

A Performance Comparison of HPCx Phase 2a to Phase 2

A. Gray¹, M. Ashworth², S. Booth¹, J.M. Bull¹, I. Bush², M. Guest²,
J. Hein¹, D. Henty¹, M. Plummer², F. Reid¹, A. Sunderland², A. Trew¹

¹*EPCC, The University of Edinburgh, James Clerk Maxwell Building,
Mayfield Road, Edinburgh, EH9 3JZ, UK*

²*CCLRC Daresbury Laboratory, Warrington WA4 4AD, UK*

February 20 2006

Abstract

The results of benchmarking communications and several popular applications on HPCx Phase 2a are presented and compared with those results from running on HPCx Phase 2. In terms of communications, there is little performance difference between the platforms. In general, the majority of application types perform up to a factor of 2 better on Phase 2a, due to the improved memory architecture. An exception is the classical molecular dynamics class of applications, several of which are studied in this paper. This type of code is not performing as well as expected on Phase 2a, sometimes slightly slower than on Phase 2. This may be explained by sensitivity to an increased latency in some part of the memory subsystem.

This is a Technical Report from the HPCx Consortium

© HPCx UoE Ltd 2005

Neither HPCx UoE Ltd nor its members separately accept any responsibility for loss or damage from the use of information contained in any of their reports or in any communication about their tests or investigations.

1 Introduction

HPCx, the national UK academic supercomputing facility, recently upgraded from Phase 2 (IBM Power4+ technology) to Phase 2a (IBM Power5 technology). This paper compares results from communication and application benchmarks on the new system to those results obtained before the upgrade.

Section 2 gives an overview of the architecture both before and after the upgrade. The communication and application benchmark results are presented in Sections 3 and 4 respectively.

2 Architecture

The HPCx Phase 2 system featured 50 IBM p690+ compute nodes, each containing 32 IBM Power4 64-bit RISC processors. Each processor operated at 1.7GHz. The 32 processors in a compute node were packaged into 4 Multi-Chip Modules (MCMs), each containing 8 processors. Each processor had a private L1 instruction cache of 64KB and L1 data cache of 32KB. The L2 combined data and instruction cache of 1.5MB was shared between 2 processors. Each MCM was equipped with a L3 cache of 128MB (i.e. this was shared between 4 processors). Within a node, MCMs communicated via shared memory. Communication between nodes was provided by an IBM High Performance Switch (HPS). A total of 4 inter-node links were available: each node had 2 network adapters, each with 2 links.

In November 2005 the HPCx service was upgraded to utilise IBM Power5 technology. This new “Phase 2a” system features 96 IBM eServer 575 compute nodes, each containing 16 1.5GHz IBM Power5 64-bit RISC processors. The 16 processors in a compute node are packaged into 2 MCMs. Each processor has a private L1 instruction cache of 64KB and L1 data cache of 32KB. The L2 combined data and instruction cache of 1.9MB is shared between 2 processors. The L3 cache of 36MB is shared between 2 processors, and is closer to the processor than for Phase 2. Within a node, MCMs communicate via shared memory and the same HPS as for Phase 2 provides communication between nodes.

Although the clock rate of the processors has decreased, it is expected that the improved memory architecture will result in higher memory bandwidths and hence facilitate a performance improvement in applications. Such improved memory bandwidths are observed from stream benchmarks [1]. For example [2], for the scale benchmark the main memory bandwidth has increased from just over 2 GB/s per processor on Phase 2 to just under 4 GB/s per processor on Phase 2a. Even higher increases are seen in the bandwidths from the different cache levels.

3 MPI Benchmarks

The Intel (formerly Pallas) MPI Benchmark Suite [3] was used to study communications. Point to point communications were studied with the use of the PingPong and Multi PingPong benchmarks. The collective communication Bcast, Allreduce and Barrier benchmarks were also run.

