The DEEP-ER way of approaching Exascale I/O and resiliency

Estela Suarez
Jülich Supercomputing Centre
NorduGrid Conference 2015

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreements 287530 (DEEP) and 610476 (DEEP-ER)

DEEP and DEEP-ER

EU-Exascale projects
20 partners
Total budget: 28,3 M€
EU-funding: 14,5 M€
Joint duration: 5 years

Visit us @ ISC'15, Frankfurt (Germany) 23.-26.07.2015

www.deep-project.eu

www.deep-er.eu

How to Address the Exascale Challenges?

- Power consumption
- Heterogeneity
- Huge levels of parallelism
- Programmability
- Scalability
- Resiliency
- Exploding data requirements
- Algorithms and application readiness

DEEP/-ER way

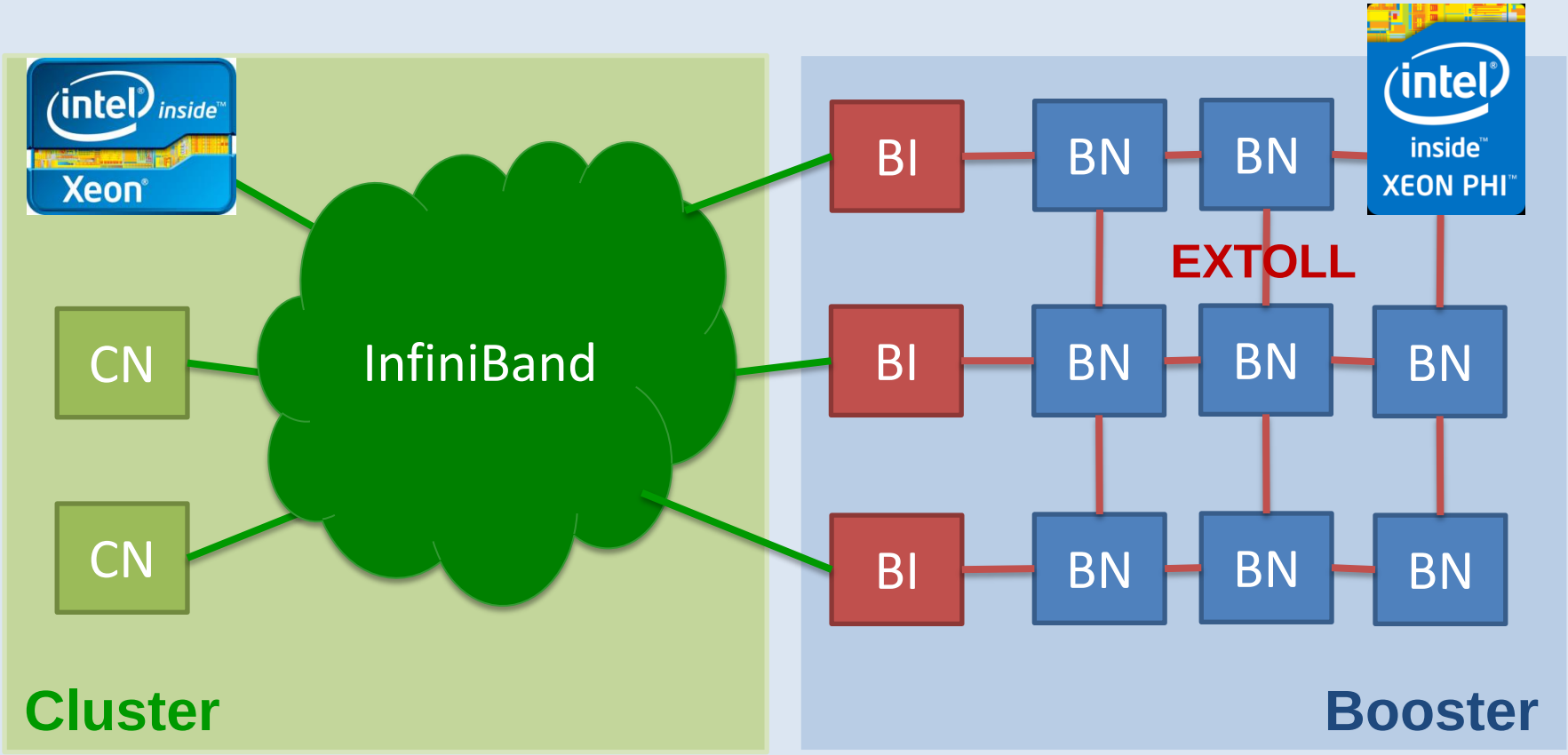
Develop an Exascale architecture tailored to the application requirements

- Match HW characteristics with application scalability patterns
- Exploit benefits of processor heterogeneity
- Profit from new memory technologies
- In an overall energy efficient envelope

reducing burden onto the programmer

- With a complete software stack based on standard components
- Hiding underlying hardware complexity
- Providing a familiar programming environment
- Including tools to analyse and optimise application performance
- Provide I/O and resiliency capabilities for data-intensive apps.

