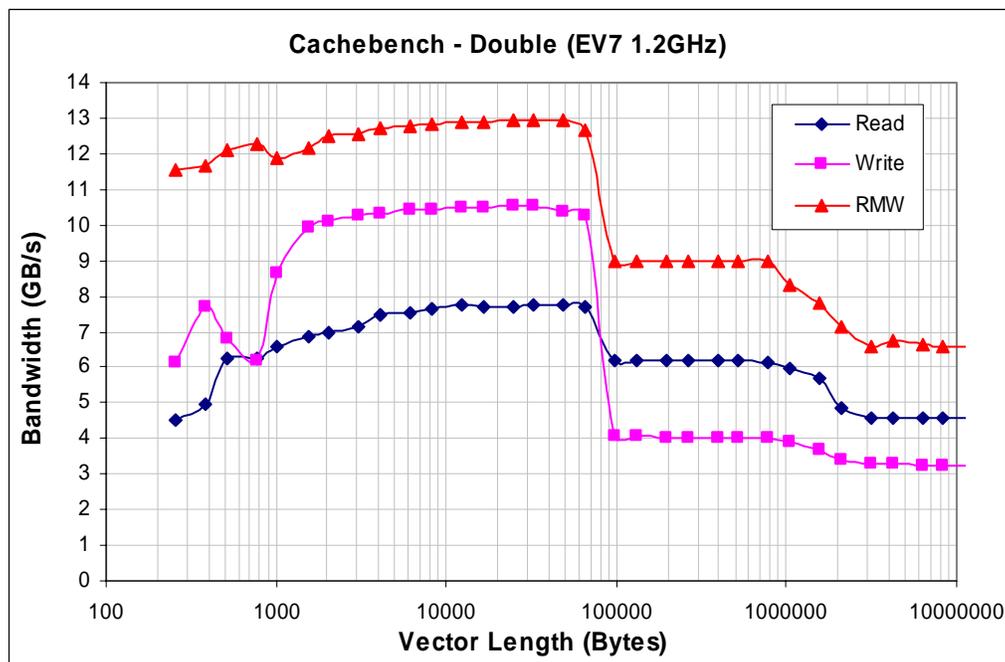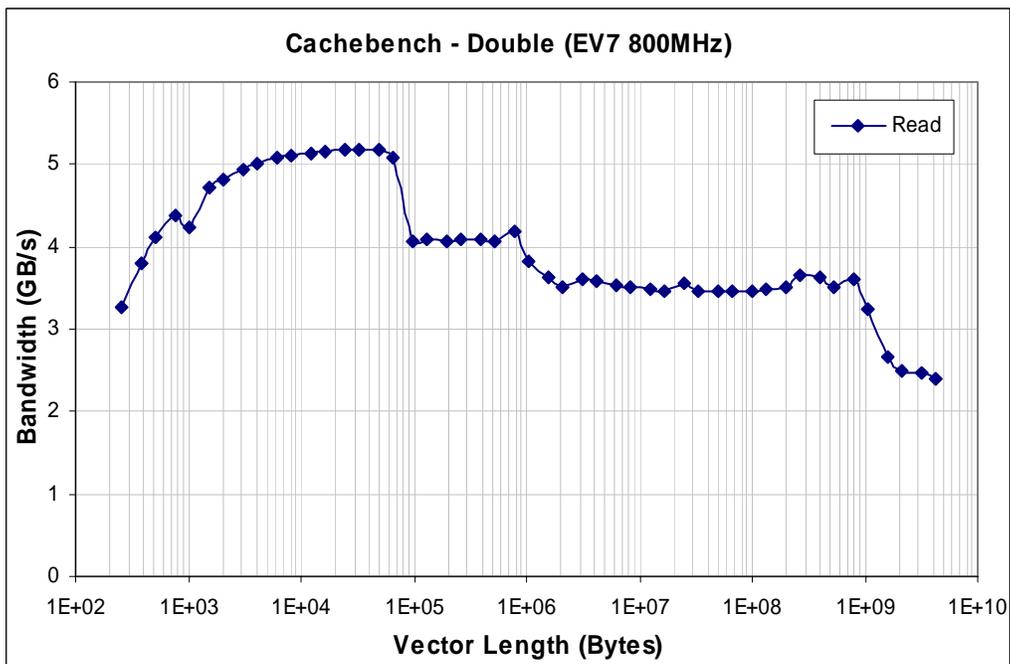

Figure 5 – Achievable peak memory bandwidths (read, write, Read-Modify-Write).
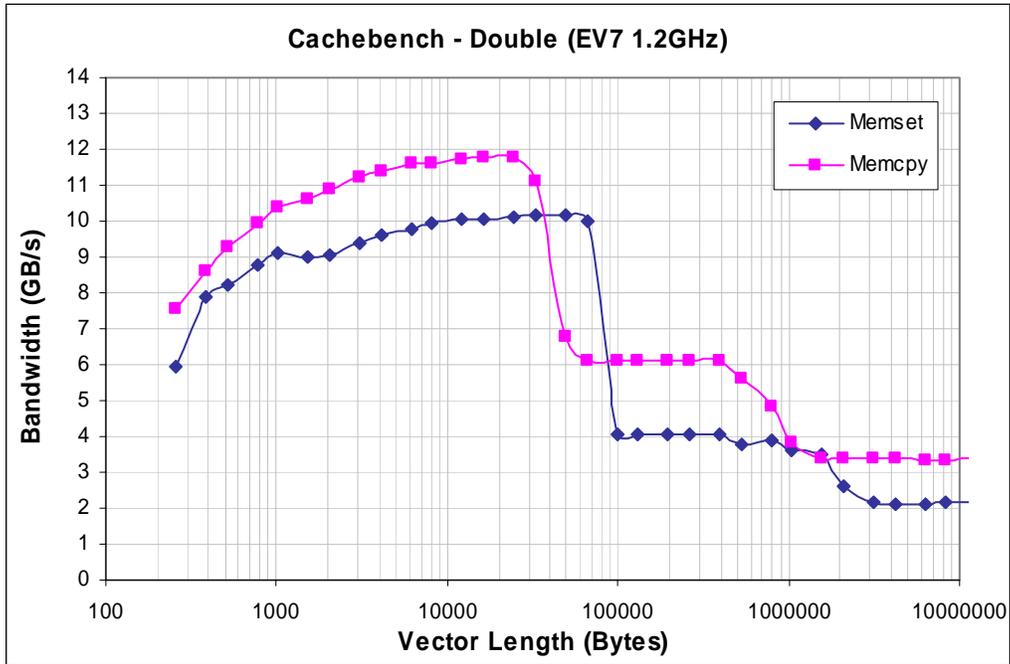
Figure 6 – Achievable peak memory bandwidths (memset, memcpy).

In addition a vector of size larger than the local processor memory was allocated and a read operation performed. This test results in memory accesses to all the available local memory along with a proportion of remote memory. The bandwidth obtained is shown in Figure 7. Note that this test was performed only on an 800-MHz EV7 machine.

Figure 7 – Achievable peak memory bandwidth (read) illustrating remote memory access on large vector sizes.

The memory bandwidth observed for the EV7 machine is very good and is summarized in Table 1. Note that the size of each level of the memory hierarchy can be seen in Figure 7 by the step changes in the memory bandwidth performance.

	Peak bandwidth (GB/s)	Latency (cycles)
L1 (64KB)	7.77	2
L2 (1.75MB)	6.20	12
Main memory (2GB)	4.60	106
Remote memory	~3.60	162-290

Table 1 – Memory performance summary.

The results show that there is only less than a factor of 2 bandwidth reduction between L1 and main memory illustrating a major strength of this machine.

It is expected that the performance of memset corresponds to that of the write performance, and that the performance of memcpy corresponds to that of the read-modify-write performance. As can be seen from Figures 5 and 6, they are indeed similar for L1 cache performance but both under-perform on main memory indicating a better implementation may be possible.

The impact of having many background processors performance memory reads did *not* have an impact on an individual processors memory performance. This is unlike many of the existing smaller SMP nodes such as the Alphaserver ES45 which can be effected by a reduction in bandwidth by a factor of 2 due to memory bus contention. This data has not been included here.

3. Intra-node Communication Performance

The achievable intra-node communication performance was measured using a number of MPI based tests. These included:

- Ping-pong message performance between two adjacent processors. This was measured for both uni-directional and bi-directional message traffic, recording both message latency and bandwidth.

- Message latency and bandwidth between a single processor and all other processors in the node to indicate the performance of between non-adjacent processors.

- Hot-spot communication performance – the achieved communication bandwidth when more than one processor communicates to a single processor.