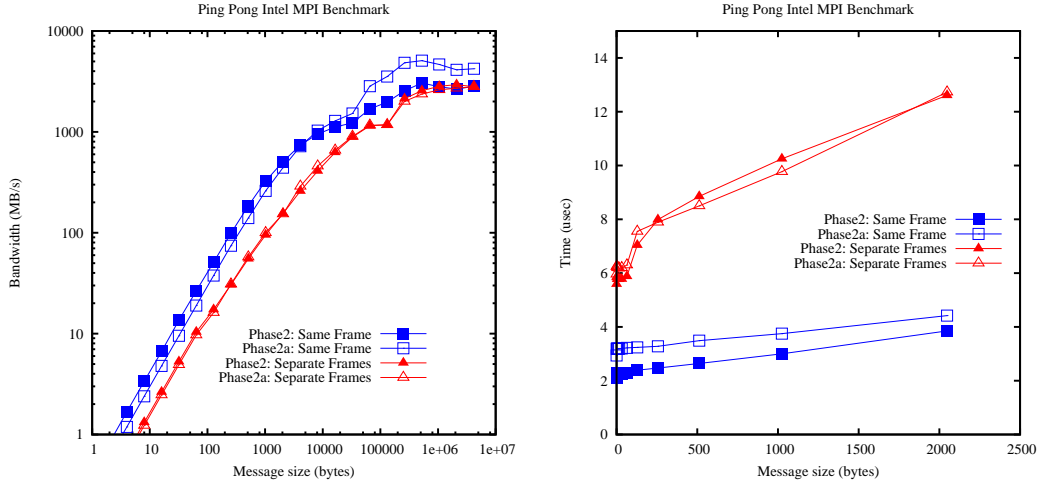


Figure 1: The dependence of bandwidth on message size (left), and time on message size for the small message size region (right) for the Ping Pong benchmark. Phase 2 results are represented by closed shapes and Phase 2a results by open shapes. Communications within a node are represented by squares and those between nodes by triangles.

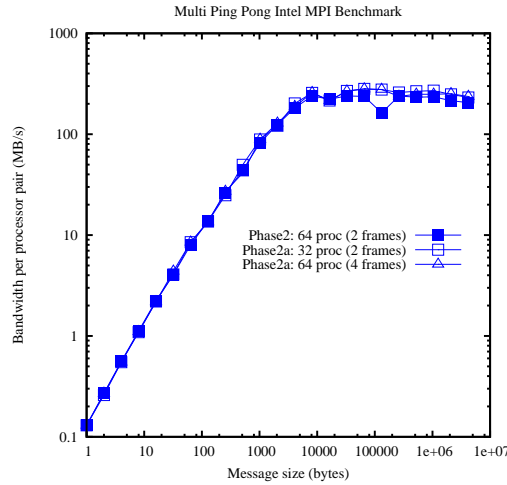


Figure 2: The dependence of bandwidth on message size for the multi Ping Pong benchmark. Phase 2 results are represented by closed shapes and Phase 2a results by open shapes.

The PingPong benchmark involves direct communication between 2 processes. Runs were carried out with the processes within the same node, then repeated with the processes on separate nodes. Figure 1 on the left shows the resulting bandwidths against message size (i.e. high results indicate good performance). It is seen that on separate nodes, i.e. communication is via the switch, there is little difference between the platforms. As expected by the improved memory architecture, the bandwidth is seen to be higher on Phase 2a for large messages. However, it is also observed that for small messages within a node, Phase 2 is performing better. This is highlighted on the right of Figure 1, which zooms in on the small message region and plots time against message