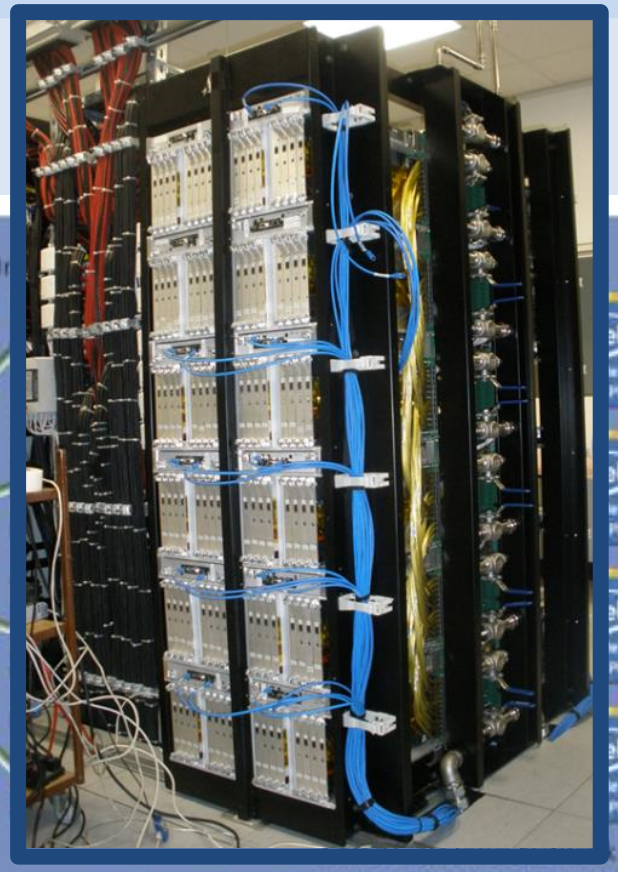
Cluster-Booster architecture



Low/Medium scalable code parts

Highly scalable code parts

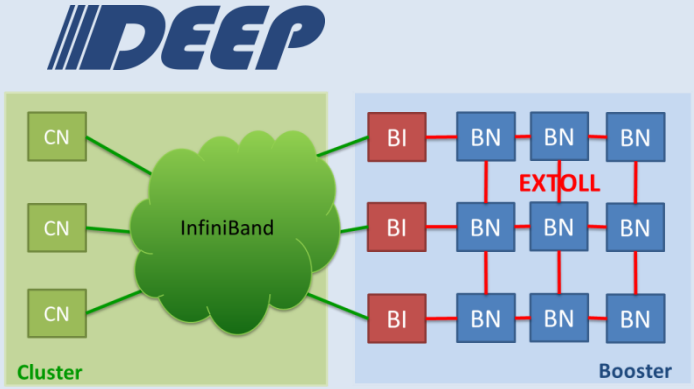
DEEP hardware

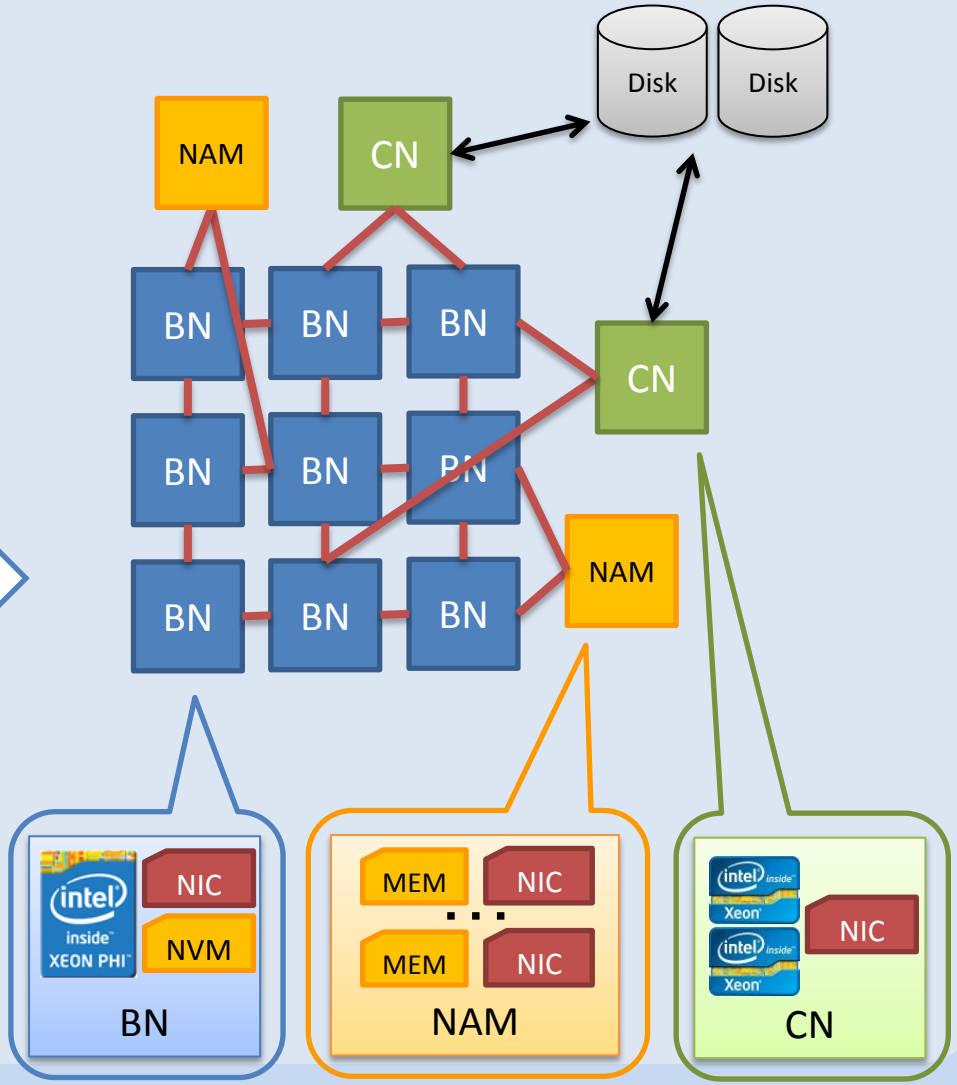


Booster (384 KNC)

Cluster (128 SB)

Enhance DEEP architecture





Legend:

