



CONTRACT NUMBER RI-222919

DEISA 2
**DISTRIBUTED EUROPEAN INFRASTRUCTURE FOR
SUPERCOMPUTING APPLICATIONS**

European Community Seventh Framework Programme
RESEARCH INFRASTRUCTURES
Integrated Infrastructure Initiative

Initial Report on the DEISA Benchmark Suite

Deliverable ID: DEISA2-D7-2.1
Due date: October 31st, 2008
Author: Michèle Weiland, UEdin-EPCC

Project start date: May 1st, 2008
Duration: 3 years

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2011)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Table of Contents

	Table of Contents	2
	List of Figures	3
	List of Tables.....	3
1	Introduction	4
1.1	Executive Summary	4
1.2	References and Applicable Documents.....	4
1.3	Document Amendment Procedure	4
1.4	List of Acronyms and Abbreviations	4
2	Background and Progress	6
2.1	Content of the Benchmark Suite	6
2.1.1	DL_POLY	6
2.1.2	ECHAM5 and IFS	7
2.1.3	Revised Content of the Benchmark Suite	7
2.2	New Platforms.....	8
2.2.1	DEISA resources	8
2.2.2	IBM Power6	9
2.2.3	Cray XT4 and X2	9
3	Porting to New Platforms	10
3.1	vip IBM Power6	10
3.1.1	Architecture	10
3.1.2	Porting to vip.....	10
3.1.3	Performance Results.....	11
3.2	HECToR Cray XT4.....	11
3.2.1	Architecture	11
3.2.2	Porting to HECToR XT4.....	12
3.2.3	Performance Results.....	12
3.3	HECToR Cray X2	13
3.3.1	Architecture	13
3.3.2	Porting to HECToR X2	13
3.3.3	Performance Results.....	13
3.4	Adapting JuBE	14
3.5	Analysis of Results.....	14
3.5.1	Astrophysics	14
3.5.2	CFD & Combustion	16
3.5.3	Earth Sciences & Climate Research.....	16
3.5.4	Life Sciences & Informatics.....	18
3.5.5	Materials Science	19
3.5.6	Plasma Physics	22
3.5.7	Quantum Chromodynamics.....	22
3.5.8	Results on the Cray X2.....	23
4	Usage of the DEISA Benchmark Suite	24
4.1	Maintenance of the Website.....	24
4.2	Access and Download Statistics.....	24
4.3	External Usage of the Benchmark Suite.....	25
5	Future Work	27
6	Conclusions	28

List of Figures

Figure 1 – GADGET performance graph.....	15
Figure 2 – RAMSES performance graph	16
Figure 3 – Fenfloss performance graph.....	17
Figure 4 – NEMO performance graph	17
Figure 5 – NAMD performance graph	18
Figure 6 – IQCS performance graph	19
Figure 7 – CPMD performance graph.....	20
Figure 8 – QuantumESPRESSO performance graph.....	20
Figure 9 – GENE performance graph	21
Figure 10 – PEPC performance graph.....	21
Figure 11 – SU3_AHIGGS performance graph.....	22
Figure 12 – BQCD performance graph	23
Figure 13 – Download statistics for the DEISA Benchmark Suite Packages	25

List of Tables

Table 1 – Summary of DEISA resources, ordered by vendors and architectures, until the end of 2009	8
Table 2 – vip IBM Power6 architecture	10
Table 3 – Performance results on <i>vip</i> . For each application, the shown results.....	11
Table 4 – HECToR Cray XT4 architecture.....	11
Table 5 – Performance results on HECToR XT4, dual core mode. As previously,	12
Table 6 – HECToR Cray X2 architecture	13
Table 7 – Performance results on HECToR X2.....	13

1 Introduction

1.1 Executive Summary

- The Benchmarking activity of the DEISA2 project, Task 2 of WP7, is responsible for maintaining the DEISA Benchmark Suite. Maintenance here is defined as: porting the Benchmark Suite to new DEISA platforms; revising and upgrading the content of the Benchmark Suite; maintaining the Benchmark Suite for regular usage.
- This document describes the work undertaken from PM1 to PM5 of the DEISA2 project.
- In Section 2, the document first explains the reasons behind the revision of the Benchmark Suite content. It then justifies the choice of platforms that the Benchmark Suite was ported to.
- In Section 3, the porting work to new platforms is described in detail. Porting problems are highlighted, and a summary of performance results for each platform is given. The performance results for each application in the Benchmark Suite are then analysed separately.
- Section 4 summarises the work put into the maintenance of the Benchmark Suite website, as well as giving an overview of any external (i.e. non-DEISA) usage of the Suite.
- Sections 5 and 6 round off the document by outlining future work and drawing conclusions based on completed work.

1.2 References and Applicable Documents

- [1] DEISA2 Description of Work (Annex I of the Grant Agreement)
- [2] DEISA web pages: <http://www.deisa.eu>
- [3] Deliverable DEISA2-D1.1: Initial Report on Management
- [4] DEISA Benchmark Suite web pages: <http://www.deisa.eu/science/benchmarking>
- [5] JuBE: <http://www.fz-juelich.de/jsc/jube/>
- [6] RZG's IBM Power6 *vip*: <http://www.rzg.mpg.de/computing/hardware/Power6>

1.3 Document Amendment Procedure

This document is prepared according to the guidelines defined by the management of DEISA2. These rules can be found in section 2.7 of the deliverable DEISA2-D1.1 [3].

1.4 List of Acronyms and Abbreviations

CPMD	Car-Parinello Molecular Dynamics application
DEISA	Distributed European Infrastructure for Supercomputing Applications
DL_POLY	Multi-purpose molecular dynamics application, developed by STFC Daresbury Laboratory, United Kingdom
DoW	Description of Work (Annex I of the Grant Agreement)
ECHAM5	Atmosphere circulation modelling code, developed at the Max-Planck Institute for Meteorology, Germany
HECToR	High-End Computing Terascale Resource, the UK's national HPC service

HPC	High Performance Computing
JuBE	Jülich Benchmarking Environment
IFS	Integrated Forecast System, developed at ECMWF
NFS	Network File System
PRACE	Partnership for Advanced Computing in Europe
STFC	Science and Technology Facilities Council, United Kingdom
WP	Work Package

2 Background and Progress

The DEISA Benchmark Suite was developed as part of the eDEISA¹ project with the goal of establishing a European benchmark suite of HPC applications, representative of the most prevalent scientific areas.

The DEISA Benchmark Suite was delivered as a fully tested and packaged suite of applications, contained inside a high-level framework (JuBE – Jülich Benchmarking Environment [5]) that can automate the compilation, the execution and the verification of the results of the applications. The DEISA Benchmark Suite software was complemented by a website [4], which contains information on the applications and the JuBE framework, as well as making the Benchmark Suite available to users via web download.

In DEISA2, the work on the Benchmark Suite is continued, focusing on the maintenance of the software and website. The maintenance task in this context is defined through three subtasks:

- regular usage of the Benchmark Suite
- porting the Benchmark Suite to new DEISA platforms
- revising and upgrading the content of the Benchmark Suite

The work in the first months of the project concentrated mainly on the second and third point, starting off with the revision of the Benchmark Suite content.

2.1 Content of the Benchmark Suite

The Benchmark Suite that was delivered by eDEISA consisted of 14 different applications in seven scientific categories, as well as low-level benchmarks (namely the HPCC benchmark suite). The handover to DEISA2 was a clear point at which a first revision of the contents of the Suite could take place. It was decided that, based on the work in eDEISA, the content of the Benchmark Suite should remain largely unchanged, as the applications cover the most important scientific areas and computational algorithms used in HPC. These findings are reinforced by research undertaken in WP6 (“Software enabling for petaflop/s systems”) of PRACE – the applications in the DEISA Benchmark Suite are shown to have considerable usage across the main HPC platforms in Europe. The results of this research can be found in the PRACE deliverable D6.1.

Two applications, DL_POLY and ECHAM5, had to be dropped from the Benchmark Suite nonetheless. The reasons and justifications for removing these applications from the Suite are outlined in the subsequent paragraphs.

2.1.1 DL_POLY

DL_POLY is a widely used multi-purpose molecular dynamics code, developed by STFC Daresbury Laboratory in the UK. DL_POLY is a licensed code which is available free of charge to academics. Because of the licensing restrictions, DEISA is not allowed to distribute the source code for the application. The DEISA Benchmark Suite website therefore stated that only the framework for the application would be available for download, redirecting users to the DL_POLY website to obtain both license and source code. This system is applied with other applications in the Benchmark Suite (e.g. CPMD). However, as the Benchmark Suite framework is adapted to a specific version of an application, it needs to be guaranteed that

¹ The eDEISA project finished at the end of June 2008.

this version is available to users even after the release of a new version of an application. Porting and integrating a new release version of an application to the Benchmark Suite can potentially involve substantial effort. The users of the Benchmark Suite should experience a fully tested and integrated software package and the compatibility of application and framework need to be guaranteed.

In addition, the performance results that are published on the DEISA Benchmark Suite website become meaningless if the versions of the applications do not remain consistent.

Unfortunately, once a new version of the DL_POLY source code was released, the previous version of the source will cease to be distributed and is no longer available to users. Given that the availability of the integrated and fully compatible version of DL_POLY can not be guaranteed, it was decided to remove the application from the Benchmark Suite.

2.1.2 ECHAM5 and IFS

ECHAM5 is a climate modelling application, developed at the Max-Planck Institute for Meteorology in Germany. Although ECHAM5 is a very important European application, it was decided to stop actively supporting this code as part of the DEISA Benchmark Suite. ECHAM5 has proven to be an extremely difficult application to port to different systems and architectures, and the amount of manpower available in the benchmarking task of DEISA2 is insufficient to justify continuing spending effort on such a technically challenging application.

IFS, the Integrated Forecast System, developed at ECMWF, was chosen to replace ECHAM5 in the Benchmark Suite. As this application is new to the Benchmark Suite, it needs to be integrated to the JuBE benchmarking framework. This work is undertaken by ECMWF under the eDEISA project.