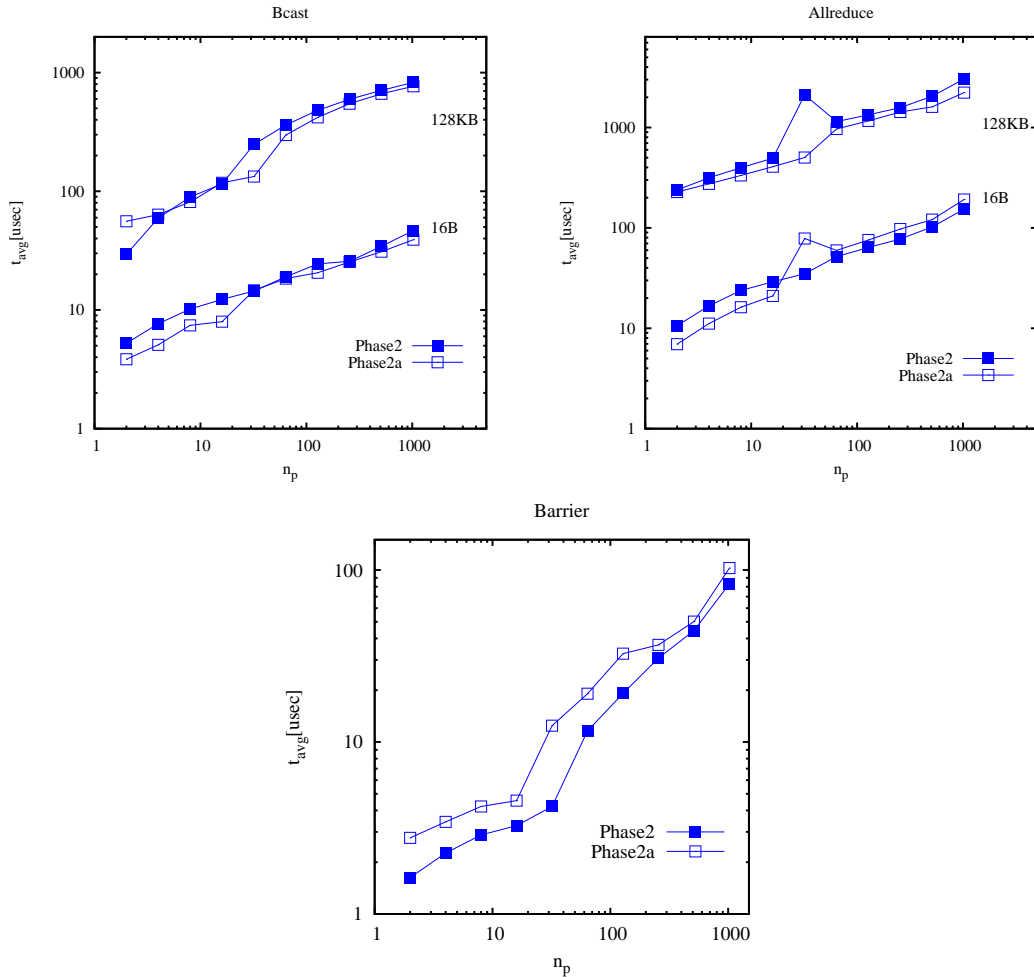


Figure 3: The dependence of the time on the number of processors for the Bcast (top left), Allreduce (top right) and Barrier (bottom) benchmarks. Phase 2 and 2a results are represented by closed and open squares respectively.

size (i.e. low results indicate good performance).

The above PingPong communications do not saturate the inter-node switch and a more rigorous and realistic test is provided by the Multi PingPong benchmark which utilises all available processors. The code has been modified to ensure that all communications utilise the switch. On Phase 2, communications between 64 processors (32 groups of 2) on 2 nodes were benchmarked. On Phase 2a, there are only 16 processors per node, so results were obtained for both 2 nodes (32 processors - 16 groups of 2) and 4 nodes (64 processors - 32 groups of 2). No noticeable difference between the platforms was observed (Figure 2), indicating that the switch is performing equally well after the upgrade. Note that Phase 2 the result at 128KB is noticeably low due to it falling in the region between the eager limit and the bulk transfer minimum message size [4].

The collective communications were compared with use of the MPI Bcast, Allreduce, and Barrier benchmarks. Resulting plots of the benchmark time against the number of processors (i.e. low results indicate good performance) are given in Figure 3. The