CN: Cluster Node

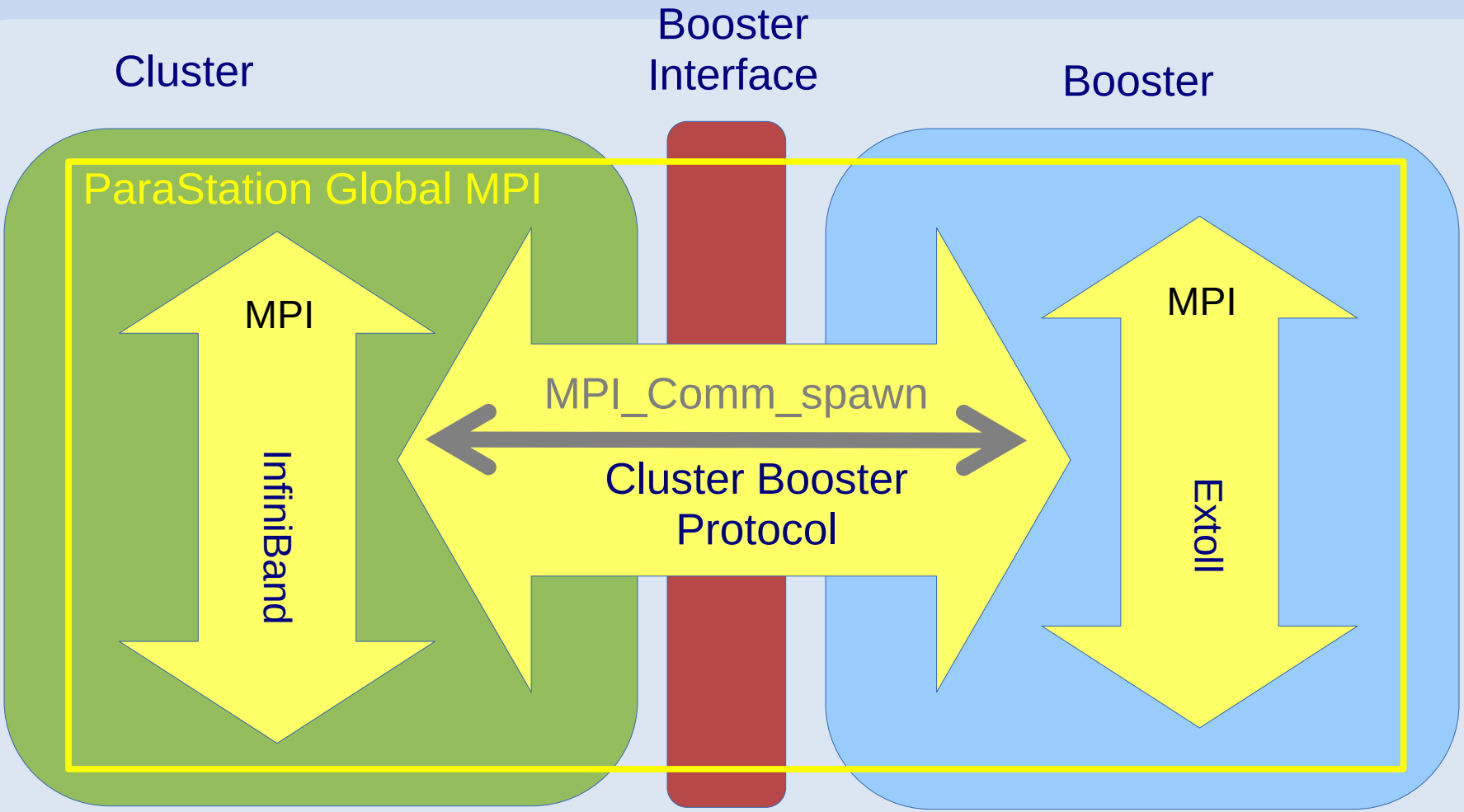
BN: Booster Node

BI: Booster Interface

NAM: Network Attached Memory

NVM: Non Volatile Memory

Programming environment



OmpSs on top of MPI provides pragmas to ease the offload process

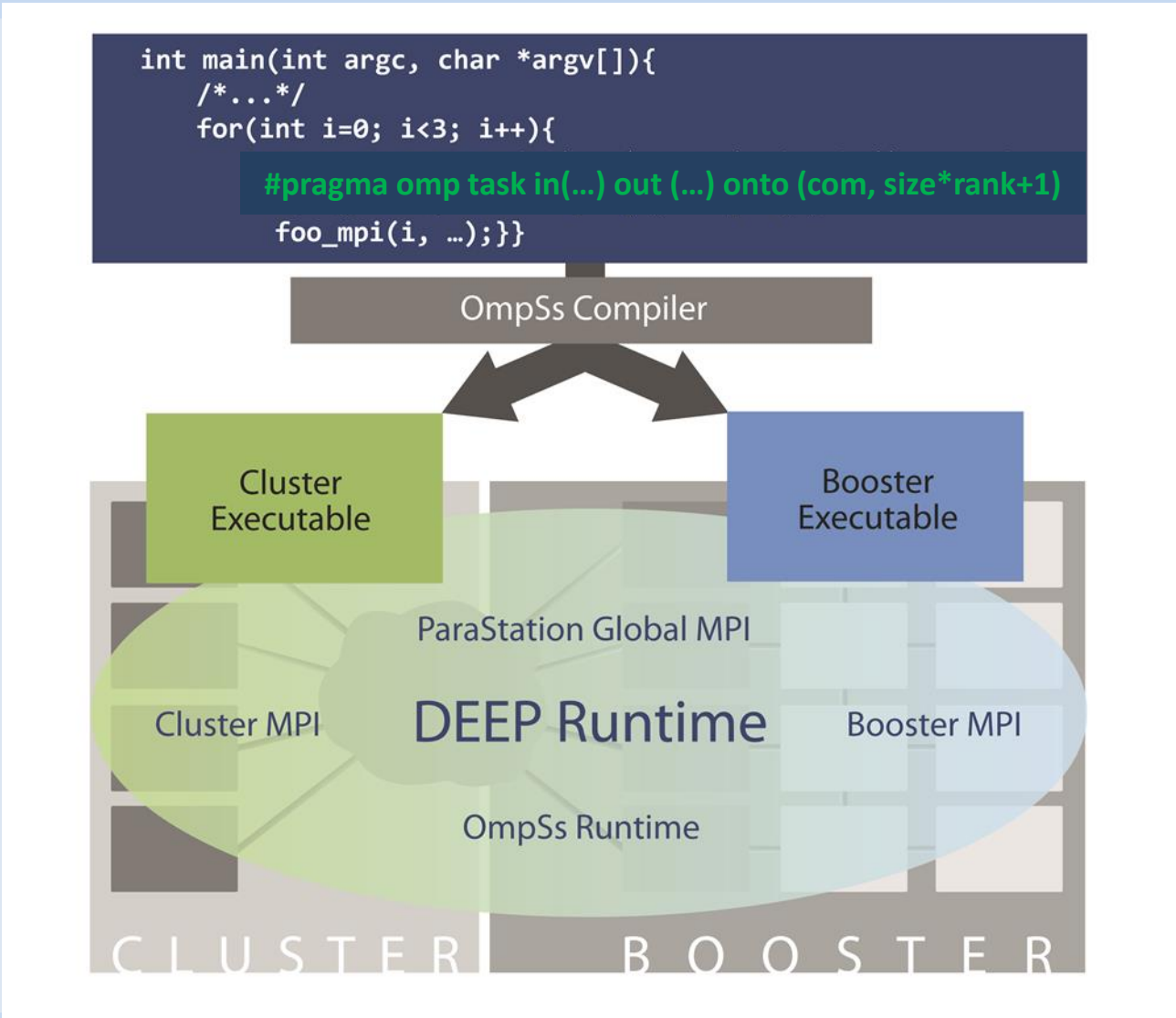
Application running on DEEP/-ER

Source code