2.1.3 Revised Content of the Benchmark Suite

The revised content of the Benchmark Suite, ordered by scientific areas, is:

- *Astrophysics:* GADGET, RAMSES
- *CFD and Combustion:* Fenfloss
- *Earth Sciences and Climate Research:* IFS, NEMO, (ECHAM5)
- *Life Sciences and Informatics:* IQCS, NAMD
- *Materials Science:* CPMD, QuantumESPRESSO
- *Plasma Physics:* GENE, PEPC
- *Quantum Chromodynamics:* BQCD, SU3_AHIGGS

and

- *Low-level benchmarks:* HPCC

2.2 New Platforms

As specified in the Description of Work, the Benchmark Suite needs to be ported to new DEISA platforms. The decision as to what systems would be addressed during the first months of the project was based on the current HPC infrastructure of DEISA, as outlined in the following section.

2.2.1 DEISA resources

Table 1 shows a summary of the DEISA infrastructure up to the end of 2009. From this summary it becomes clear that three types of systems cover the majority of the DEISA landscape: Cray MPP systems, IBM SMP clusters and IBM Blue Gene systems. Under eDEISA, the Benchmark Suite was already ported to LRZ's SGI Altix, CINECA's IBM BCX cluster and partly to the NEC SX-8 at HLRS (as well as to FZJ's IBM Power4+ system *JUMP* and CSC's dual-core Cray XT4 *Louhi*, both of which are no longer in operation under the original specification and are being upgraded).

Vendors	Systems	DEISA partners	Comment
Cray	XT4 MPP	EPCC	will be upgraded in October 2009
	XT4 + XT5 MPP	CSC	in process of being upgraded at time of writing
	X2 Vector unit	EPCC	
IBM	Power5 SMP cluster	CINECA	will be decommissioned in March 2009 ²
		EPCC	
	Power6 SMP cluster	ECMWF	
		FZJ	
		RZG	
		SARA	
	BG/P	FZJ	
		IDRIS	
		RZG	will be upgraded in size (12k to 16k cores)
	BCX cluster	CINECA	
PowerPC cluster	BSC		
NEC	SX-8	HLRS	
SGI	Altix	LRZ	

Table 1 – Summary of DEISA resources, ordered by vendors and architectures, until the end of 2009

It was decided that the initial porting effort should concentrate on one of the IBM Power6 SMP clusters as well as a Cray MPP system, as these two architectures form a significant part of the overall DEISA resources.

² The replacement for this system has not been specified yet.

2.2.2 IBM Power6

Four IBM Power6 cluster installations are currently part of the DEISA infrastructure. It was decided that the Benchmark Suite should initially be ported to RZG's system, called *vip*. The ECMWF system was not yet available during the period the work would have been undertaken, and the FZJ system is very small at only 448 cores³. Between the remaining two machines from RZG and SARA, *vip* was chosen because it is a slightly more "standard" system (the operating system on *vip* is IBM's AIX, whereas SARA's machine uses Linux) as well as being the larger one of the two systems.

2.2.3 Cray XT4 and X2

In the context of the DEISA infrastructure, Cray is the second most important vendor of HPC hardware next to IBM. The Benchmark Suite was ported to the CSC's Cray XT4 system *Louhi* under eDEISA, but *Louhi* was a relatively small system that would only allow for performance runs up to 512 cores. This system is being upgraded at the time of writing. It was decided to port the Benchmark Suite to EPCC's Cray XT4, a large system at 11,328 cores⁴.

An X2 vector component is part of the *HECToR* service. It was decided to port a small number of applications (CPMD, Fenfloss, IQCS and SU3_AHIGGS) to this architecture. Two of the applications, CPMD and Fenfloss, have successfully been ported to vector architectures in the past. IQCS and SU3_AHIGGS have never previously been run on a vector system and the porting process would be an opportunity to study the feasibility of such a task.

³ <http://www.fz-juelich.de/jsc/jump>

⁴ <http://www.hector.ac.uk/service/hardware/>

3 Porting to New Platforms

Porting the DEISA Benchmark Suite to a new platform consists of two steps: porting the applications; and adapting the JuBE framework. This section describes the new platforms in detail, provides and analyses performance results and explains system specific modifications to JuBE. The performance results reproduced here are for illustrative purposes only and are not complete. The complete performance results, ordered by applications and including information on the compilers (versions and options) that were used, are available on the website under <http://www.deisa.eu/science/benchmarking/results>. It is also important to note that the results presented here reflect the applications' performance achieved without putting effort into in-depth platform specific optimisation. Any performance results therefore do not necessarily reflect the best performance that can be achieved on a system.

3.1 vip IBM Power6

3.1.1 Architecture

The new IBM SMP cluster *vip* was installed at RZG during spring 2008. Using the 4.7GHz Power6 processors with 2 GB of memory per core and Infiniband interconnect, the system has a theoretical peak performance of 120 Teraflop/s [6]. Table 2 summarises the details of the machine's specification.

vip – IBM eServer575			RZG
Shared memory cluster – 6,400 cores (200 nodes with 32 cores each)			
Processors	Cache	Switch	OS
Power6 4.7 GHz Dual-core Memory: 125 nodes @ 64GB 75 nodes @ 128GB 2 nodes @ 256GB	L1: 64 KB DC + 64 KB IC (per core) L2: 4MB (per core) L3: 32 MB (per chip – 2 cores)	Infiniband	AIX

Table 2 – vip IBM Power6 architecture

3.1.2 Porting to vip

Porting the applications to the IBM Power6 *vip* was mostly straightforward, using the configuration files from the IBM Power4 *JUMP* as templates.

The only difficulties with the system were experienced when running NEMO on large processor counts (2048 cores) with the small dataset (GYRE.25), resulting in a deadlock of the execution. After investigation, it became clear that the load balancing of the application deteriorates severely as the number of processes increases. On 2048 cores, the load balancing worsens to the point where the MPI messages cause a deadlock. The load balancing problem is inherent to the application itself, so it cannot be fixed easily. The deadlock however can be solved by using asynchronous “sends”; this can be changed by setting the parameter `c_mpi_send="I"` in the `namelist.in` input file. NEMO has been ported to a variety of different platforms in the frame of DEISA, but *vip* is thus far the only systems that requires the use of asynchronous communication.

3.1.3 Performance Results

The applications were tested with (where possible) runs between 64 and 2048 cores. Table 3 summarises the performance results for each application, using the largest dataset available in the Benchmark Suite.

IBM Power6 – vip					
<i>Categories</i>	<i>Applications + Datasets</i>	<i>From (#cores)</i>	<i>To (#cores)</i>	<i>Efficiency</i>	<i>Comment</i>
Astrophysics	GADGET benchmark dataset, no IO	64	2048	64→2048	
		169.46s	8.92s	0.593	
	RAMSES Sedov3D 1024	256	2048	256→2048	insufficient memory for runs under 256 cores
CFD + Combustion	Fenfloss Cavity_224	64	256	64→256	does not scale beyond 256 cores
		77.45s	42.63s	0.454	
Earth Sciences + Climate Research	NEMO GYRE.50, no IO	128	512	128→512	insufficient memory for runs under 128 cores – also does not scale beyond 512 cores (time for 2048 core run: 607s)
		1,106s	381s	0.725	
Life Sciences + Informatics	NAMD 4f2hc	64	512	64→512	does not scale beyond 512 cores
		12,428s	3,456s	0.449	
	IQCS 34 Qubits	128	2048	128→2048	insufficient memory for runs under 128cores
Materials Science	CPMD H2O_256mol	64	512	64→512	
		26.04s	4.21s	1.00	
	QuantumESPRESSO AUSURF112 taskgroup	64	256	64→256	no performance numbers for largest dataset yet
Plasma Physics	GENE strong_512	256	2048	256→2048	insufficient memory for runs under 256 cores
		8.37s	1.30s	0.804	
	PEPC Sphere 5M	64	1024	64→1024	
QCD	BQCD 48*48*48*96, 50 iter.	64	2048	64→2048	
		2481.87s	51.55s	1.50	
	SU3_AHIGGS 256^3 lattice, 100 iter.	64	2048	64→2048	
		734.20s	30.50s	0.752	

Table 3 – Performance results on vip. For each application, the shown results are for the largest dataset currently available in the Benchmark Suite

3.2 HECToR Cray XT4

3.2.1 Architecture

The UK's new national HPC service, *HECToR*, was launched in January 2008. The main component of the system is a Cray XT4 MPP machine that uses 2.8 GHz dual core AMD Opteron processors and the Cray Seastar network, with a theoretical peak performance of 60 Teraflop/s. Table 4 summarises the system's specification.

HECToR – Cray XT4			EPCC
Massively parallel supercomputer – 11,328 cores (5,664 nodes with 2 cores each)			
<i>Processors</i>	<i>Cache</i>	<i>Switch</i>	<i>OS</i>
AMD Opteron 2.8 GHz Dual-core 3 GB memory per core	L1: 64 KB DC + 64 KB IC (per core) L2: 1 MB (per core) L3: -	Cray Seastar2	CNL

Table 4 – HECToR Cray XT4 architecture

3.2.2 Porting to HECToR XT4

Porting the applications to the Cray XT4 was mainly straightforward by again using the system specific files created for *Louhi* as templates for *HECToR*. However, there were difficulties with two codes: IQCS and QuantumESPRESSO.

IQCS includes the number of cores it runs on in the source code during the compilation step. The code compiles, but fails during the linking step due to memory problems – this happens for 64 core runs and 33 Qubits, as well as 128 core runs and 34 Qubits. After some investigation of the problem, the source code was modified to allow dynamic allocation, which resolved the linking issue.

QuantumESPRESSO on the other hand compiles, links and executes apparently without problems, however the verification of the results it generates fails. The reason for this is currently not clear and also needs to be investigated in more detail.

3.2.3 Performance Results

Similar to the IBM Power6, the applications were tested in a range of 64 to 2048 cores with all their datasets. In addition, on *HECToR* the performance runs were executed in both dual core and single core mode. In single core mode, only one of the cores on each chip will be used for computation, but the application has access to all the memory on that chip. Applications that are memory intensive will thus see a performance increase in single core mode.

Table 5 summarises the dual core performance results on the *HECToR* XT4 system. Complete results (also for single core mode) can be found on the website.