- Barrier performance – latency for MPI barrier

- Broadcast performance – achieved bandwidth for MPI broadcast

3.1 MPI Communication Performance

The performance of both uni-directional and bi-directional MPI communication between two adjacent PEs using a ping-pong test is shown in Figures 8 and 9. Figure 8 shows the duration (latency) and Figure 9 shows the achieved bandwidth for messages of size between 1 and 1,000,000 bytes.
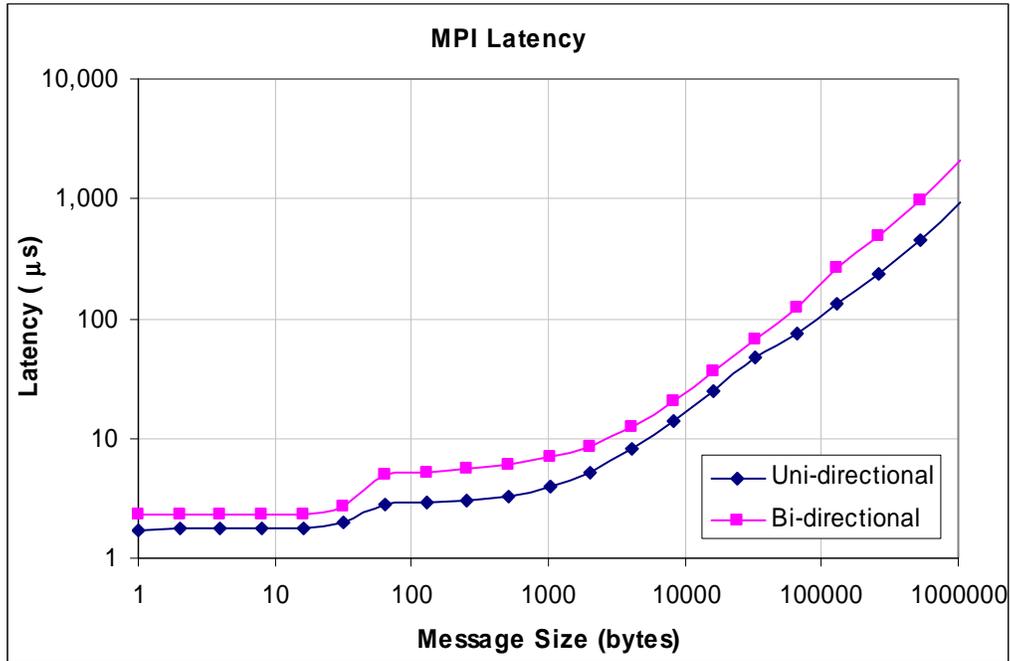
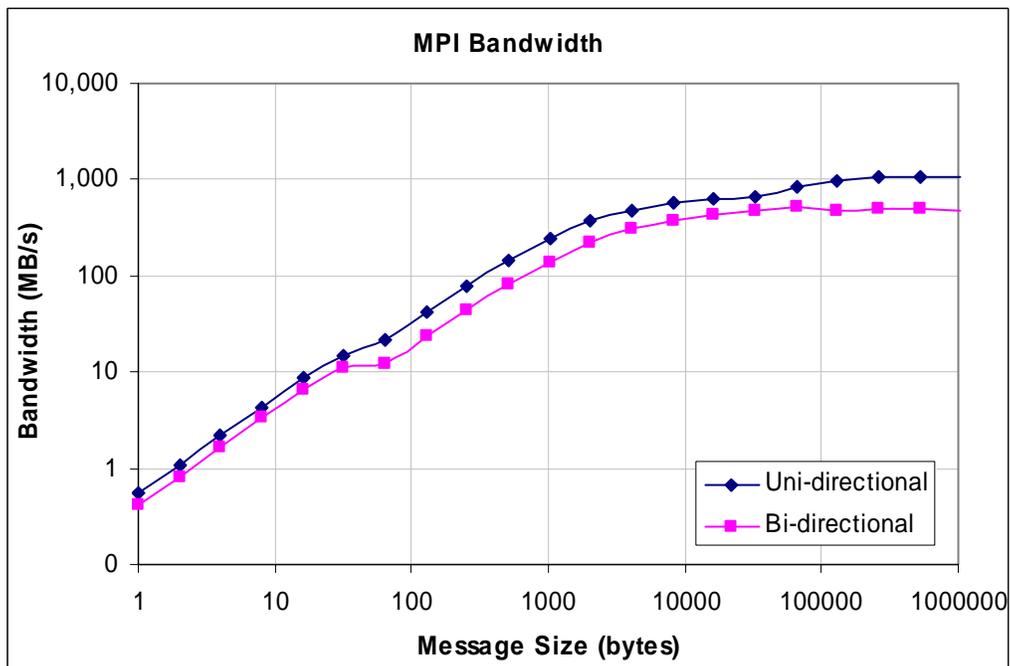**Figure 8** – MPI message latency between two adjacent PEs.

Figure 9 – MPI ping-pong message bandwidth between two adjacent PEs.

The achieved latency of a small message was 1.7µs for a uni-directional message and 2.2µs for bi-directional messages. The ping-pong bandwidth on a message of size 1,000,000 bytes was 1.08GB/s for a uni-directional message and 485MB/s for bi-directional messages. Note that the bi-directional bandwidth is quoted for the achieved bandwidth for each direction in the communication. The bi-directional bandwidth in each direction is just under half of the uni-directional bandwidth.