Compiler

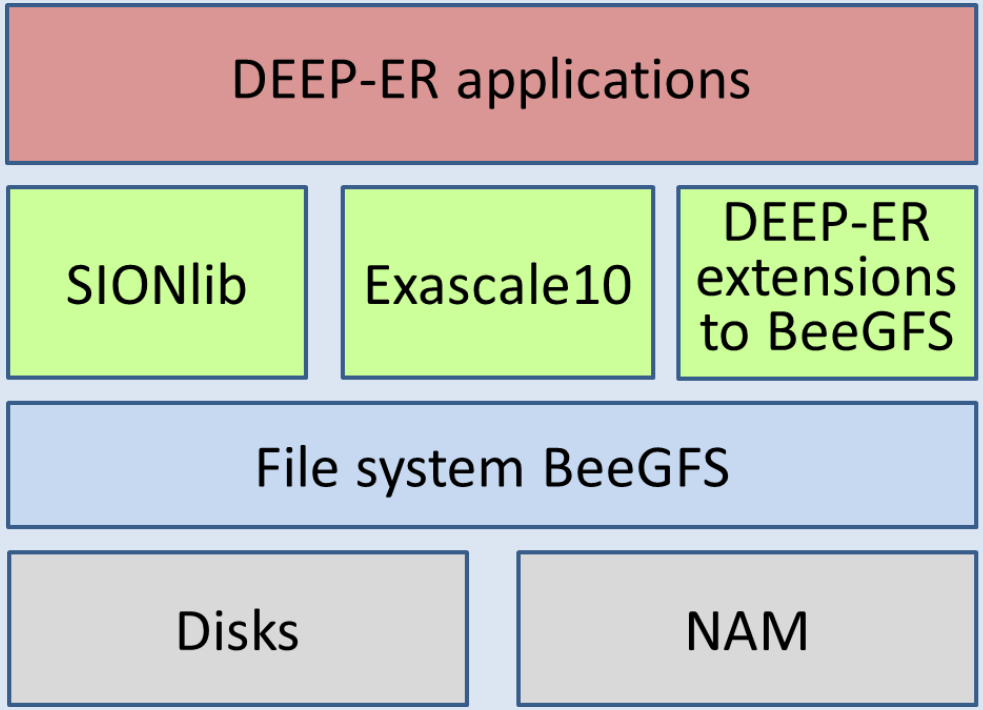
Application binaries

DEEP Runtime



Scalable I/O

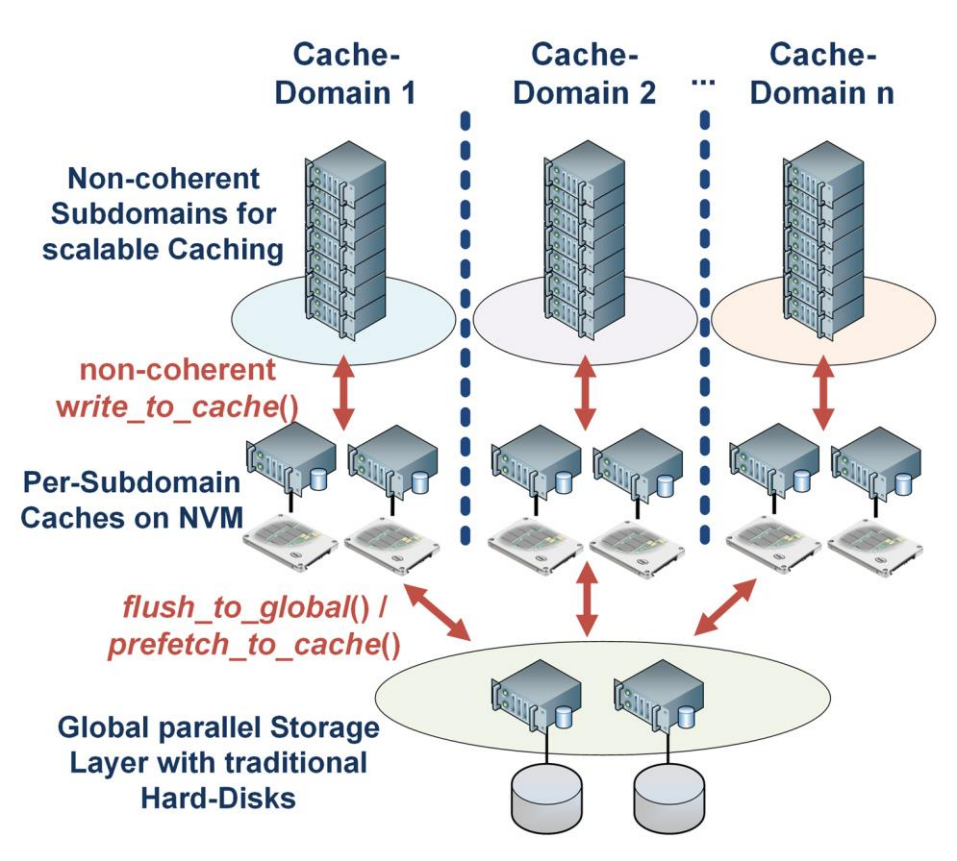
- **Goal: Improve I/O scalability on all usage-levels**
  - BeeGFS leverages architecture and novel memory technologies
  - Extended I/O APIs combine performance with ease of use