Cray XT4 dual core – HECToR					
Categories	Applications	From (#cores)	To (#cores)	Efficiency	Comments
Astrophysics	GADGET benchmark dataset, no IO	64	1024	64→1024	dataset does not run on 2048 cores
		105.38s	7.55s	0.872	
	RAMSES Sedov3D 1024	256	2048	256→2048	insufficient memory for runs under 256
CFD + Combustion	Fenfloss Cavity_224	64	1024	64→1024	dataset does not run on 2048 cores
		113.56s	8.75s	0.811	
Earth Sciences + Climate Research	NEMO GYRE.50, no IO	128	1024	128→1024	insufficient memory for runs under 128 cores – also does not scale beyond 1024
		2079s	386s	0.673	
Life Sciences + Informatics	NAMD 1GND	64	512	64→512	no results for largest dataset (4f2hc) yet
		2,449.2	832.65s	0.367	
	IQCS 34 Qubits	256	2048	256→2048	insufficient memory for runs under 256
Materials Science	CPMD H2O_256mol	64	512	64→512	
		52.64s	11.51s	1.00	
	QuantumESPRESSO AUSURF112 taskgroup	64	512	64→512	
Plasma Physics	GENE strong_512	256	2048	256→2048	insufficient memory for runs under 256
		14.69s	1.94s	0.946	
	PEPC Sphere 5M	64	1024	64→1024	
QCD	BQCD 48*48*48*96, 50 iter.	64	2048	64→2048	
		2,635.5	96.86s	0.850	
	SU3_AHIGGS 256^3 lattice, 100 iter.	64	2048	64→2048	
		771.90s	24.40s	0.988	

Table 5 – Performance results on HECToR XT4, dual core mode. As previously, the shown results are for the largest dataset currently available in the Benchmark Suite

3.3 HECToR Cray X2

3.3.1 Architecture

In August 2008, a vector component was added to the *HECToR* service. The X2, also called “Black Widow”, consists of 112 vector processing cores, each capable of 25 Gigaflop/s. Details of the X2’s architecture are outlined in Table 6.

HECToR – Cray X2 “Black Widow”			EPCC
Vector processing system – 112 cores (28 nodes with 4 cores each)			
<i>Processors</i>	<i>Cache</i>	<i>Switch</i>	<i>OS</i>
X2 processors 1.6 GHz per core 7.5 GB memory per core	L1: 16 KB DC + 16 KB IC (per core) L2: 512 KB (per core) L3: 8MB (per node)	Cray YARC	CNL

Table 6 – HECToR Cray X2 architecture

3.3.2 Porting to HECToR X2

It was decided to restrict the porting to the vector component to four applications only (those with potential suitability for vector architecture): CPMD, Fenfloss, IQCS and SU3_AHIGGS. Compiling the four codes on the X2 platforms was unproblematic. IQCS however ran into memory difficulties when executing runs with more than 31 Qubits. It was therefore decided to reduce the dataset sizes and run 28, 29 and 30 Qubits as these datasets fit into memory. The 31 Qubits dataset could only be run on 64 cores on the X2.

Although CPMD should be an unproblematic application on the X2 vector system, at the time of writing the application had not been run on this system and therefore no performance results are presented here. There are two reasons for this: firstly, the FFTW library, which is required by the application, is not yet available on the X2 and thus CPMD cannot yet be run on the system. In addition, a number of intrinsic Fortran functions (such as `system`, `sleep` or `large`) which are used in the code are not currently supported by the X2 compiler. Once the FFTW library becomes available and the intrinsic functions are supported (maybe in a later version of the compiler), the CPMD runs on the X2 will be completed.

3.3.3 Performance Results

Table 7 summarises the performance of the four selected applications on the Cray X2. The timings for all the datasets are included.

Cray X2 – HECToR						
<i>Categories</i>	<i>Applications + Datasets</i>		<i>From (#cores)</i>	<i>To (#cores)</i>	<i>Efficiency</i>	<i>Comments</i>
CFD + Combustion	Fenfloss	Cavity_224	8	64	8→64	
			296.09s	33.55s	1.102	
Life Sciences + Informatics	IQCS	28 Qubits	16	64	16→64	
			11.44s	2.48s	1.153	
		29 Qubits	32	64	32→64	
			11.75s	6.07s	0.967	
		30 Qubits	32	64	32→64	
23.51s	12.17s	0.965				
Materials Science	CPMD		-	64	8→64	not yet available
			n/a	24.19s	n/a	
QCD	SU3_AHIGGS		8	64	8→64	does not vectorise
			-	-	-	

Table 7 – Performance results on HECToR X2

3.4 Adapting JuBE

The JuBE framework is a generic Perl and XML environment that allows for user-friendly and fully integrated compilation, execution and verification of benchmarks. The specific setup of the *HECToR* XT4 system requires one substantial change in the JuBE framework. *HECToR* uses two main filesystems: NFS on the login nodes (i.e. */home*) and Lustre on the compute nodes (i.e. */work*). The Lustre filesystem is the only filesystem that is accessible by the compute nodes, which means that the compute nodes can only read files from and write files to */work*. Executables and any input data files need to be located on */work* for jobs to run successfully, and all output generated from this job will be written to */work*. In addition, unlike */home*, */work* is not a backed up environment. If users want to make sure their data is safely backed up, they need to keep copies on the login nodes.

Before the JuBE framework was adapted to cope with this specific setup of *HECToR*, it assumed that it is possible to compile a source code and submit a job from the */home* directory of any system.

This solution is not feasible for *HECToR*. The behaviour of JuBE can now be altered by using a recently implemented option named `-cmpdir <dir>`, which stands for “compilation directory”. Users can select the directory that is used for compilation. The option `-tmpdir` allows users to choose the execution directory, which in the case of *HECToR* needs to be on the */work* filesystem. Both options can be used in conjunction, thus giving users the possibility to choose separate spaces for compilation and execution. After compilation, all necessary redirections and copy commands into the execution space are done automatically.

The described options can be used in the following way:

```
jube -start <bench-example.xml> -tmpdir <dir> -cmpdir <anotherdir>
```

3.5 Analysis of Results

The following sections of this report present the performance of the benchmark applications, classed by scientific areas, on the new platforms. The performance results on *vip* and *HECToR* (XT4) are presented as well as to the two older platforms, *JUMP* and *Louhi*, which are now no longer operational.

For a handful of applications, there are no definitive performance results on *vip* available yet. All applications have been ported to the machine and run successfully. However, for some applications the initial performance achieved was below what would be expected. As this is a very new platform there has not yet been time to investigate the reasons for this or to experiment with compiler flags and environment settings. We would expect that the performance figures for *vip* that are included here would also improve after this investigation. Full performance results will be presented in a future deliverable.

The final section gives a summary of the performance results achieved with on the Cray X2 vector platform.

3.5.1 Astrophysics

Both GADGET and RAMSES were unproblematic to port to *vip* the new platforms. Figure 1 shows the performance of GADGET across machines (for the Cray XT4 platforms, the results for both dual core and single core mode are plotted). The application scales well on all systems, though unfortunately it was not possible to perform runs up to 2048 cores on *HECToR* with the current dataset. The poorest performance is shown by *JUMP*, which is perhaps unsurprising as this system has the CPUs with the lowest clock rate (1.7GHz). The

single core runs on the Cray XT4 systems marginally outperform the dual core runs, which shows that the application benefits from increased memory and memory bandwidth, but this is not crucial to its performance.

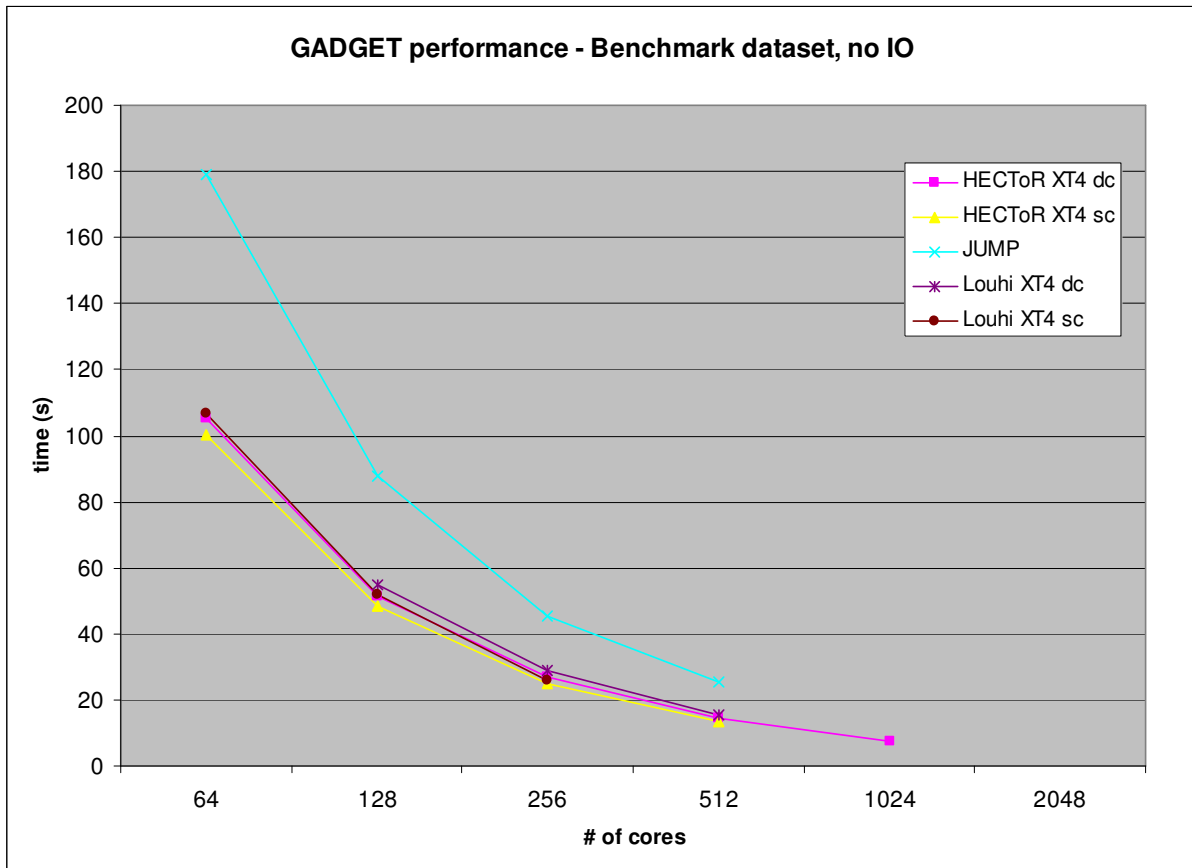


Figure 1 – GADGET performance graph

Figure 2 shows the timings of RAMSES on *vip* and *HECToR*. The results plotted here are for the largest dataset, *Sedov3D 1024* – there are no results for this dataset on either *JUMP* or *Louhi*. Independent of the dataset, *vip* always gives the best performance for this application by about 20%. In addition, the single core XT4 runs outperform the dual core runs considerably. These two observations lead to the conclusion that RAMSES is computationally intensive and benefits from the high clock rate of the Power6 chip as well as from the increased memory bandwidth of single core mode. The details of the interconnect do not seem to influence the execution times.