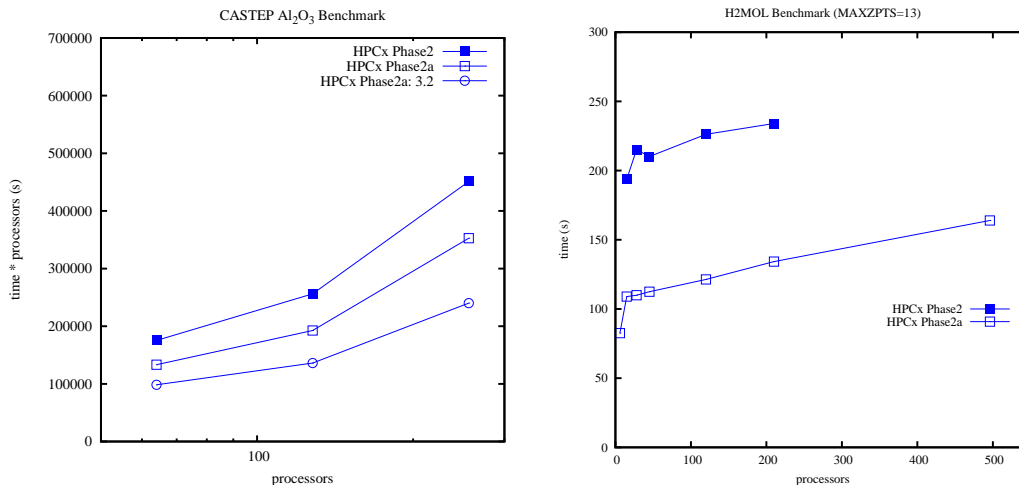


Figure 4: The dependence the total processor time on the number of processors for the CASTEP Al_2O_3 benchmark (left) and the time on the number of processors for the H2MOL benchmark (right). Closed and open shapes denote Phase 2 and Phase 2a results respectively.

discontinuities in the curves are explained by the sizes of the nodes, remembering that the Phase 2 and Phase 2a nodes have 32 and 16 processors respectively.

Bcast and Allreduce results are given for 2 different message sizes: 16B and 128KB. It is seen that for these that there is not much difference between the platforms. For both Bcast and Allreduce, within a node at 16B Phase 2a performs slightly better, perhaps due to less contention within the smaller Phase2a node. It is seen that on Phase 2a for 32 processors, i.e. 2 nodes, the performance is worse than expected from the results for other processor counts. For the Barrier benchmark Phase 2 has a slight advantage at all numbers of processors.

4 Application Benchmarks

This section presents benchmarking results for a number of popular applications, comparing Phase 2 to Phase 2a. Plots will usually show the time multiplied by the number of processors (i.e. the cost of the job) against the number of processors: ideal scaling would be seen as a straight horizontal line. The applications studied were CASTEP, H2MOL, PCHAN, AIMPRO, MDCASK, LAMMPS, NAMD and DL_POLY.

4.1 CASTEP

The CASTEP software package can be used to perform molecular dynamics simulations and provide an atomic-level description (including information regarding energies, forces, and stresses, and calculations of optimum geometries, structures and spectra) of a wide range of materials and molecules [5]. Here results are given for an Al_2O_3 system: a 270 atom slab cell of aluminium oxide sampled with 2 k-points.

Benchmark results from Phase 2 are compared with those from Phase 2a in the left of Figure 4. It is seen from the squares that Phase 2a has a clear advantage over Phase

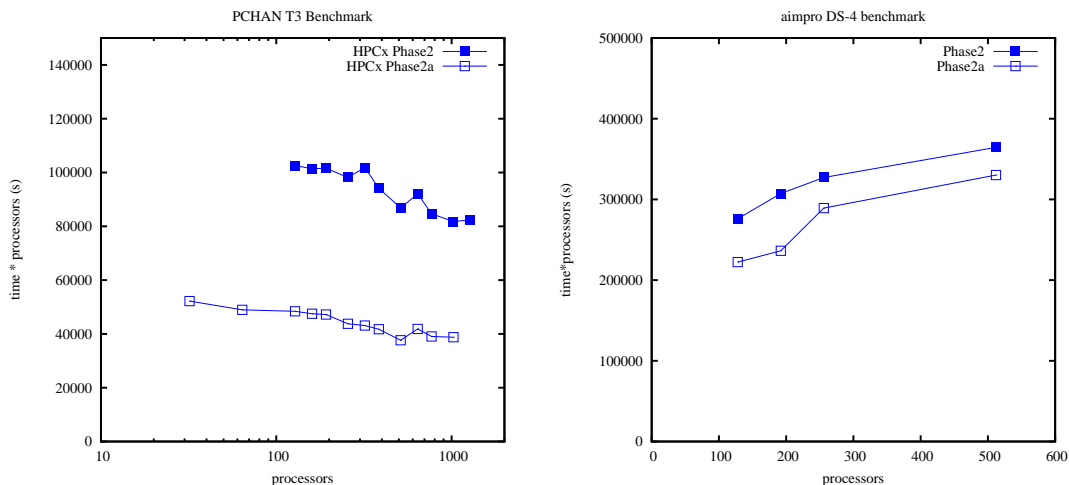


Figure 5: The dependence the total processor time on the number of processors for the PCHAN T3 (left) and AIMPRO DS-4 (right) benchmarks. Closed and open shapes denote Phase 2 and Phase 2a results respectively.