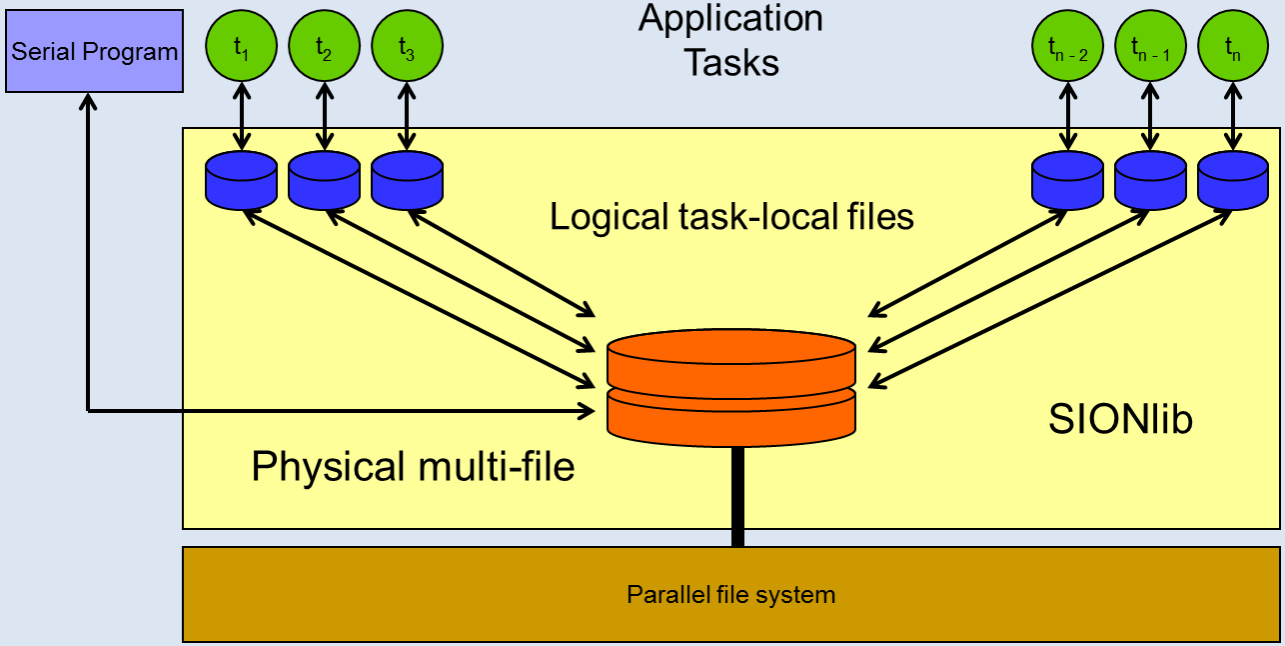
I/O software

BeeGFS® developed by Fraunhofer

- Support underlying hardware
- Two instances:
  - Global FS on HDD server
  - Cache FS on NVM at node
- API for cache domain handling
  - ✓ – Synchronous version implemented
  - WIP – Asynchronous version ongoing
- Functionality to set stripe-size at user-level implemented
  - ✓ – Exploits user knowledge to decide the placement of physical data in the file system
  - ✓ – Co-design request from applications