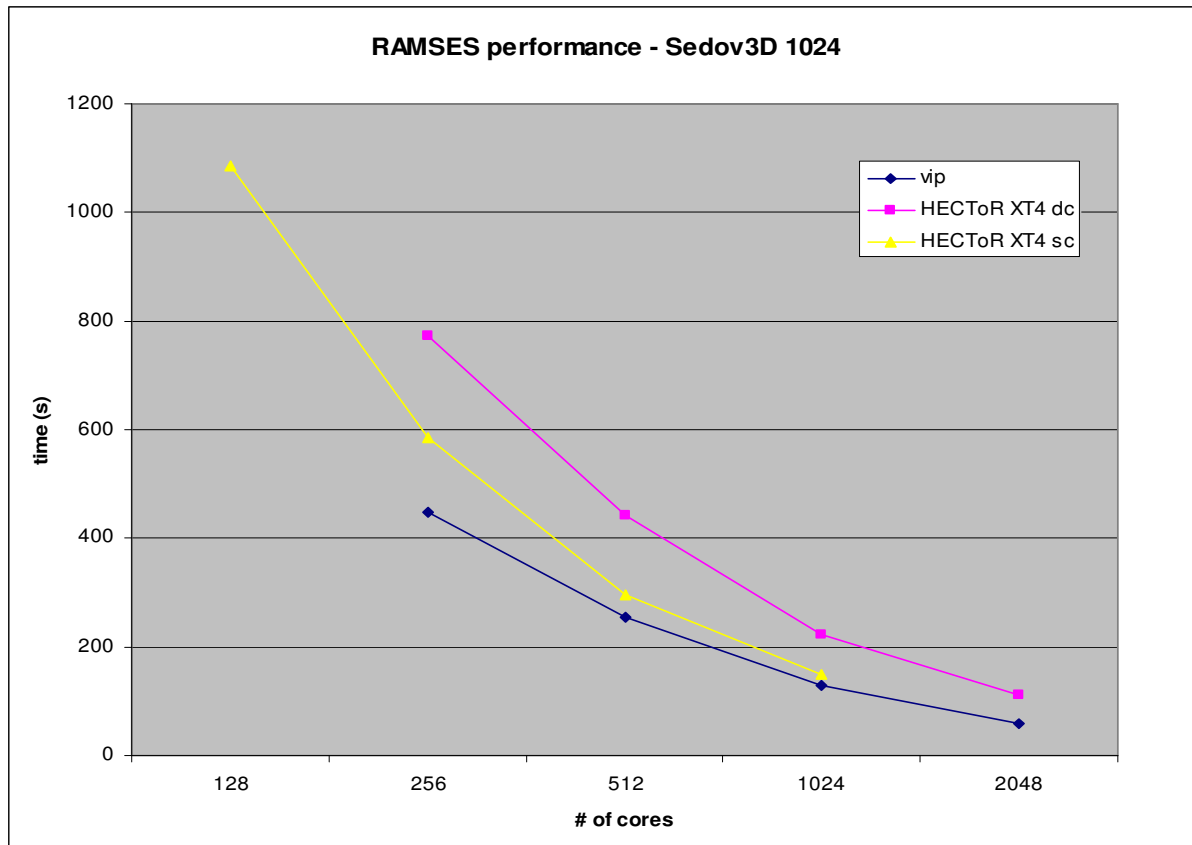


Figure 2 – RAMSES performance graph

3.5.2 CFD & Combustion

Figure 3 summarises the performance of Fenfloss using the *Cavity_224* dataset. The results presented all show good scaling. Both *HECToR* and *Louhi* exhibit good performance, and even the results *JUMP* are competitive. The Cray XT4 systems manage the global communications well and the code scales up to 1024 cores.

3.5.3 Earth Sciences & Climate Research

Figure 4 summarises the timings of NEMO with the *GYRE.50* datasets without disk IO. Up to 256 cores, the performance of NEMO is best on *vip*; however the timings then increase and the application stops scaling. This requires investigation, and the larger *GYRE..150* dataset might also be more appropriate for these large core counts. The difference in performance between single core and dual core is nearly a factor of two, which again suggests that this code is memory bandwidth limited.