3.2 Point to Point Communication Performance within a node

The performance of a uni-directional communication using a Ping-Pong test was recorded for all PEs within a node communicating with PE 0. The latency obtained (for a 0 sized message) and the bandwidth achieved on a message of size 1MB are shown in Figures 10 and 11 respectively.

-	1.6	1.8	2.0
1.3	2.3	2.1	2.0
2.2	2.4	2.4	2.5
2.3	2.4	2.5	2.4

Figure 10 – MPI latency (µs).

-	1.14	1.12	1.14
1.14	1.13	1.13	1.13
1.14	1.13	1.11	1.13
1.13	1.12	1.12	1.13

Figure 11 –MPI bandwidth (GB/s).

Both figures show a 4x4 processor map of the PEs in a node. The latency increases as the distance (in processor hops) increases. In fact 2 processors are engineered on the same board in the machine. Processors on the same board have a slightly lower latency than those that are not. In the processor maps shown in Figures 10 and 11 each vertical pair of processors reside on the same board, thus the latency between PE 0 (top-left) and PE 1 (second from top on left) is smaller than the latency between PE 0 and PE 2 (top, second from left). Also note that the PEs within a node are connected in a 2-D torus topology and thus the PE (lower-right) is only 2-hops distant from PE 0.

The bandwidth between PE 0 and any other PE is approximately a constant at 1.13GB/s. The processor ID layout is as shown in Figure 1, and the distance in hops for this experiment is shown in Figure 12.

0	1	2	1
1	2	3	2
2	3	4	3
1	2	3	2

Figure 12 – distance (hops) from processor 0.

3.3 Hot-spot Communication Performance

The achieved bandwidth performance under the hot-spot communication traffic is shown in Figure 13. Hot-spot tests the situation when 1 or more PEs simultaneously communicates to a single PE in a repetitive mode. In the case shown in Figure 13 one or more PEs sent a message of size 256KB to processor 0.

The achievable bandwidth on this test actually increases as more processors perform the simultaneous communication approaching a maximum of just over 1.9GB/s. This indicates that the available bandwidth exceeds that which can be used by a single pair of processors.
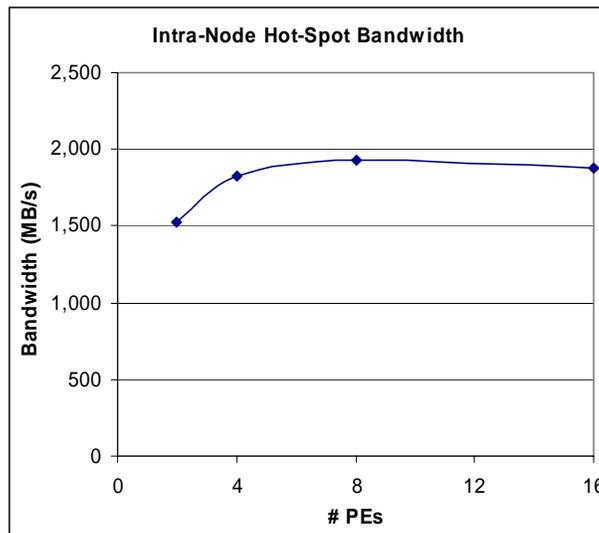
Figure 13 – Achievable bandwidth under the Hot-Spot traffic

3.4 MPI Barrier Latency and Broadcast Bandwidth.

The performance of MPI barrier is shown in Figure 14. The in-node barrier takes 11.2µs on the 1.2-GHz machine. This is actually larger than a Quadrics QsNet based barrier which takes 7µs for an internode barrier on 512 nodes [6]. The performance of MPI broadcast is shown in Figure 15. The achievable bandwidth decrease as more processors are involved in the broadcast. This is due to the broadcast operation relying on messages to propagate through the 2D torus topology. The Quadrics QsNet uses additional hardware support to improve its barrier and broadcast performance. The bandwidth decreases from 1.01GB/s on 2 PEs down to 300MB/s when using all 16 PEs in the EV7 machine.