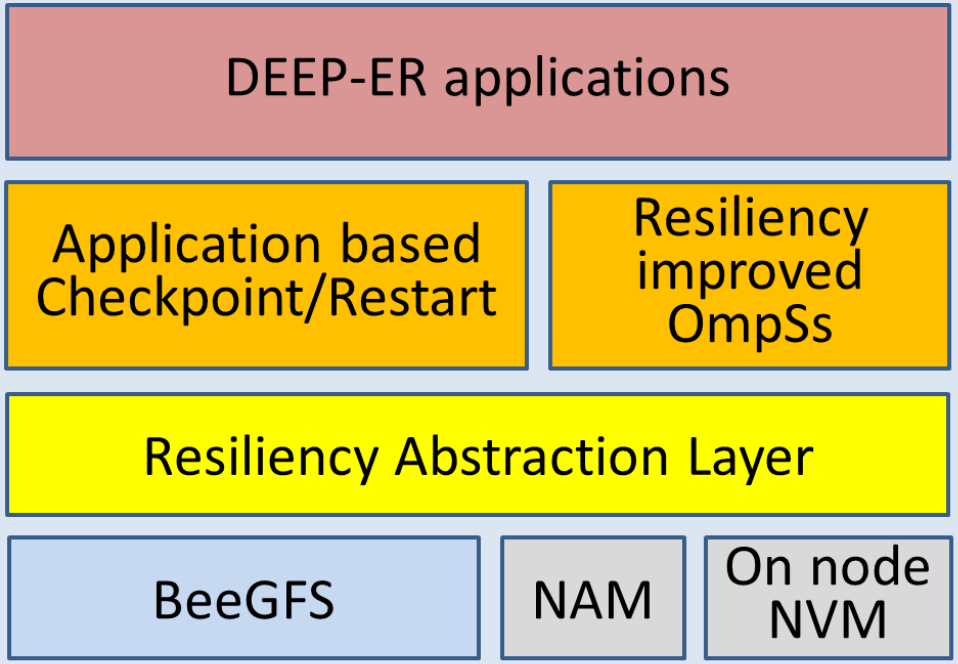
I/O software SIONlib



- Status:
  - ✓ – Restructuration of communication layers
  - ✓ – Used by applications for I/O and checkpointing
  - ✓ – Coordination with SCR for buddy checkpointing
  - WIP – Support BeeGFS cache FS

Resiliency

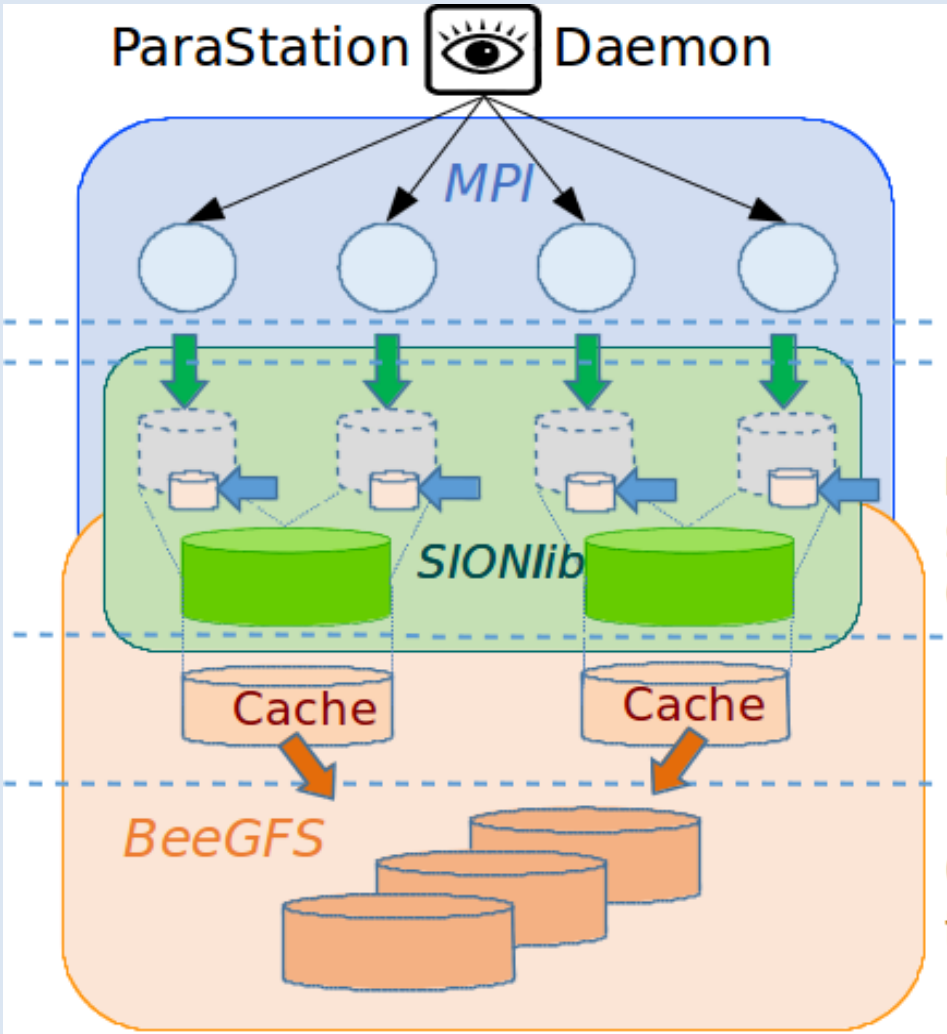
- **Hierarchical, distributed checkpoint/restart scheme**
  - Multi-level checkpointing/restart: NVM, NAM, HDD storage
  - OmpSs extensions for automatic task resiliency



Application-based C/R

- More efficient than transparent system-wide strategies
  - Based on SCR (library developed at LLNL)
- Checkpoint/Restart (C/R) layer will
  - Give application hints on when and where to checkpoint
    - Event-driven Monte Carlo failure model + analytical closed formula
  - Hide meta-data handling from user
- Combine local and global strategies
  - High frequency of local checkpoints
  - Intermediate frequency of buddy checkpoints → use SIONlib and BeeGFS
  - Low frequency of global FS checkpoint → use SIONlib and BeeGFS
- Explore the use of new memory technologies (NAM)

Checkpointing
Integration SCR/SIONlib/BeeGFS



Compute Nodes

CP-Framework API

SIONlib / sion_fwrite()
(node-local I/O view)

Local Buddy Checkpoints

Shared SIONlib Files
(one per BeeGFS Cache Domain)

Non-coherent write()

BeeGFS Cache Domains

BeeGFS API / flush()