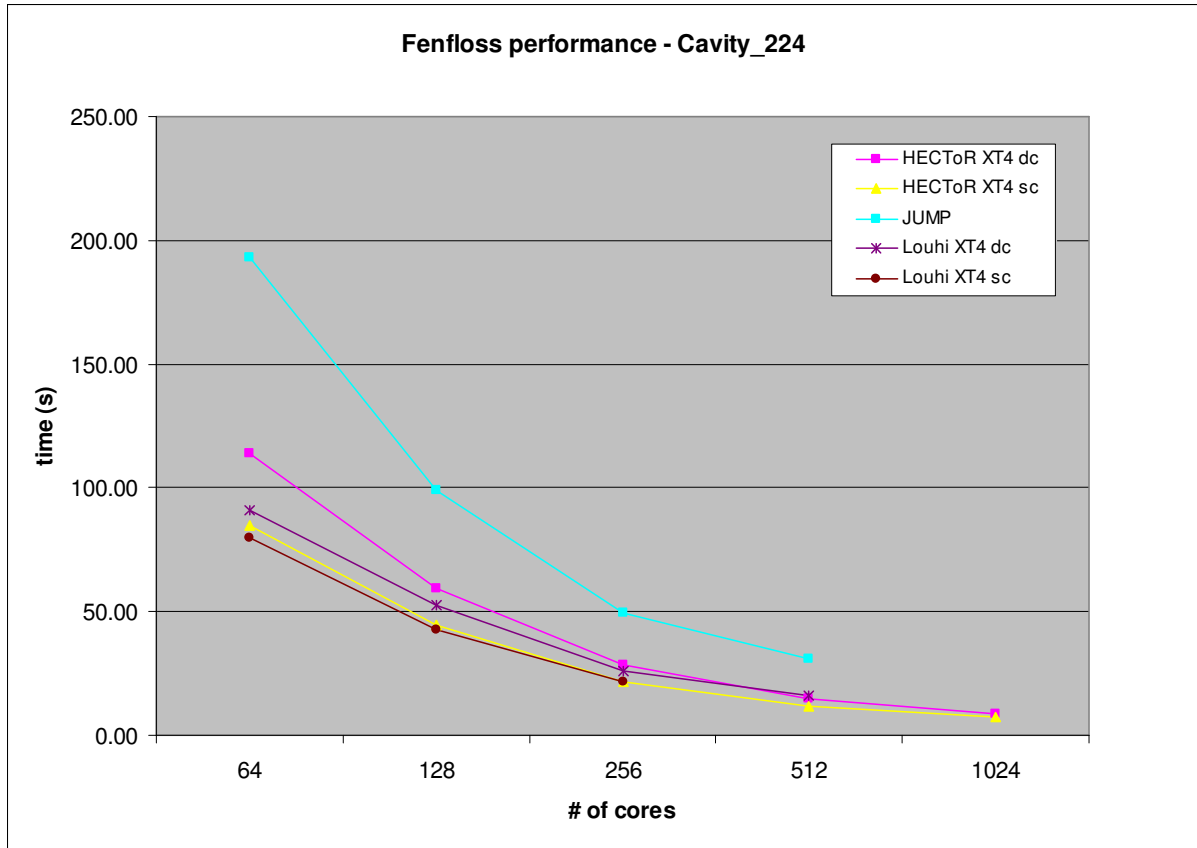


Figure 3 – Fenfloss performance graph

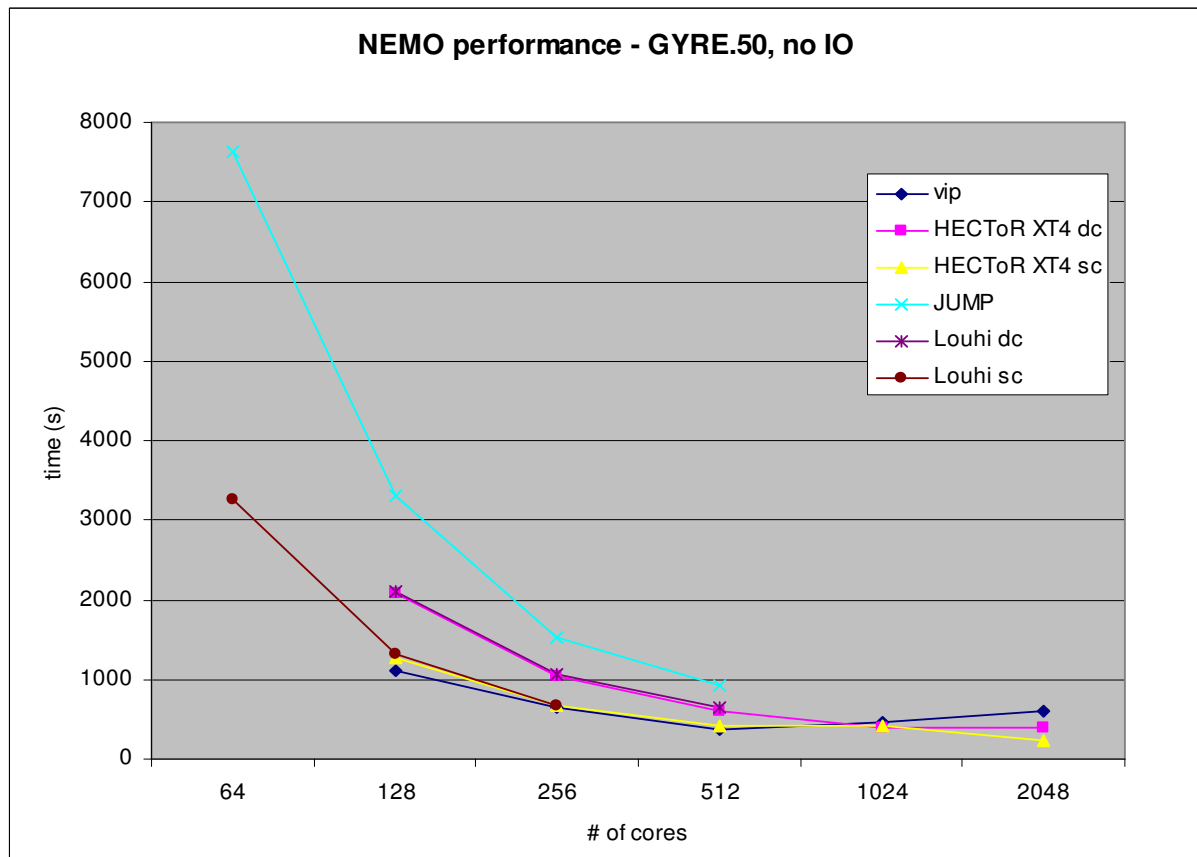


Figure 4 – NEMO performance graph

3.5.4 Life Sciences & Informatics

The performance of NAMD with the *IGND* dataset is shown in Figure 5. The timing differences between single and dual core mode are small, indicating that memory bandwidth is a minimally limiting factor for the application with this dataset. *vip* shows the best performance up to 256 cores and then stops scaling. As for NEMO, this requires further investigation and perhaps the dataset used as also too small for the fast Power6 CPU. On *JUMP*, *Louhi* and *HECToR* the scaling is acceptable for this dataset. The best performance on 512 cores is shown by *HECToR* in single core mode, 9% faster than *HECToR* in dual core mode.

Figure 6 shows the performance of IQCS on *HECToR* and *JUMP*. Due to memory restrictions, the larger datasets cannot run on small numbers of cores. The performance numbers for 34 Qubit runs are incomplete for *Louhi*, therefore the results here are limited to *JUMP* and *HECToR*. IQCS creates datasets “on the fly” and fills up memory entirely, thus it is not surprising that the increased memory bandwidth available in single core mode increases performance by approximately 40%.

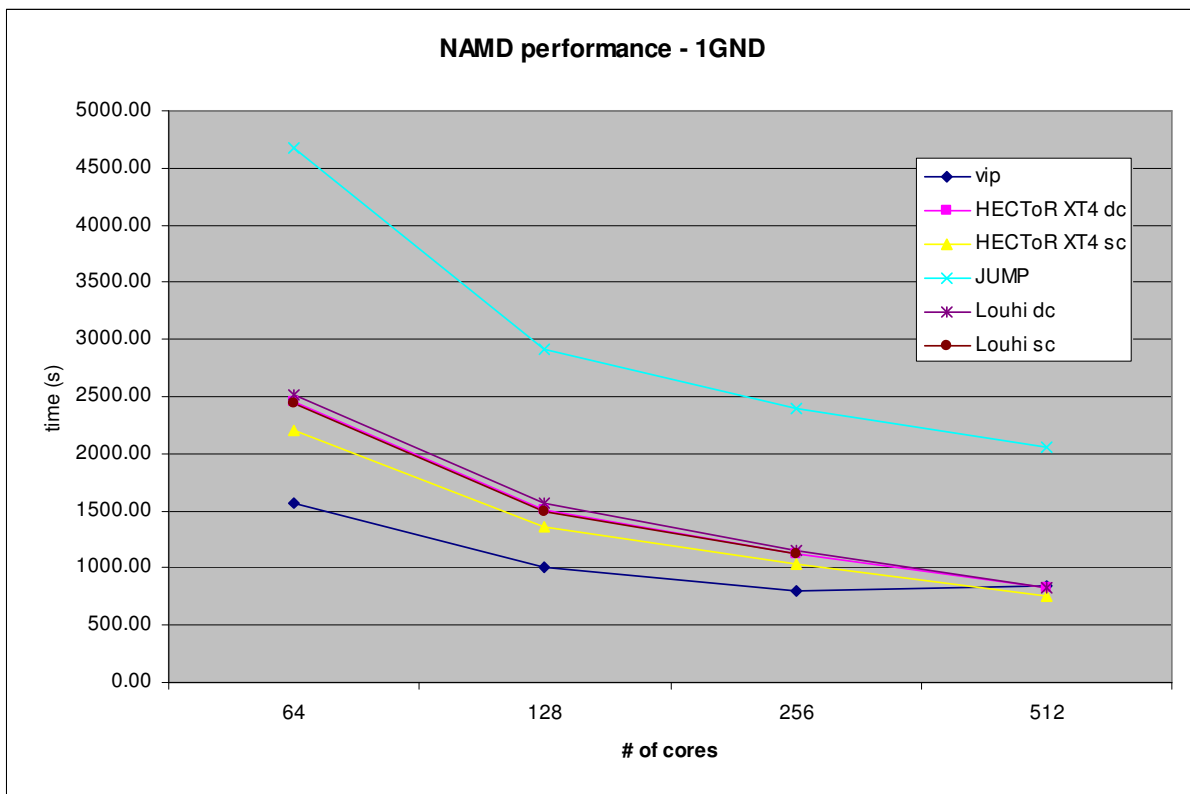


Figure 5 – NAMD performance graph

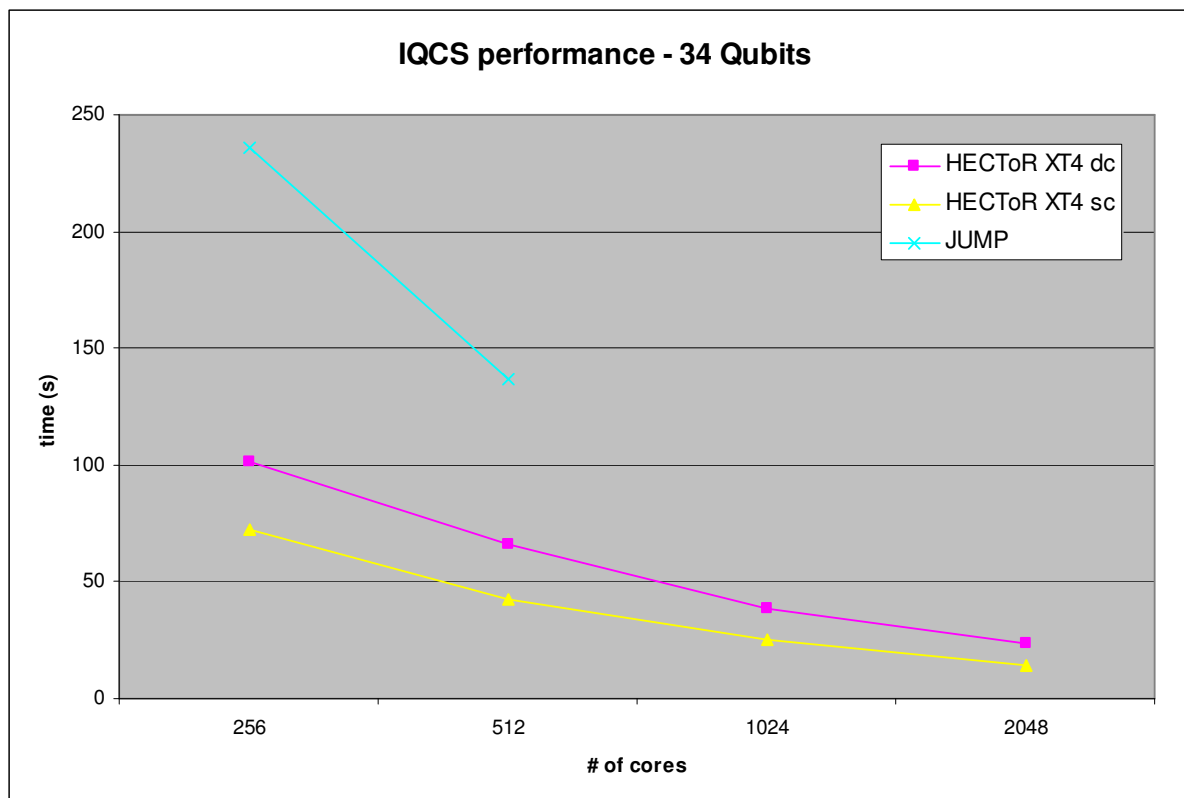


Figure 6 – IQCS performance graph

3.5.5 Materials Science

CPMD performance results up to 512 cores for the *H2O_256mol* dataset are plotted in Figure 7. The differences in speed between the different systems are quite clear here: the application benefits from a high clock rate and thus performs well on *vip*, yet it is also memory bandwidth limited and therefore shows increased performance in single core mode. The scaling on *vip* is similar to that on *HECToR*. Here, the scaling drops on *JUMP* and *Louhi*. In this case, the interconnect does not play a crucial role when it comes to code performance and CPU clock speed combined with memory bandwidth determine the runtime of the application. This would also explain why the two older platforms achieve the poorest results.

Figure 8 shows the performance of QuantumESPRESSO up to 512 cores, which is the point at which the application (using this dataset) stops scaling on all platforms. The results shown here are only for *vip* and *HECToR*, as a different metric was used to measure the performance than on previous systems. On 64 cores, the performance on *vip* is nearly three times better than on *HECToR* dual core. However on *vip* the code stops scaling at 256 cores, which indicates that the dataset is again too small for the fast Power6. The performance on *HECToR* is approximately 10% better in single core than in dual core mode, which indicates that the code benefits slightly from improved memory bandwidth.