2 with the improvement roughly a factor of 1.3. Also shown are Phase 2a results from using a new version of CASTEP (3.2.1 as opposed to 3.1) which has been optimised for HPCx, and this version is seen to further improve the performance and scaling (where the flatter the curve, the better the scaling) such that the performance improvement at 256 processors is up to a factor of 1.9 on the Phase 2 results.

It therefore seems that CASTEP is significantly benefiting from the improved memory bandwidth of Phase 2a.

4.2 H2MOL

H2MOL calculates the redistribution of energy between electrons and nuclei when hydrogen molecules are heated by short intense laser pulses [7]. Such calculations can be compared with experiment to help to develop the understanding of such molecular behaviour in general.

Note that, unlike the other applications studied, the number of grid points is directly proportional to the number of processors therefore ideal scaling would result in a constant simulation time for different numbers of processors. Hence the right of Figure 4 plots the time against the number of processors. It is seen that H2MOL is showing a almost factor of 2 improvement on Phase 2a, and is therefore thought to be greatly benefiting from the improved memory structure.

4.3 PCHAN

The PCHAN code is designed to simulate the flowing of fluids to study turbulence [8]. This is achieved by solving the governing equations of motion from first principles.

The negative gradients of the curves in the left of Figure 5 demonstrate that PCHAN has better than ideal scaling. This can be explained by the fact that the problem size per processor is becoming more suited to the cache structure as the number of processors increases.

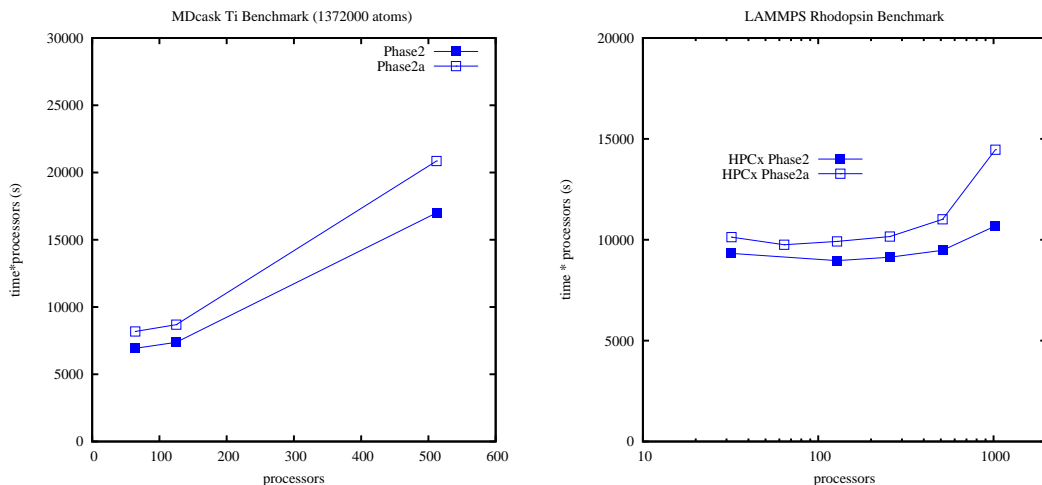


Figure 6: The dependence the total processor time on the number of processors for the MDCASK Ti (left) and LAMMPS Rhodopsin (right) benchmarks. Closed and open shapes denote Phase 2 and Phase 2a results respectively.

It is seen that PCHAN is showing a factor of 2 improvement on Phase 2a, and is therefore again thought to be greatly benefiting from the improved memory bandwidth of the new system.