Global Storage with traditional Hard-disks

Task-based C/R

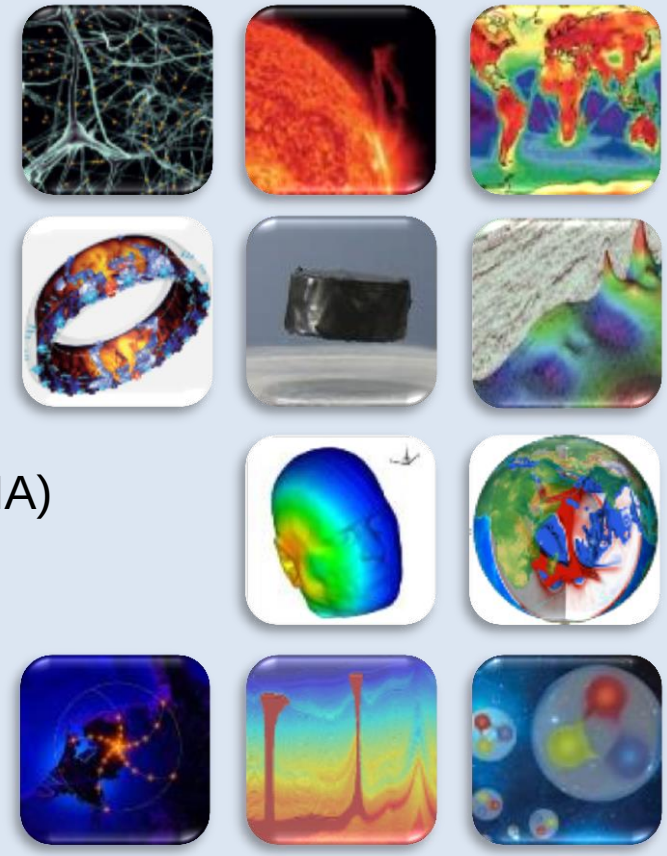
Implemented on:

- Straightforward idea:
  - Save the inputs of the task before running it
  - In case of failure, re-execute the task with the original inputs
- Task-based resiliency provides:
  - A **transparent, fine grained** and **lightweight** protection against **transient errors**
  - Recovers from remote task offloads
- Status:
  - ✓ Design completed
  - WIP – Implementation ongoing
  - WIP – Investigation ongoing on integration with both ParaStation MPI and SCR

Applications

DEEP + DEEP-ER applications

- Brain simulation (EPFL)
- Space weather simulation (KULeuven)
- Climate simulation (CYI)
- Computational fluid engineering (CERFACS)
- High temperature superconductivity (CINECA)
- Seismic imaging (CGG)
- Human exposure to electromagnetic fields (INRIA)
- Geoscience (BADW-LRZ)
- Radio astronomy (Astron)
- Oil exploration (BSC)
- Lattice QCD (UREG)



Goals

- Gather requirements for co-design
- Evaluation of DEEP/-ER architecture and programmability

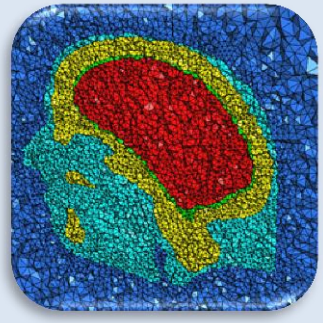
Inria

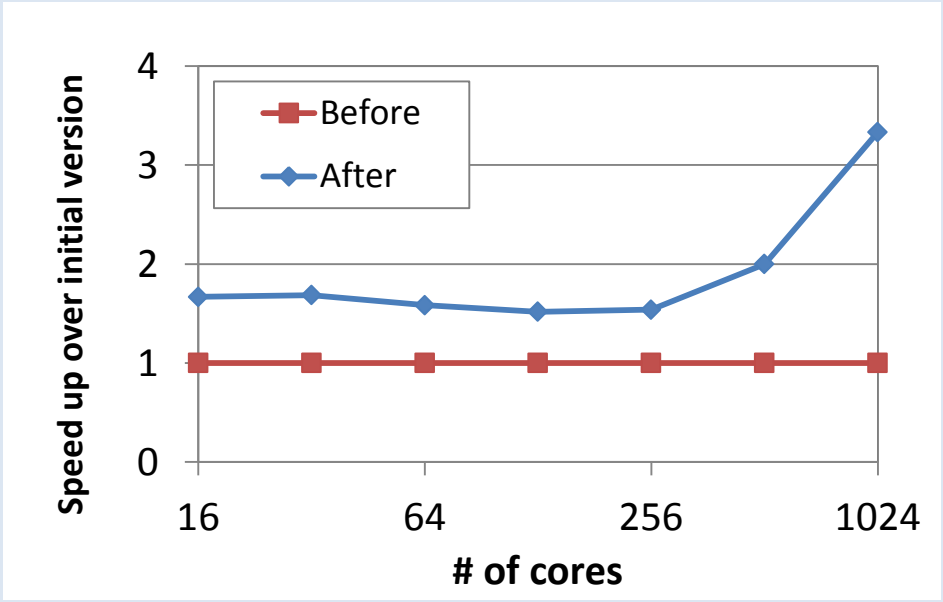
Results will be presented at PARCO2015*

Improvements applied below:

