



CONTRACT NUMBER 508830

**DEISA**

**DISTRIBUTED EUROPEAN INFRASTRUCTURE FOR  
SUPERCOMPUTING APPLICATIONS**

**European Community Sixth Framework Programme**  
RESEARCH INFRASTRUCTURES  
Integrated Infrastructure Initiative

**FLASH on the DEISA Infrastructure**

Deliverable ID: D-JRA2-3.2

Due date: October, 31, 2006

Actual delivery date: May 28, 2007

Lead contractor for this deliverable: EPCC, University of Edinburgh, UK

Project start date : May 1<sup>st</sup>, 2004

Duration: 4 years

<b>Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)</b>		
<b>Dissemination Level</b>		
<b>PU</b>	Public	X
<b>PP</b>	Restricted to other programme participants (including the Commission Services)	
<b>RE</b>	Restricted to a group specified by the consortium (including the Commission	
<b>CO</b>	Confidential, only for members of the consortium (including the Commission Services)	

## Table of Contents

Table of Contents .....	1
1 Introduction.....	1
1.1 Acknowledgements .....	2
1.2 References .....	2
1.3 Document Amendment Procedure .....	2
1.4 List of Acronyms and Abbreviations .....	2
2 The Improved Virgo Simulation (IVS).....	3
2.1 Platform Independent Optimisations .....	3
2.2 Platform Dependent Optimisations.....	3
3 Code Migration.....	3
3.1 Code Migration method .....	3
3.2 Code Migration implementation.....	4
4 Code performance.....	4
4.1 Memory use.....	4
5 Conclusions.....	4
5.1 Outstanding issues and future work .....	5

## 1 Introduction

This documentation constitutes deliverable D-JRA2-3.2, in its entirety, and outlines a version of FLASH for cosmological simulations on the DEISA infrastructure. FLASH is a highly modular, Fortran90, parallel Adaptive Mesh Refinement (AMR) code which employs MPI.

A version of FLASH, as employed by the Virgo Consortium, has been ported to the DEISA infrastructure, where it was adapted and optimised for cosmological simulations. The code has been further adapted to include the ability to migrate between DEISA platforms during the course of a simulation. This version of the FLASH code is hereafter referred to as the Improved Virgo Simulation, or the IVS.

Originally, the IVS and this report itself, formed the content of public report D-JRA2-3.2, however, FLASH is used under license and cannot be redistributed, therefore, as such, the code will not be publicly available as planned. However, and most importantly, the code is now available to the Virgo consortium. Further, our optimisations to FLASH are expected to be incorporated into the public release of FLASH by the FLASH team in Chicago and, hence, will benefit all registered FLASH users. The code migration scripts are, however, available to the public from the authors.

This report is therefore a brief overview of the final code. The details of the porting, profiling, optimisation and implementation of the final code can be found in [3].

This short report continues as follows. Section 2 outlines the Improved Virgo Simulation code, and includes an overview of the platform independent and independent optimisations introduced by DEISA. Code migration, implemented using both UNICORE and the DEHSL, is discussed in the following section. The next Section briefly discusses performance in terms of execution time and memory use. Finally, the conclusions and suggestions for future work are presented.

## 1.1 Acknowledgements

The software used in this work was in part developed by the DOE-supported ASC / Alliance Center for Astrophysical Thermonuclear Flashes at the University of Chicago. We thank Dan Sheeler, Paul Ricker and Anshu Dubey from the FLASH center for their guidance and support.

We thank the Virgo Consortium for our ongoing our close and fruitful collaboration. In particular, we thank Tom Theuns, John Helly, Craig Booth, Richard Bowers and Carlos Frenk.

Finally, we thank Stefan Haberhauer (NEC) and David Vicente and Judit Gimenez (BSC) for their help with the porting and profiling of the code to their platforms.