9
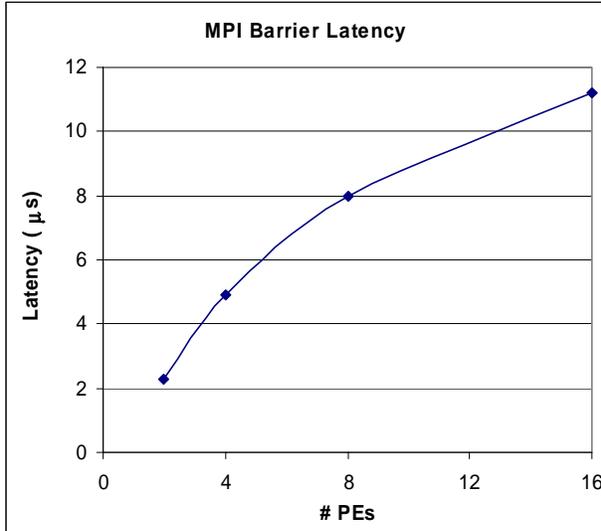
Figure 14 – MPI barrier performance
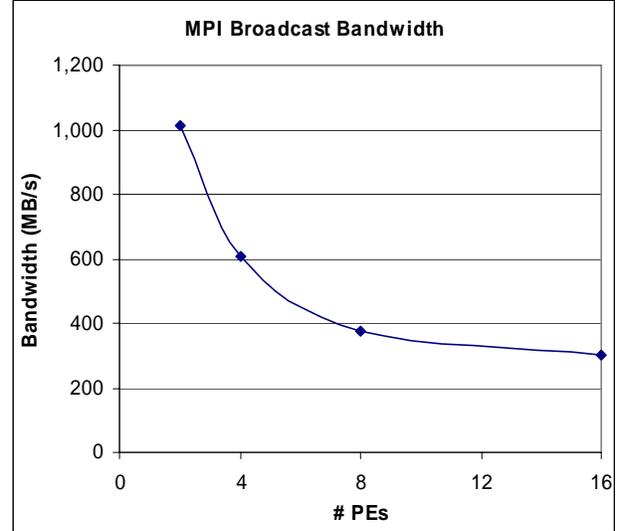
Figure 15 – MPI broadcast performance.

4. Application Performance

The performance of several applications of interest to Los Alamos National Laboratory were measured on the EV7 Alphaserver machine. Performance is detailed here for SAGE, MCNP, and SWEEP3D. SAGE is a multidimensional multi-material hydrodynamics code with adaptive mesh refinement [7]. MCNP is a general purpose Monte-Carlo N-Particle that can be used for neutron, photon, electron, or coupled transport [4]. SWEEP3D is a time independent, Cartesian-grid, single-group, discrete ordinates deterministic particle transport code [2]. Each application was executed in a number of different configurations as described below.

4.1 SAGE

The performance of SAGE was examined in two different studies. The first considers a sequential test whilst varying the size of the spatial grid in terms of the number of cells processed in a single iteration of the code. This is to examine the impact of the memory hierarchy as it is possible for small spatial grids to be L2 cache resident whereas larger grids are not. The second study considers a single spatial grid size whilst varying the number of processors used in a weak scaling study (i.e. keeping the number of cells per processor at a constant). Both studies are described below.

i) SAGE – Cells per PE scaling

This is a sequential test of SAGE whilst scaling the number of cells in the spatial grid. The number of cells was varied from 14^3 to 58^3. Note that SAGE uses a 3D spatial cube by default – hence the number of cells were varied as a cubic power. The result of this scaling is shown in Figure 16 using the number of cells that can be processed in one second (CC/s) as a metric. Ideally this should be a constant for all problem sizes.
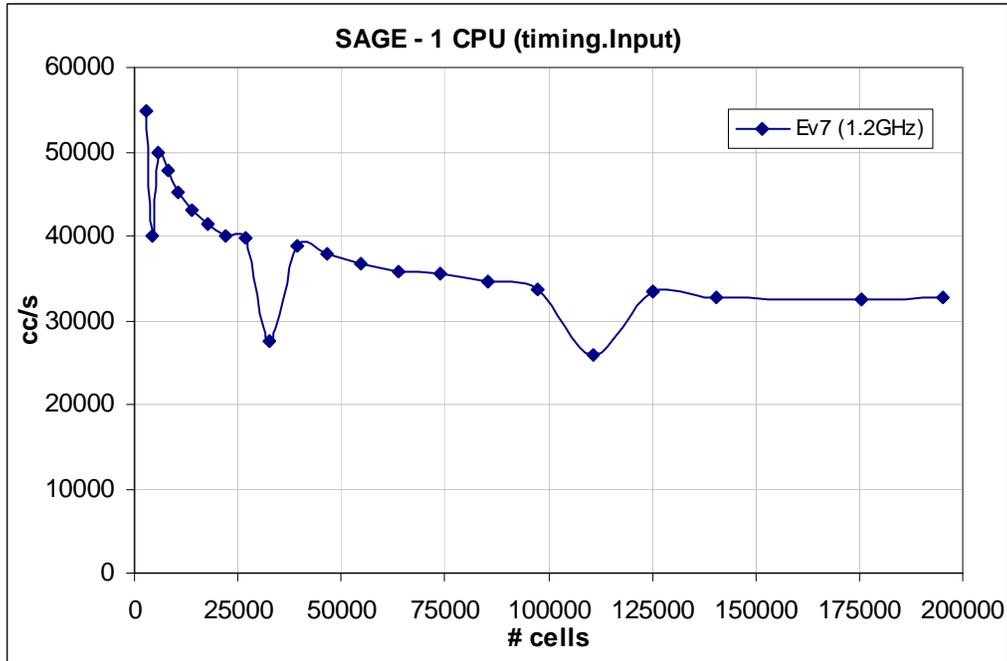
Figure 16 – Sequential Performance of SAGE when varying the spatial grid size.