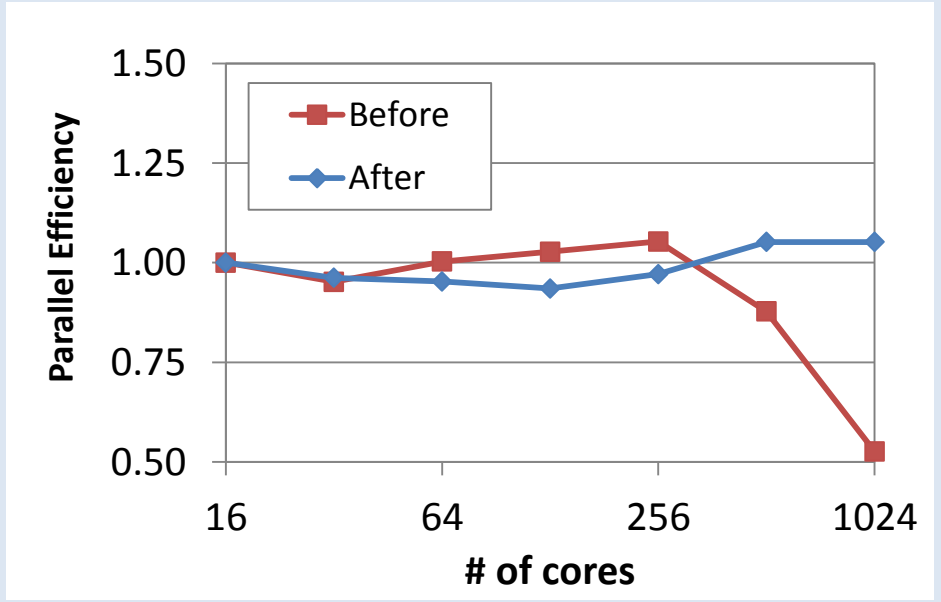
- Non-blocking communication
- Renumbering scheme
- Vectorisation and locality

Setup:
- Human head
- DEEP Cluster
- Mesh: 1.8 million cells
- 16 processes per node
- Pure MPI.
- P1 approximation.





Performance improvement up to 3.3x

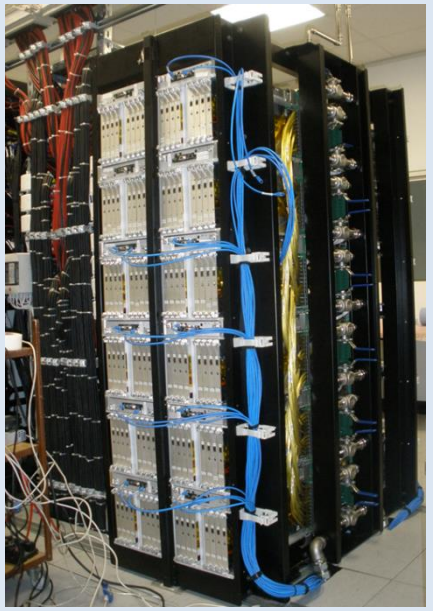


Almost perfect parallel efficiency now

*Leger R., Alvarez Mallon D., Duran A., Lanteri S., "Assessing the DEEP-ER Cluster/Booster Architecture with a fine-element type solver for bioelectromagnetics", Submitted to PARCO2015, Contribution ID: 25. www.parco2015.org

DEEP Status (M42)

- **Hardware status:**
  - DEEP Cluster (@JSC) → Applications work
  - Full Booster (384 KNCs) → in bring-up
  - Energy Efficiency Evaluator (16 KNCs) (@LRZ)
  - ASIC Evaluator (32 KNCs)→ under installation
- **Software status:**
  - Development completed
  - Installation on first Booster Chassis done
  - Tuning and optimization ongoing
- **Scientific Applications:**
  - Optimised and benchmarked in several platforms
  - Demonstration on DEEP Booster will start now



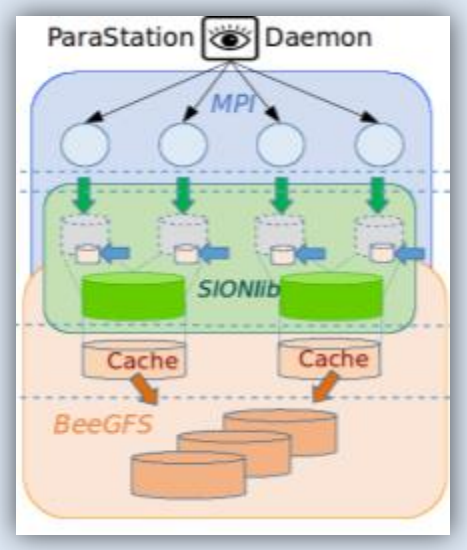
Booster

ASIC Evaluator

DEEP-ER Status (M18)

- **Hardware status:**
  - Overall design finished
  - Prototype under development
  - NAM in development
  - NVM good performance with applications
- **Software status:**
  - Programming environment from DEEP
  - Being extended for I/O and resiliency
    - I/O: BeeGFS, SIONlib, Exascale10
    - SCR, task-based resiliency, failure model
- **Scientific Applications:**
  - Applications analysed, various optimisations done
  - I/O and checkpointing strategies identified

On-node NVM
(Intel DC P3700)



Buddy checkpointing

Take-aways

- **DEEP**: Develop the Cluster-Booster architecture
  - Handles heterogeneity in an innovate way
  - Maps application (scalability) characteristics onto hardware
  - Easy-to-use and familiar programming environment
- **DEEP-ER**: addresses I/O and resiliency
  - Explores new memory technologies
  - Develops scalable I/O strategies
  - Resiliency strategies:
    - Multi-level checkpoint/restart
    - Task-based resiliency
- Using a high variety of applications for co-design and demonstration

Want to know more?

Contact us:

DEEP
pmt@deep-project.eu

DEEP-ER
pmt@deep-er.eu

LinkedIN
http://linkd.in/1KiBe3y

Twitter
@DEEPprojects

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement n° 287530 and n° 610476

Partners

Fraunhofer ITWM

CGGVERITAS

CERFACS

CINECA

The Cyprus Institute

Barcelona Supercomputing Center
Centro Nacional de Supercomputacion

ASTRON

EUROTECH
Imagine. Build. Succeed.

intel

German Research School for Simulation Sciences

EPFL
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

PARTEC CLUSTER COMPETENCE CENTER

KU LEUVEN

Mellanox TECHNOLOGIES

Universität Regensburg

SEAGATE

JÜLICH
FORSCHUNGSZENTRUM

lrz

RUPRECHT-KARLS-
UNIVERSITÄT
HEIDELBERG