## 1.2 References

- [1] ASC / Alliances Center for Astrophysical Thermonuclear Flashes, [flash.uchicago.edu](http://flash.uchicago.edu).
- [2] E. Breitmoser, G. Pringle and M. Antonioletti, *Specifications Document for D-JRA2-3.2*, DEISA deliverable D-JRA2-3.1, EPCC, University of Edinburgh, UK, 2006.
- [3] E. Breitmoser, G. Pringle, O. Bournas and M. Antonioletti, *Documentation for D-JRA2-3.2*, DEISA deliverable D-JRA2-3.3, EPCC, University of Edinburgh, UK.
- [4] Radoslaw Hubert Ostrokski, *Porting, Profiling and Optimising an AMR Cosmology Simulation*, MSc in HPC dissertation, The University of Edinburgh, 2006.

## 1.3 Document Amendment Procedure

Intentionally left blank.

## 1.4 List of Acronyms and Abbreviations

<b>AMR</b>	<b>A</b> daptive <b>M</b> esh <b>R</b> efinement
<b>BSC</b>	<b>B</b> arcelona <b>S</b> upercomputing <b>C</b> enter
<b>FFT</b>	<b>F</b> ast <b>F</b> ourier <b>T</b> ransform
<b>FFTW</b>	<b>F</b> astest <b>F</b> ourier <b>T</b> ransform in the <b>W</b> est
<b>HDF5</b>	<b>H</b> ierarchical <b>D</b> ata <b>F</b> ormat <b>5</b>
<b>HLRS</b>	<b>H</b> och <b>L</b> eistungs <b>R</b> echenzentrum <b>S</b> tuttgart
<b>IDRIS</b>	<b>I</b> nstitut du <b>D</b> eveloppement et des <b>R</b> essources en <b>I</b> nformatique <b>S</b> cientifique
<b>RZG</b>	<b>R</b> echen <b>Z</b> entrum <b>G</b> arching
<b>SARA</b>	<b>S</b> tichting <b>A</b> cademisch <b>R</b> ekencentrum <b>A</b> msterdam

## 2 The Improved Virgo Simulation (IVS)

The Improved Virgo Simulation, or IVS, is an improved version of FLASH2.5, specifically adapted for Virgo's Cosmological Simulations. The key adaptations are the introduction of FFTs, new input routines to read HDF5 files and Code Migration.

We have now ported and profiled the IVS code on the IBM clusters at EPCC (HPCx), IDRIS, RZG, the IBM Linux cluster (Mare Nostrum) at BSC. A port to the SGI at SARA proved problematic and was not successfully completed [3]. The code has also ported and profiled on the NEC SX-8 at HLRS by [4].

### 2.1 Platform Independent Optimisations

FLASH 2.5 employs a Multi-Grid technique as a Poisson solver. The traditional method of parallelising the iterative, Multi-Grid technique involves multiple serial bottlenecks per time step. To avoid this restriction, the IVS employs the Multi-Grid to a certain extent, but then employs an FFT Poisson solver. This FFT solver is non-iterative, i.e. exact, and does not incur any serial bottlenecks.

FLASH 2.5 employs HDF5 for output routines only, thus new HDF5 routines were also introduced to ensure portability across DEISA's heterogeneous infrastructure.

### 2.2 Platform Dependent Optimisations

For each DEISA platform we investigated, we ensured that we used the optimal compiler options.

Replacing sets of MPI calls with alternative formations (i.e. replacing blocking with non-blocking calls) and hand-optimisations, such as removing invariants from loops, etc, both proved ineffective across the DEISA platforms investigated.

## 3 Code Migration

A method of Code Migration was investigated and implemented. In this context, if a code starts running on one particular platform and finishes on another platform of a potentially different architecture or from a different vendor, then the code is said to have migrated. This is not to be confused with Job Migration, where a job is taken from one batch queue and migrated to another platform's batch queue.

This form of code migration was successfully introduced to the IVS, within the DEISA infrastructure, seamlessly utilising several of the DEISA platforms.

Allowing simulations to migrate between DEISA platforms is a very important feature. Not only for large astrophysical simulations, but for any large simulations which require more time to run than any DEISA site permits within a single batch job.