The performance degrades as the number of cells increases. This is expected due to the limited capacity of the cache. On the smaller problem sizes a large utilization of the L2 cache is possible. On the larger problem sizes very little re-use of L2 cache is possible and hence resulting in a large utilization of main memory.

The performance levels off above 125,000 cells (when main memory is mainly utilized). The dips in performance are due to the number of cells being close to or an exact function of 2 (for instance 4096, and 32768). Having such a number of cells results in poor cache performance resulting from ping-pong interference causing a higher degree of cache misses.

The performance decrease over the range of cell numbers is only 40%. This is a small decrease and is attributed to the good main memory bandwidth of the EV7 machine.

ii) SAGE - Scalability

A scalability test of SAGE was performed on between 1 and 16 processors contained within the EV7 node. The number of cells per processor was set at a constant of 13,500 and thus resulted in a weak-scaling study.

Results are shown in Figure 17 using the number of cells processed in one second per processor (CC/s/pe) metric. The CC/s/pe should ideally be constant but decreases due to parallel overhead.
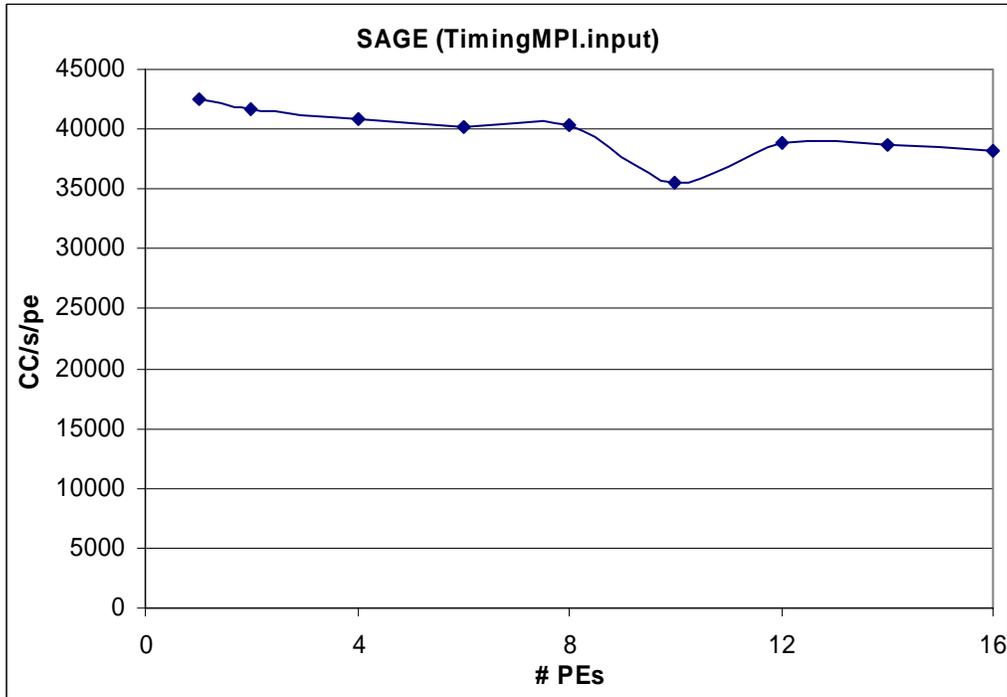
Figure 17 – Scaling behavior of SAGE on the EV7 machine (in node) using the CC/s/pe metric

It can be seen from Figure 17 that the performance decrease up to 16 processors is small (from 42,500 on a single processor down to 38,200 on 16 processors). This represents a high 90% efficiency on 16 processors. The unexpected performance on 10 processors can most likely be attributed to a poorer cache utilization that can occur from the only approximate weak scaling behavior of SAGE.

4.2 MCNP

The performance of MCNP is examined here in a strong scaling study. The number of particles that are processed in each cycle was set at a constant and divided up across the number of slaves available for processing. The geometry in which the particles move was replicated across the processors being used. The processing of each particle within a cycle is independent and thus communication occurs at the start and end of a cycle between all slaves and a master processor. The number of cycles, C, and particle histories per cycle, nps, was set to be (C=1010, nps=1,000) and (C=210, nps=10,000) in two separate scalability tests.