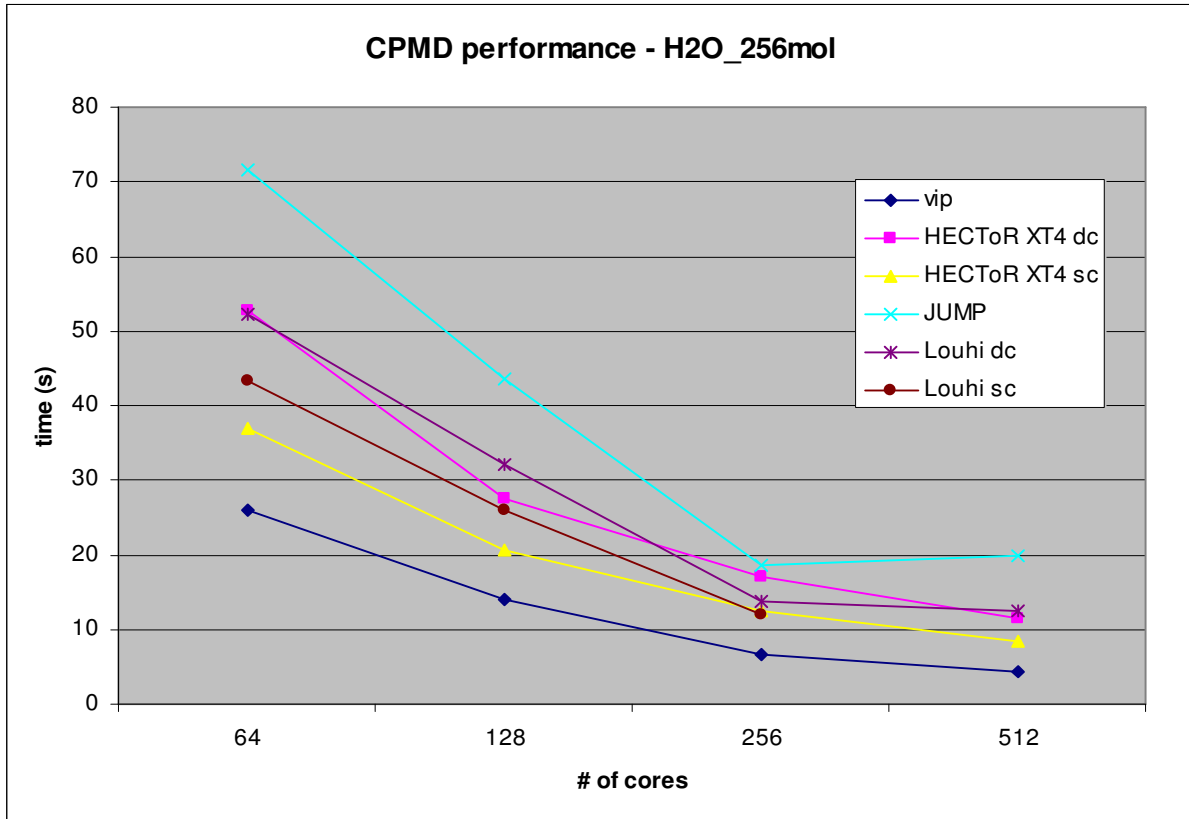


Figure 7 – CPMD performance graph

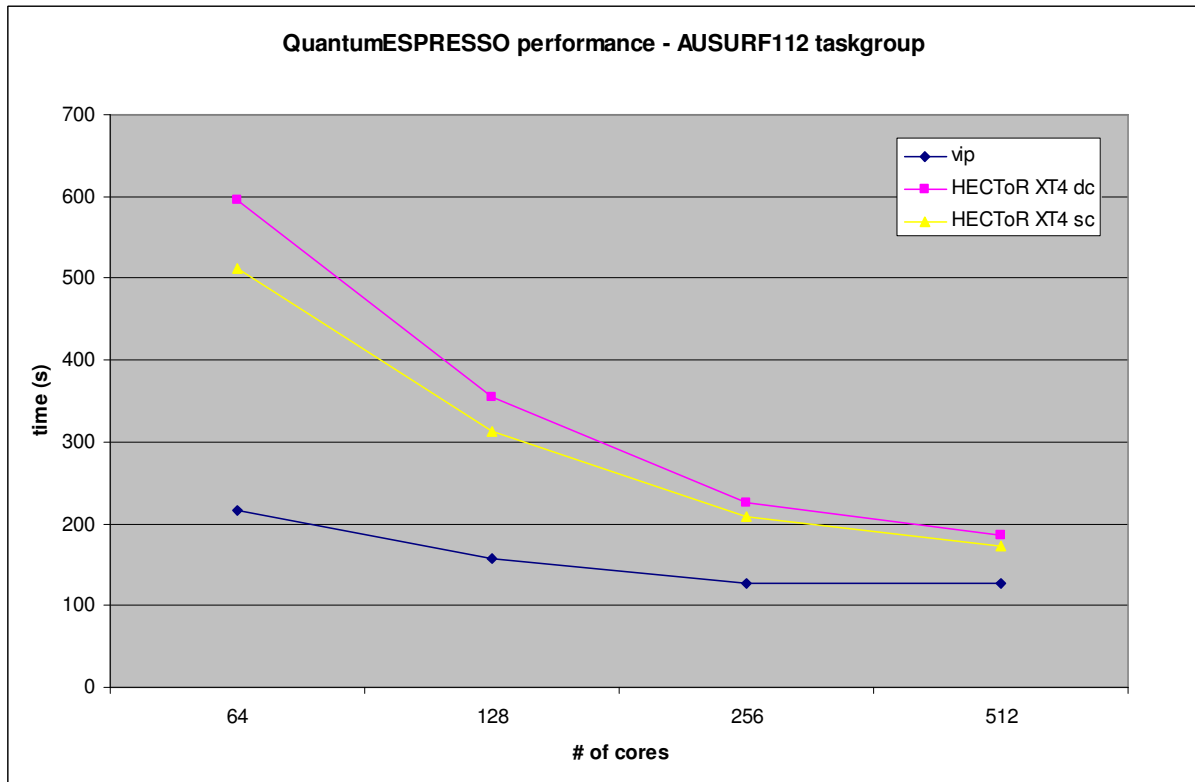


Figure 8 – QuantumESPRESSO performance graph

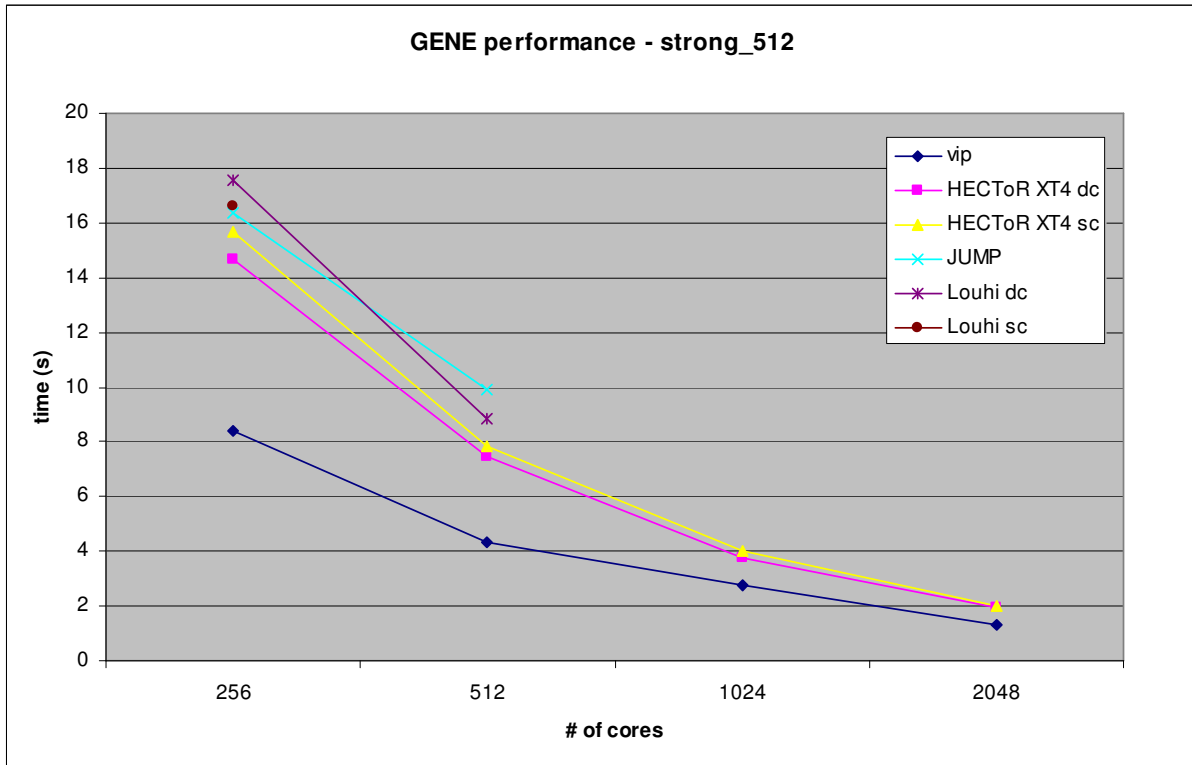


Figure 9 – GENE performance graph

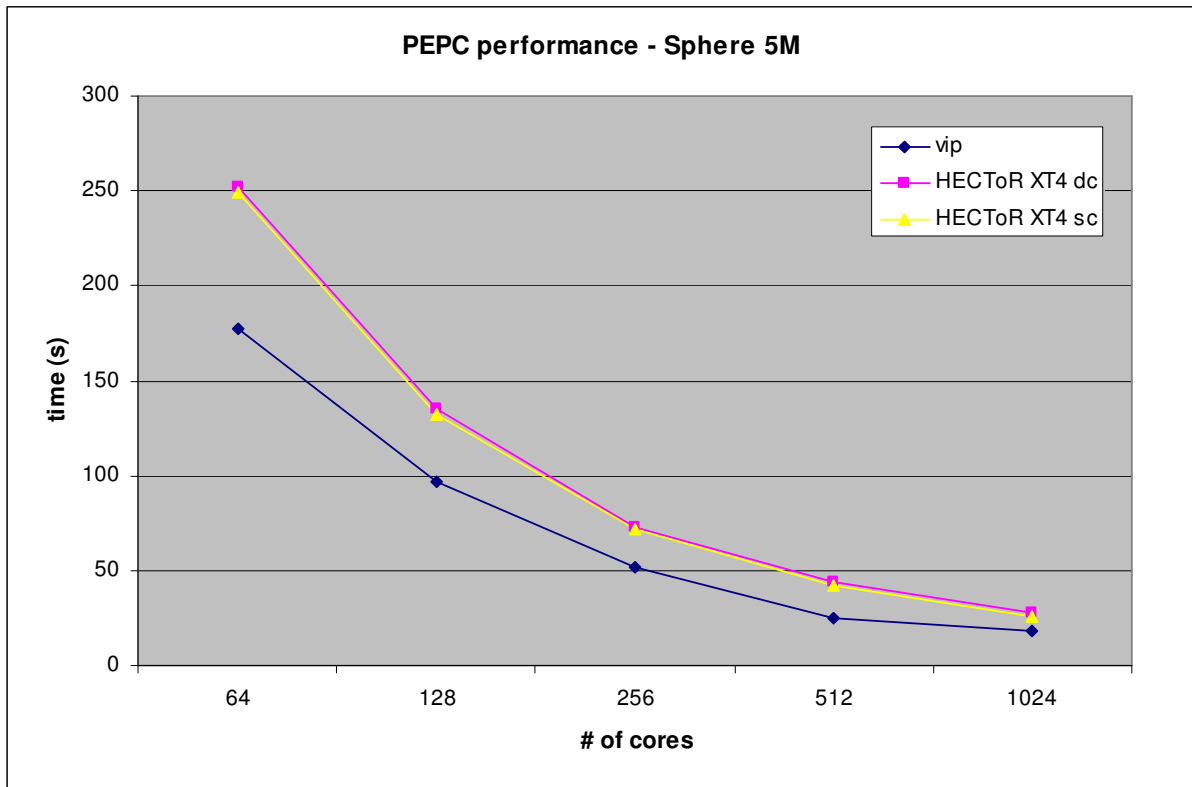


Figure 10 – PEPC performance graph⁵

⁵ Performance numbers for JUMP and Louhi are incomplete and are therefore not shown in this graph.

3.5.6 Plasma Physics

GENE's *strong_512* dataset does not fit into memory under 256 cores on any of the systems; the timings from 256 to (where possible) 2048 are summarised in Figure 9. GENE shows excellent performance and scaling up to 2048 cores on the IBM Power6 system. The application clearly seems to benefit greatly from the 4.7GHz CPU.

PEPC behaves very similarly to GENE (see Figure 10), with *vip* again showing speed superior to that of *HECToR*. The performance of single and dual core mode is nearly identical, which leads to the conclusion that memory bandwidth does not influence the run times of the application. Computation clearly outweighs inter-processor communication.

3.5.7 Quantum Chromodynamics

Figure 11 summarises the performance of SU3_AHIGGS on a 256^3 lattice with 100 iterations. Although the application is memory bandwidth limited, the difference in performance between single core and dual core mode is negligible. As would be expected given the increase in clock speed and memory, the performance on *HECToR* is slightly better than the performance in *Louhi*.