4.4 AIMPRO

AIMPRO (Ab Initio Modelling PROgram) determines the structure of atoms using the Born and Oppenheimer approximation: the nuclei are treated classically and the electrons quantum mechanically [12]. Here the DS-4 benchmark, which incorporates 433 atoms, 12124 basis functions and 4 k-points, is used.

From the right of Figure 5 it is seen that Phase 2a is outperforming Phase 2 by around a factor of 1.2: AIMPRO is also benefiting from the improved memory bandwidth of the new architecture.

4.5 MDCASK

MDCASK is a molecular dynamics code which was originally developed to study radiation damage in metals [9]. It operates by calculating the energies of and forces on, and determining the motions of, the atoms in the system which is characterised by specific interatomic potentials and boundary conditions.

Unlike the previous results, from the left of Figure 6 it is seen that MDCASK is performing worse on Phase 2a, with the degradation factor higher than the ratio of processor clock frequencies. By the increased performance degradation at 512 processors, the code is seen to scale worse on Phase 2a than Phase 2, but this can probably be explained by the smaller node size (and hence larger number of switch hops) on Phase 2a.

Such classical molecular dynamics codes as MDCASK are characterised by many strided memory accesses. These results could be explained by sensitivity to an increased

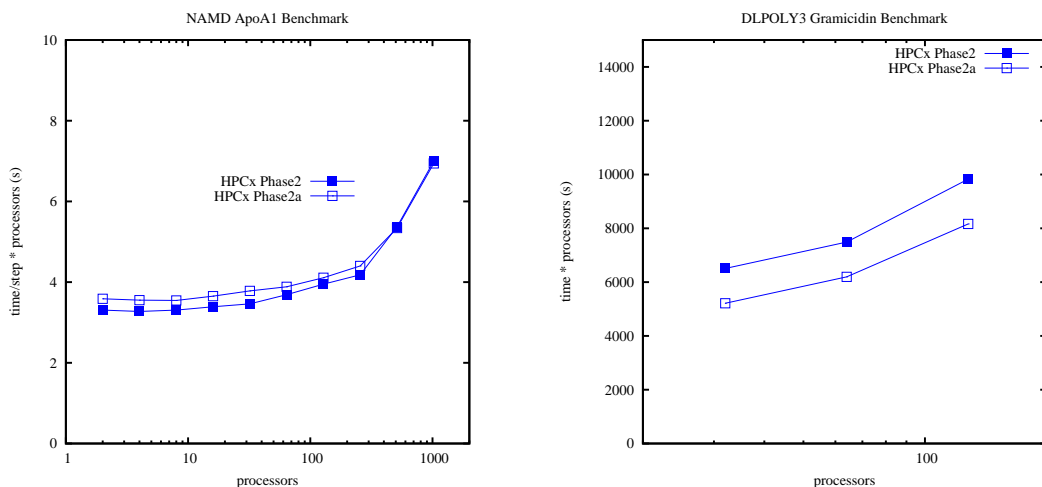


Figure 7: The dependence the total processor time on the number of processors for the NAMD ApoA1 (left) and DL_POLY Gramicidin (right) benchmarks. Closed and open shapes denote Phase 2 and Phase 2a results respectively.

latency in some part of the memory subsystem on Phase 2a. This is currently being investigated [10].

4.6 LAMMPS

LAMMPS is a molecular dynamics package which solves classical physics equations and is able to simulate a wide range of materials including atomic, molecular, metallic and hybrid systems [11]. Here a rhodopsin system, with 2048000 atoms, is benchmarked using LAMMPS 2001.

From the right of Figure 6 it is seen that Phase 2 is outperforming Phase 2a by a factor of the clock frequency ratio at the lower processor counts, and the scaling on Phase 2a is poorer leading to a higher performance difference at 1024 processors. It is thought that similarly to MDCASK the performance degradation of this classical molecular dynamics code may be due to a sensitivity to memory latency.

4.7 NAMD

The NAMD molecular dynamics code is designed to simulate biomolecular systems such as proteins [13].

Results are given for the ApoA1 benchmark, which involves 92000 atoms. It is seen from the left of Figure 7, that for this classical molecular dynamics code, that Phase 2a is performing slightly worse than Phase 2. The expected reasons for this are those already discussed for MDCASK and LAMMPS.