### 3.1 Code Migration method

When the IVS is submitted to a batch queue, that queue's maximum time limit is passed to the code. At each time step, a check is performed to measure how much time remains. If the remaining time is less than the time to run the next iteration, the code dumps a so-called Checkpoint file. This file is sufficient for the same simulation to continue running from the point at which the restart file was created. The code is then automatically resubmitted to the DEISA infrastructure. The process repeats until the simulation is complete.

Only platforms on which the IVS code has already been installed will be used in the migration process, i.e. the executable is *not* staged as this would affect performance. However, the checkpoint files *are* staged, since reading such large datafiles at execution time over any network can be expensive when running on 1000s of CPUs.

### **3.2 Code Migration implementation**

The IVS can invoke code migration on DEISA using either the UNICORE Workflow or a workflow script invoking the DESHL.

A UNICORE Workflow was created, which allowed for automatic job submission and the necessary data staging. The platforms utilised were predetermined. Resource discovery will be implemented once DEISA has brokering.

From a user's point of view, DESHL might be considered as being a command line interface version of UNICORE, however, this is a significant over simplification. One major difference between DESHL and the UNICORE Workflow process is that dependencies between tasks cannot be declared explicitly. Thus we have created a bash script, which runs on a local workstation, that manages the job submissions, job monitoring and staging of data, using the DESHL as it engine.

## **4 Code performance**

The Improved Virgo Simulation code is now up to a factor of 9 faster than the original form of the Virgo Simulation code, and the time spent in communications was reduced from 49% to 34%. The code scales reasonable well for such relatively small simulations. The fastest system was found to be HPCx (IBM Power5), followed by HLRS (NEC SX/8), RZG/IDRIS (IBM Power4) and then BSC (IBM PowerPC).

### **4.1 Memory use**

The size of the possible simulation is determined by the amount of memory available. Larger simulations were not possible as the code employs more memory than expected. It was found that the new FFT routines use a liberal amount of memory but, further, it is suspected that the old, serial HDF5 v1.4.4 output routines are to blame.

## **5 Conclusions**

This report is only a brief outline of the final code on DEISA. A more detailed description can be found at [3].

The Improved Virgo Simulation code has been ported onto a number of DEISA platforms. A good speed-up of up to a factor of 8.9 could be achieved over the Virgo Simulation code by replacing the multi grid based Poisson solver by a spectral FFT based Poisson solver. This result can possibly still be improved if larger data sets are investigated where the benefit of the FFTW would be clear.

It was the first time any FLASH code had been ported to a vector machine, namely the NEC SX-8 at HLRS [4]. Despite of the FFTW package being specially optimized for cache-based architectures, the IVS competed very well when compared with the other architectures, and indeed, the performance of FFTW was not the limiting factor on the SX-8.

Code Migration has been introduced to FLASH, allowing long astrophysical simulations to migrate between DEISA platforms. Large FLASH simulations, which require more time to run than any DEISA site permits within a single batch job, can

now run to completion, seamlessly utilising several of the DEISA platforms if required. This Code Migration is not possible without key elements of the DEISA infrastructure, namely either the UNICORE Workflow manager or the DESHL.

### **5.1 Outstanding issues and future work**

We have ported the IVS to DEISA to allow *long* cosmological simulations, but to allow for *larger* simulations, especially on HPC systems that do not invoke virtual memory, future work remains to be performed. Such work should consider:

- reducing the code's memory use
- refactor the MPI communications, where sets of point-to point communications might benefit by a replacement with global communications, or synchronous replace with standard sends, or blocking calls replaced with non-blocking calls.
- consider a re-write of the remaining time intensive subroutines;
- switch to the latest version of HDF5 and employ parallel I/O;
- investigate FFT packages more appropriate for vector machines for HLRS;
- complete the port to the SGI at SARA.

The work done to date by JRA2 enables the cosmologists of the Virgo Consortium to run their simulations on different platforms and the DEISA infrastructure enables them to produce scientific results exceeding those that are possible at their home installation.