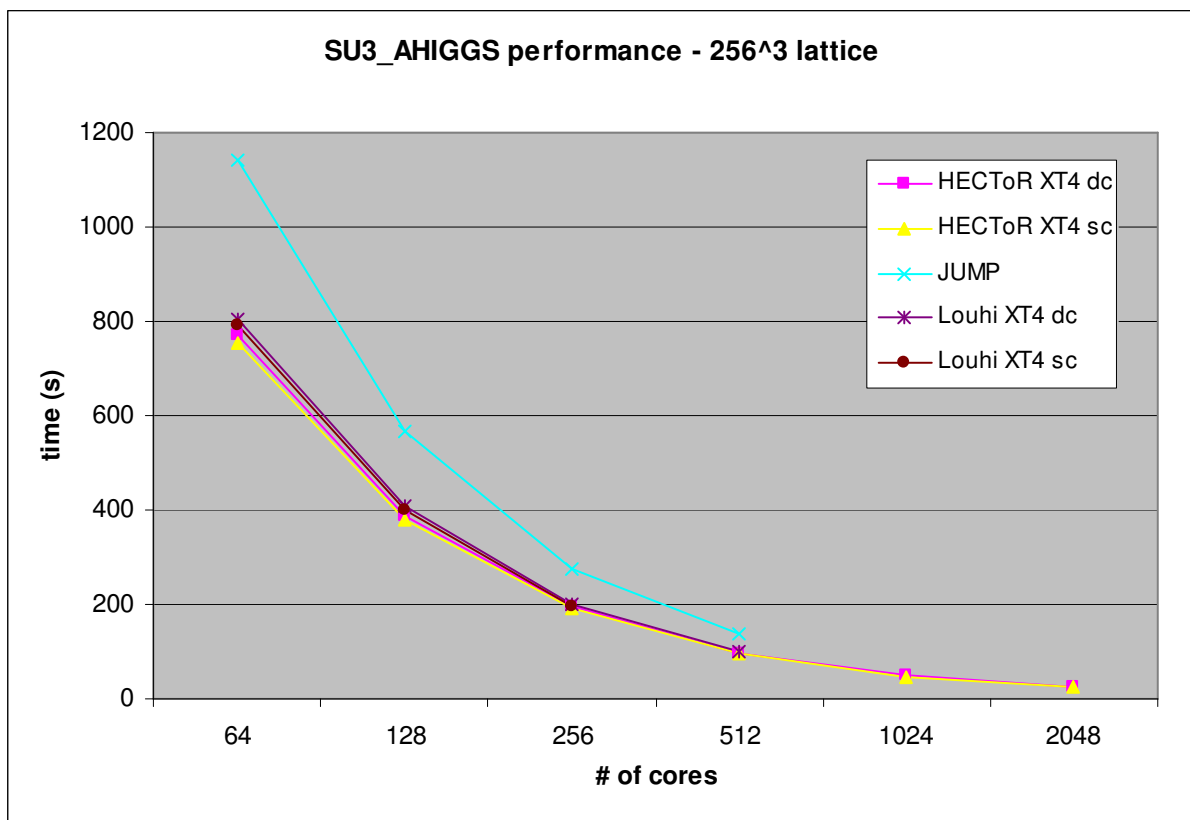


Figure 11 – SU3_AHIGGS performance graph

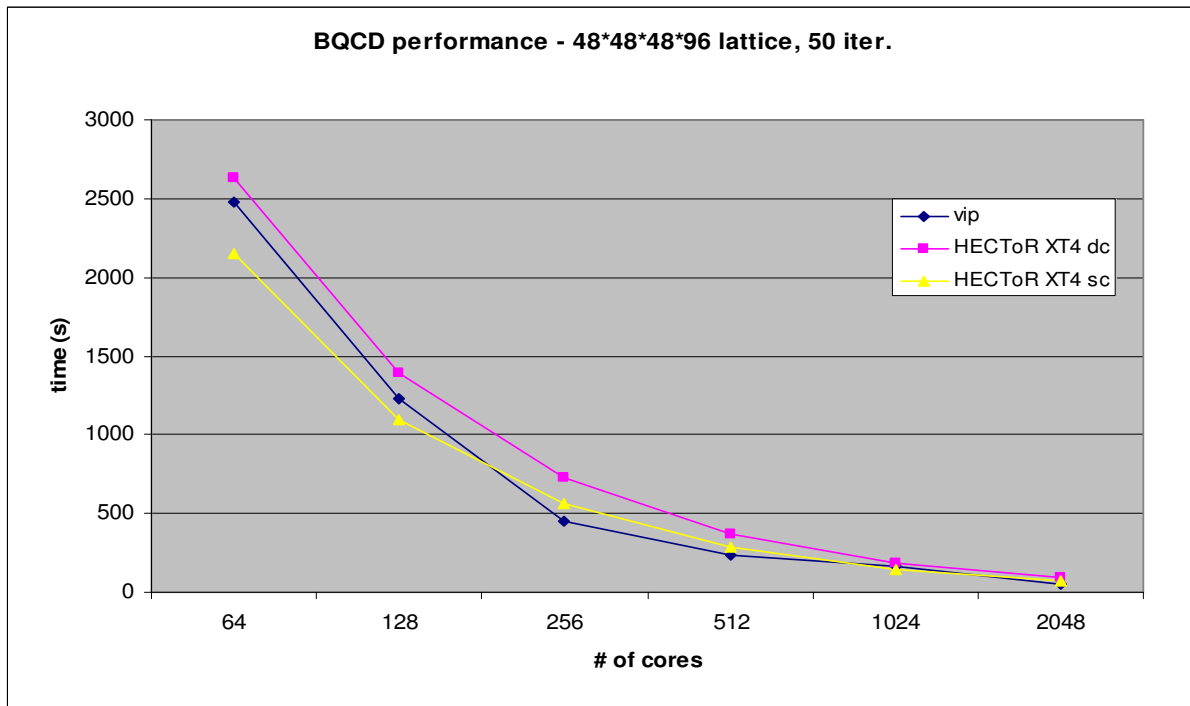


Figure 12 – BQCD performance graph

The performance of BQCD on *HECToR* and *vip* is summarised in Figure 12 (there are no performance numbers for this dataset on *JUMP* and *Louhi*). As the graph shows, the code benefits from the increased memory bandwidth that comes with single core runs on the XT4 and is fastest on *HECToR* for low processor counts. The preliminary results need investigation in more depth.

3.5.8 Results on the Cray X2

As mentioned earlier, only four of the DEISA benchmark applications were selected to be ported to *HECToR*'s vector component, the Cray X2. Two of these applications are known to successfully run on vector systems (namely *Fenfluss* and *CPMD*), while the remaining two codes, *SU3_AHiggs* and *IQCS*, have never previously been ported to a vector system.

Fenfluss, which runs successfully on the NEC SX-8 system at HLRS, was straightforward to port onto the X2. The absolute performance of the application is poorer, with run times increasing by a factor of three, yet the scaling is super linear. This drop in performance can be explained by memory bandwidth (a limiting factor for this code) which is 28.5 GB/s on the X2 and 64 GB/s on the SX-8. Comparing the performance to the XT4, the performance increase is approximately a factor of 4; this is achieved without significant optimisation of the code for this platform.