The time per particle history while varying the number of slave processors used is shown in Figure 18. This is effectively the grind time of this code. Note that the number of processors used in each case is actually the number of slaves + 1 (i.e. plus a master processor who accumulates results).

MCNP Ev7 (CritExp)
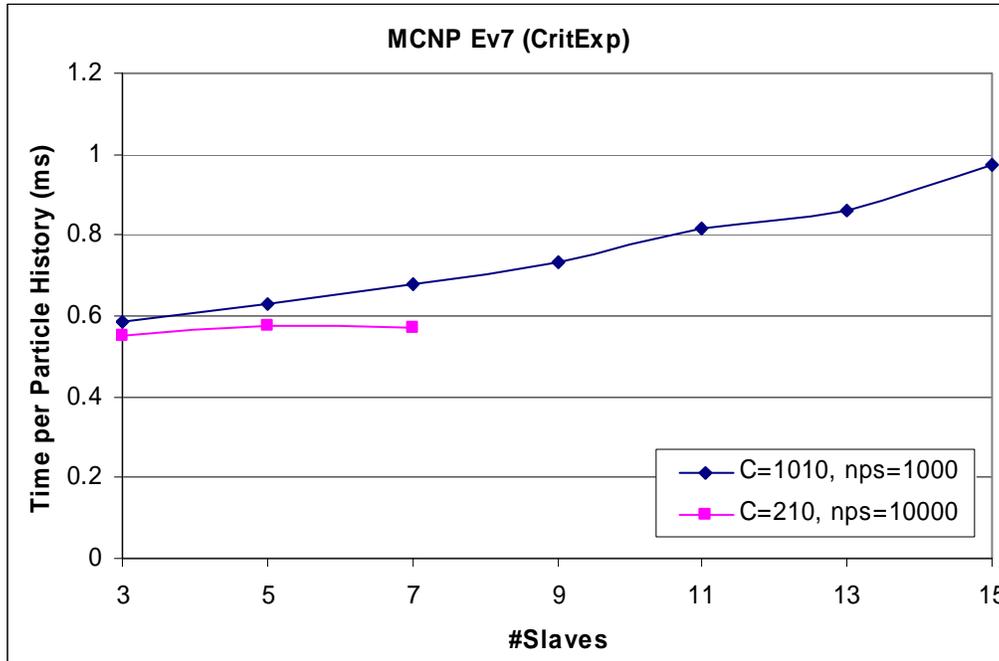
Figure 18 – Time per particle history in MCNP using the critexp input deck on the EV7 machine

The code is subject to a high degree of communication on the smaller problem (C=1010, nps=1,000) resulting in a low efficiency when using all 15 slaves (<60%).

The scalability of the larger problem (C=210, nps=10,000) is expected to be much better due to a decrease in the degree of communication. The time per particle history when using 3 slave processors on the larger problem is ~0.55ms on the EV7 machine. All configurations of the larger problem sizes were not run, but the few measurements made indicate a better scaling behavior than the smaller problem.

4.3 Sweep3D

The performance of Sweep3D was also measured on the EV7 machine. A problem of size 50-cubed with 1-k plane per block and 1 angle per block was run. The total run time was measured in a weak scaling analysis. Observed parallel efficiency on 16 processors was about 90% on the EV7 machine. The single-processor time on EV7 was found to 30% faster than the single-processor time on an EV68 (1GHz) ES45 processor.

5. Comparative Performance

The results of the performance tests measured on the EV7 machine are compared with the performance obtained on current Alpha-systems in this section. The comparisons are made on a like for like basis unless stated. A comparison of performance is shown in Table 2. Currently no MPI performance is available for the 1.25GHz ES45, and no inter-node MPI performance is available on the EV7.

Note that the benchmarked EV7 system was a prototype, and that the current performance may not reflect the achieved level of performance possible on production systems after system and application tuning.

	ES40 [1] (EV68)	ES45 (EV68)	ES45 (EV68)	EV7
System Characteristics				
Clock	833 MHz	1 GHz	1.25 GHz	1.2 GHz
Node size (CPUs)	4	4	4	16
L1 Cache	64 KB	64 KB	64 KB	64 KB
L2 Cache	8 MB	8 MB	16 MB	1.75 MB
Memory Performance				
Latency (cycles): L1	2	2	-	2
L2	12	19	-	12
Main	168	170	-	106
Remote Memory	-	-	-	162-290[1]
Read Bandwidth (GB/s): L1	4.93	6.47	7.89	7.77
L2	3.97[2]	6.07[2]	7.52[2]	6.20
Main	1.70[2]	2.27[2]	2.27[2]	4.58
Remote Memory	-	-	-	3.60
MPI performance				
Intra-Node (Point to Point)				
Uni-directional: Latency (μs)	6.2	4.9	-	1.7
Bandwidth (MB/s)	695	792	-	1,080
Bi-directional: Latency (μs)	12.7	8.9	-	2.2
Bandwidth (MB/s)	317	379	-	485
Inter-Node[3] (QsNet – Elan3)				
Uni-directional: Latency (μs)	5.6	4.5	-	-
Bandwidth (MB/s)	199	293	-	-
Bi-directional: Latency (μs)	9.8	7.4	-	-
Bandwidth (MB/s)	79	132	-	-

Table 2 – Comparison of various performance characteristics of Alpha machines.