4.8 DL_POLY

DL_POLY is a classical molecular dynamics code which can be used to simulate systems with very large numbers of atoms [6]. Here, DL_POLY3, which is parallelised by domain decomposition and is suitable for large numbers of processors (as opposed to DL_POLY2

which uses a Replicated Data strategy and is suitable for of order 100 processors), was run a system of gramicidin molecules in water with a total of 792,960 atoms.

It is seen from the right of Figure 7 that DL_POLY is performing significantly better on Phase 2a. Compared to the other classical molecular dynamics codes studied, DL_POLY appears to not be so sensitive to the apparent memory latency issue on Phase 2a, and is benefiting from the improved memory bandwidth.

5 Conclusions

The performance of HPCx Phase 2a has been compared to that of Phase 2 via the results of communications and applications benchmarks. Little difference is observed between the two platforms for the Bcast and Allreduce benchmarks, although the Barrier benchmark is performing slightly worse after the upgrade. Most classes of applications studied are benefiting from the improved memory bandwidth on Phase 2a to perform significantly better, compared to Phase 2, that what is naively expected from the clock frequency difference. In some cases up to a factor of 2 improvement over Phase 2 is observed, even with the slower clocks! Classical molecular dynamics applications are, however, performing worse than expected, and this may be due to a sensitivity to a latency increase in some part of the memory substructure on Phase 2a.

5.1 Acknowledgements

We wish to thank Mike Ashworth, Martin Plummer and Andy Sunderland for providing results for this paper.

References

- [1] J.D. McCalpin, A Survey of Memory Bandwidth and Machine Balance in Current High Performance Computers, IEEE Technical Committee on Computer Architecture (TCCA) Newsletter, 1995.
- [2] Joachim Hein, private communication.
- [3] Code and documentation available from http://www.intel.com/software/products/cluster/downloads/mpi_benchmarks.htm
- [4] Alan Gray, Joachim Hein and Stephen Booth, Improved MPI with RDMA, *HPCx Technical Report 0505*, http://www.hpcx.ac.uk/research/hpc/technical_reports/HPCxTR0505.pdf
- [5] M D Segall *et al* 2002, First-principles simulation: ideas, illustrations and the CASTEP code. *J. Phys.: Condens. Matter* **14** 2717-2744; <http://www.tcm.phy.cam.ac.uk/castep/>
- [6] Smith, W. & Forester, T. 1996, DL_POLY: A General Purpose Parallel Molecular Dynamics Simulation Package. *J. Molec. Graphics* **14**, 136; Smith, W., Yong, C. & Rodger, M. 2002, DL_POLY: Applications to Molecular Simulation. *Molecular Simulation* **28**, 385; Todorov, I. & Smith, W. 2004, DL_POLY_3: The CCP5 National UK

- Code for Molecular Dynamics Simulations. *Phil. Trans. R. Soc. Lond. A* **362**, 1835;
http://www.cse.clrc.ac.uk/msi/software/DL_POLY/
- [7] D. Dundas, K.J. Meharg, J.F. McCann and K.T. Taylor, 2003, Dissociative ionization of molecules in intense laser fields *Eur. Phys. J. D* **26**, 51-57;
<http://www.hpcx.ac.uk/research/atomic/h2mol.html>
- [8] PCHAN Webpage,
<http://www.cse.clrc.ac.uk/arc/pchan.shtml>
- [9] MDCASK README
<http://www.llnl.gov/asci/platforms/purple/rfp/benchmarks/limited/mdcask/mdcask.readme.html>
- [10] Ian Bush, private communication.
- [11] Plimpton, S. 1995, Fast Parallel Algorithms for Short-Range Molecular Dynamics. *J. Comp. Phys.* **117**, 1–19;
<http://www.cs.sandia.gov/~sjplimp/lammps.html>
- [12] AIMPRO Website,
<http://aimpro.ncl.ac.uk/>
- [13] Phillips, J., Zheng, G., Kumar S. & Kalé L. 2002, NAMD Biomolecular Simulation on Thousands of Processors. In *Proceedings of the SC2002 Conference*. IEEE Press;
<http://www.ks.uiuc.edu/Research/namd/>