IQCS suffers from memory limitations on the X2 and only small datasets were therefore run successfully. Although the code scales very well, the absolute performance is not satisfactory. Comparing the 31 Qubit runs on 64 cores on the XT4 and the X2, the performance increase is approximately 10%. However for a system such as the X2, an increase in performance by a factor of 6 would be the target. The results for *IQCS* suggest the code does not vectorise very well.

The performance of *SU3_AHIGGS* on the X2 is extremely poor – the code does not vectorise. In order to potentially increase the code's performance, a considerable amount of effort would have to be spent on this application, which is beyond the scope of the Benchmark maintenance task.

4 Usage of the DEISA Benchmark Suite

The Benchmark Suite is aimed at both internal (i.e. DEISA) and external users. This section will summarise the effort that enables the availability of the DEISA Benchmark Suite on the web, as well as the current use of the Suite outside DEISA.

4.1 Maintenance of the Website

The DEISA Benchmark Suite website provides a gateway for distributing the software to users and supply documentation and performance results. Part of the maintenance of the Benchmark Suite thus needs to involve the updating of the website contents.

As a first step, the website needs to reflect that the content of the Benchmark Suite was modified, and therefore all references and links to DL_POLY needed to be removed. For ECHAM5 a note was added to explain that, although this code is still available for download, it is no longer actively supported. IFS has not been added to the list of applications yet and is thus far not available for download; this will be done as soon as the code is fully integrated and tested, and once the licensing details have been sorted out.

In addition to the updates that relate to the Benchmark Suite content, the performance results achieved on the new platforms were added to the “Results” section of the website. Information on what compilers, compiler flags and libraries were used to achieve the performance results for each application are also published in order to allow users to reproduce these results independently if they wish, or even help them with porting the applications to an unsupported platform.

The new platform specific JuBE framework files, which integrate each application into the Benchmark Suite and which enable user-friendly compilation, execution and verification of the benchmarks need to be uploaded to the website. New packages containing these additional files need to be created, both for the entire Benchmark Suite (containing all the applications) and for each application separately, and be made available to users.

4.2 Access and Download Statistics

In order to gain a picture of how often the Benchmark Suite (or parts of it) was downloaded since the start of the DEISA2 project, some statistics were extracted from the web server. According to the server access logs, 300 package files were downloaded in the five months between the start of May and the end of September 2008.

Figure 13 shows that, after a slow start with only one download in May, there has been steady access to the download section of the Benchmark Suite website. It is interesting to see that the separate application packages seem to be as popular with users as the package containing the entire Suite. Unfortunately the statistics currently do not contain enough data to extract information about the user base.

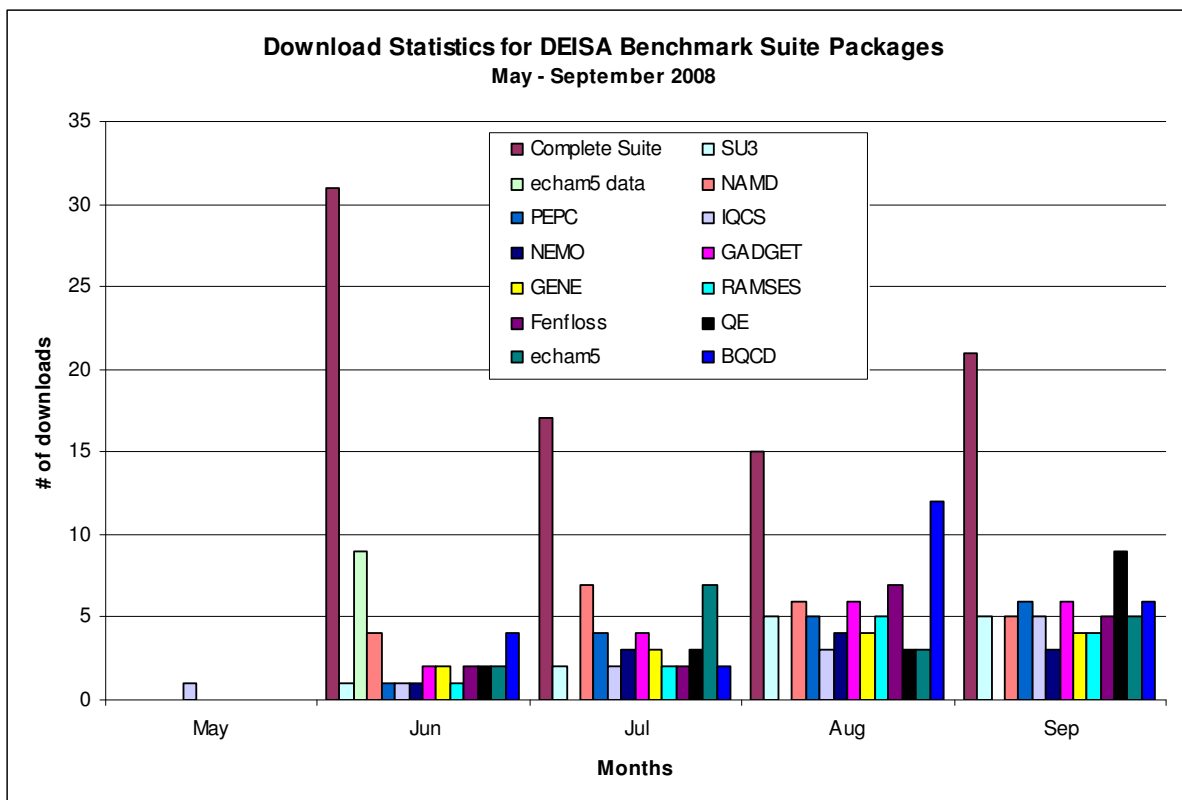


Figure 13 – Download statistics for the DEISA Benchmark Suite Packages between May and September 2008

4.3 External Usage of the Benchmark Suite

To date the most high-profile “user” of (at least part of) the DEISA Benchmark Suite is PRACE. A significant number of applications from the DEISA Benchmark Suite also form part of the PRACE benchmarks⁶. PRACE used the DEISA Benchmark Suite as a starting point for the application and software analysis that led to setup of their benchmark.

As nearly a third of all applications in the PRACE benchmark were already part of the fully integrated DEISA Benchmark Suite, and given that the structure of the DEISA Benchmark Suite software allows for applications and platforms to easily be added or removed from the package, PRACE decided to use the Benchmark Suite provided by DEISA.

Rather than starting a benchmark development from scratch, this allowed PRACE to use a tested software package as the basis of their work. PRACE need to add the remaining applications in their benchmark to the framework, however a variety of different platforms and architectures are already specified in the framework and this will facilitate the integration of additional systems.

PRACE approached the DEISA Benchmark team for information on the JuBE framework and the integration of applications into this framework. The communication channel went through the email address provided on the DEISA benchmarking website, benchmarking@deisa.eu⁷.

⁶ Applications that are in both the DEISA and PRACE benchmarks are: NAMD, CPMD, GADGET, NEMO, ECHAM5, PEPC, as well as the kernels of BQCD and SU3_AHIGGS, which form a QCD benchmark.

⁷ This email address is forwarded to the DEISA benchmarking team, i.e. Task 2 of WP7.

Members of PRACE then downloaded the specific applications from the DEISA the website. A couple of key PRACE staff were given direct access to the DEISA Benchmark Suite development repository. The content of this repository is identical to the content that is available for download on the DEISA website – the main difference to PRACE is that the content of the repository is generally made available online after the development for certain systems was completed and fully tested, which can take several weeks. Giving PRACE direct access allows them to get the most up-to-date development and incorporate it into their own work. The duplication of effort is thus limited. The DEISA Benchmark Suite development can also benefit considerably from a closer collaboration: any work completed by PRACE on new platforms will be passed on to DEISA. The benefit here is twofold: applications that are in both the DEISA and PRACE benchmarks might be ported to a new system by PRACE before it is addressed in DEISA and thus information on the build procedures can be shared by PRACE⁸. In the same way, system specific JuBE files can be shared between PRACE and DEISA, again reducing the duplication of required work.

⁸ This is especially relevant in the case of ECHAM5, which is part of the PRACE benchmark and no longer actively supported in the DEISA Benchmark Suite.

5 Future Work

In the next six months of the project, we will continue porting the applications in the Benchmark Suite to DEISA platforms, in particular addressing the Blue Gene platform. Due to the very specific architecture of the Blue Gene systems, this is very likely to be a challenging task for a number of the applications. In addition, it will be necessary to analyse the datasets that are currently available with the Benchmark Suite and assess for which applications we will need to develop new datasets. Ideally, each application (where applicable) should have three datasets: a small dataset for the testing of the executable; a medium size dataset for runs up to 512 or 1024 cores; a large dataset for runs up to 4096 or 8192 cores. The need for and feasibility of this will need to be investigated and assessed. In addition, we will also investigate the performance on *vip* of those applications for which no definite results could be presented in this document.

Porting the DEISA Benchmark Suite to the BlueGene architecture as well as assessing the current datasets will most likely exhaust the efforts available for the benchmarking task. Nevertheless, one additional aim is to improve on the current state of the website. Ideally, a registration step will be implemented for the download area of the pages: this would allow us to gather additional statistics and possibly relax some of the licence related restrictions on certain applications.

6 Conclusions

To this date, the DEISA Benchmark Suite has been ported to a number of different architectures, which are representative of a large fraction of the DEISA infrastructure. The Benchmark Suite is available online, together with documentation, reference results and compiler information that should allow users to reproduce all performance numbers. The fact that the Benchmark Suite was picked up by PRACE as a starting point for their work on a benchmark shows that the DEISA Benchmark Suite is a flexible and well integrated piece of software.

The content of the Benchmark Suite was revised slightly and two applications were dropped: DL_POLY (for licensing reasons) and ECHAM5. On the other hand, IFS was added to the Benchmark Suite, although it is not yet available for download (IFS is still being integrated to the Benchmark Suite – this work is done under eDEISA).

The past months were focussed on porting the Benchmark Suite to an IBM Power6 cluster and a large Cray XT4 system. A couple of applications were also ported (more or less successfully) to the X2 vector system. One aim of the project is to get an overall view of the performances of different applications on the systems in the DEISA infrastructure. By porting to both the IBM and the Cray system, we added a significant part to the overall performance picture we are trying to build.