Notes on Table 2:

1 – Remote memory latency on the EV7 varies on the distance (PE hops) between data locality and PE accessing data.
2 – The memory bandwidth on the ES40 and ES45 decrease depending on the number of PEs simultaneously accessing memory (a decrease up to a factor of 2 is possible). No decrease occurs on the EV7.
3 – Peak values for inter-node Latency and Bandwidths are quoted. These can decrease depending on distance between nodes, and physical lengths of wires used.

The performance comparison show a number of significant performance improvements of the EV7 1.2-GHz machine in comparison to the existing EV68 ES45 1-GHz machine. These can be summarized as:

- the main memory bandwidth is a factor of 2 better
- in-node MPI latency is almost a factor of 3 better,
- in-node MPI bandwidth is 30% better

6. Summary

The performance evaluation of the EV7 Alphaserver has shown that the machine has an excellent main memory bandwidth which is almost a factor of two greater than existing systems. In addition, there is only a factor of 2 decrease in the memory read bandwidth between L1 cache at 7.77GB/s and main memory at 4.6GB/s. The bandwidth from remote memory within the node is also high at approximately 3.6GB/s. The small L2 cache (1.75MB) will have a negative impact on application performance on larger problem sizes.

The MPI communication performance compares well with current systems - point-to-point message latency between adjacent processors is low at 1.7µs and bandwidth between adjacent processors is just over 1GB/s. However, the latency is high when compared to remote memory latency (1.7µs vs. 135-240ns), and the bandwidth is low when compared to peak remote memory bandwidth (1GB/s vs. 3.6GB/s).

The MPI latency increases as the distance between processors increases with the maximum being 2.4µs. The bandwidth between any two processors is a constant (at just over 1GB/s). However, the barrier latency was 11.2µs for all 16 processors – this seems large when compared with clusters interconnected with Quadrics QsNet [6]. The broadcast bandwidth also decreases as the number of processors increase due to a lack of hardware support – the bandwidth for all 16 processors was 300MB/s.

The application performance showed that in-node scaling was good resulting in high efficiencies on most codes (90% at 16 processors for SAGE, and SWEEP). On a detailed analysis of scaling the spatial grid in SAGE, the performance decrease from running a small problem (cache bound) to a large problem (main memory bound) was only 24%.

Due to the excellent main memory bus bandwidth, a higher performance should be achievable on the EV7 machine in comparison to a similarly clocked existing Alphaserver ES45.

Acknowledgements

The authors which to thank Richard Foster, Niraj Srivastava, Zarka Cvetanovic, and Ed Benson for access to the EV7 machine, and technical discussions on the performance of the Alphaserver systems.

References

[1]   Z. Cvetanovic, R.E. Kessler, "Performance Analysis of the Alpha 21264-based Compaq ES40 System", Procc. 27th ISCA, Vancouver, June 2000, pp. 192-202.

[2]   K.R. Koch, R.S. Baker, R.E. Alcouffe, "A Parallel Algorithm for 3D Sn Transport Sweeps", LA-CP-92-406, Los Alamos National Laboratory, 1992.

[3]   K. Krewell, "Alpha EV7 Processor: A High Performance Tradition Continues", Microprocessor Report, San Jose, April 2002.

[4]   G. McKinney, "A Practical Guide to Using MCNP with PVM", Trans. Am. Nucl. Soc. 71, 397, 1994

[5]   P. Mucci, K. London, "Low Level Architectural Characterization Benchmarks for Parallel Computers", Technical Report UT-CS-98-394, University of Tennessee, 1998.

[6]   F. Petrini, W.C. Feng, A. Hoisie, S. Coll, E. Frachtenberg, "The Quadrics Network: High-Performance Clustering Technology", *IEEE Micro*, 22(1), 2002, pp. 46-57.

[7]   R. Weaver, Major 3-D Parallel Simulations, BITS -Computing and communication news, Los Alamos National Laboratory, June/July, 1999, 9-11, http://www.lanl.gov/orgs/cic/cic6/bits/99june_julybits/opener.html