SEVENTH FRAMEWORK PROGRAMME
Research Infrastructures

INFRA-2010-2.3.1 – First Implementation Phase of the European High Performance Computing (HPC) service PRACE

PRACE-1IP

PRACE First Implementation Project

Grant Agreement Number: RI-261557

D9.3.3
Report on prototypes evaluation

Final

Version:	1.0
Author(s):	Lennart Johnsson, Gilbert Netzer, SNIC/KTH
Date:	29.03.2013

Project and Deliverable Information Sheet

PRACE Project	**Project Ref. №: RI-261557**	
	Project Title: PRACE First Implementation Project	
	Project Web Site: http://www.prace-project.eu	
	Deliverable ID: D9.3.3	
	Deliverable Nature: Report	
	Deliverable Level: PU	**Contractual Date of Delivery:** 31 / March / 2013
		Actual Date of Delivery: 31 / March / 2013
	EC Project Officer: Leonardo Flores Añover	

* - The dissemination level are indicated as follows: **PU** – Public, **PP** – Restricted to other 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).

Document Control Sheet

Document	**Title: Report on prototypes evaluation**	
	ID: D9.3.3	
	Version: 1.0	**Status:** Final
	Available at: http://www.prace-project.eu	
	Software Tool: Microsoft Word 2007	
	File(s): D9.3.3-GN-20130329-01-final.docx	
Authorship	**Written by:**	Lennart Johnsson, Gilbert Netzer, SNIC/KTH
	Contributors:	Eric Boyer, CINES Stephan Graf, Juelich Wilhelm Homberg, Juelich Giannis Koutsou, CaSToRC Josip Jakic, IPB Radek Januszewski, PSNC Nikola Puzovic, BSC Thomas Roeblitz, SIGMA/UiO Ole Widar Saastad, SIGMA/UiO Björn Schembera, HLRS Georg Schwarz, Juelich Hayk Shoukourian, LRZ Volker Strumpen, JKU Stephane Thiell, CEA Guillaume Colin de Verdière, CEA Torsten Wilde, LRZ
	Reviewed by:	Alan Simpson, EPCC; Dietmar Erwin, FZJ
	Approved by:	MB/TB

Document Status Sheet

Version	Date	Status	Comments
0.1	17/October/2012	Draft	Skeleton with initial structure.
0.2	5/February/2013	Draft	First compilation of contributions.
0.3	12/February/2013	Draft	Result section for WP feedback.
0.4	10/March/2013	Draft	Final WP feedback round.
0.5	15/March/2013	Draft	Internal Review
0.6	25/March/2013	Draft	For MB/TB approval.
1.0	29/March/2013	Final version	

Document Keywords

Keywords:	PRACE, HPC, Research Infrastructure

Table of Contents

List of Figures

List of Tables

References and Applicable Documents

[ACML] AMD Core Math Library, http://developer.amd.com/tools/cpu-development/amd-core-math-library-acml/

[Agarwal-1994] A High Performance Parallel Algorithm for 1-D FFT, Supercomputing '94, Proceedings, p. 34-40, 14-18 Nov 1994 doi:10.1109/SUPERC.1994.344263, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=344263&isnumber=8021

[AMD-2011] Practical Power Gating and Dynamic Voltage/Frequency Scaling, S. Kosonocky, http://www.hotchips.org/wp-content/uploads/hc_archives/hc23/HC23.17.1-tutorial1/HC23.17.111.Practical_PGandDV-Kosonocky-AMD.pdf

[AMD-Intel] The Mother of All CPU Charts 2005/2006, Nov 2005, http://www.tomshardware.com/reviews/mother-cpu-charts-2005,1175.html

[ATLAS] Automatically Tuned Linear Algebra Software (ATLAS), http://www.netlib.org/atlas

[Belady-2007] In the Data Center, Power and Cooling Costs More Than the IT Equipment It Supports, C. Belady. February 1, 2007, Electronics Cooling. http://www.electronics-cooling.com/2007/02/in-the-data-center-power-and-cooling-costs-more-than-the-it-equipment-it-supports

[BLAS] Basic Linear Algebra Subprograms, http://www.netlib.org/blas/

[Brill-2008] Findings on Data Center Energy Consumption Growth May Already Exceed EPA's Prediction Through 2010!, K. G. Brill, The Uptime Institute, 2008, http://uptimeinstitute.org/content/view/155/147

[Calxeda] http://www.calxeda.com

[Curley] Open Innovation 2.0 and 21st Century Industrial Research, M. Curley, http://www.intel.com/content/dam/www/public/emea/eu/en/documents/eric/day1-martin-curley.pdf

[Daily-2009] Weather supercomputer used to predict climate change is one of Britain's worst polluters, Daily Mail, August 27, 2009, http://www.dailymail.co.uk/sciencetech/article-1209430/Weather-supercomputer-used-predict-climate-change-Britains-worst-polluters.html

[Dally-2008] Efficient Embedded Computing, W.J. Dally, J. Balfour, D. Black-Shaffer, J. Chen, R. C. Harting, V. Parikh, J. Park, and D. Shef, IEEE Computer, pp. 27-32, July, 2008

[DARPA-2008] ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems, DARPA, September 28, 2008, http://users.ece.gatech.edu/mrichard/ExascaleComputingStudyReports/exascale_final_report_100208.pdf

[Dhry] A synthetic computing benchmark program, http://www.ct.se/dhrystone/index.html

[EPA-2007] Report to Congress on Server and Data Center Energy Efficiency, U.S. Environmental Protection Agency, Energy Star Program. August 2, 2007, Public Law 109-431, http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf

[ESFRI-2006] European Roadmap for Research Infrastructures, Report 2006, The European Strategy Forum for Research Infrastructures (ESFRI), ftp://ftp.cordis.europa.eu/pub/esfri/docs/esfri-roadmap-report-26092006_en.pdf

[ESSL] Engineering Scientific Subroutine Library, http://publib.boulder.ibm.com/epubs/pdf/a2322683.pdf

[EuroBen] EuroBen Benchmark Programs, http://www.hpcresearch.nl/euroben/programs.php

[Eurostat-2013] Electricity Prices for Industrial Consumers, Eurostat, http://epp.eurostat.ec.europa.eu/tgm/table.do?tab=table&plugin=0&language=en&pcode=ten00114

[Feng-2005] A Power Aware Run-Time System for High Performance Computing, C. Hsu and W. Feng, http://public.lanl.gov/radiant/pubs/sss/sc2005.pdf

[FFTPACK] FFTPACK, http://www.netlib.org/fftpack/

[FFTW] FFTW, http://www.fftw.org/

[Frigo-1999] Cache Oblivious Algorithms, M. Frigo, C. E. Leiserson, H. Prokop, and S. Ramachandran, in 40th Annual Symposium on Foundations of Computer Science, pp. 285-298, NewYork, USA, October 1999

[FZJ] SIONlib: Scalable I/O Library for Parallel Access to Task-Local Files, http://www.fz-juelich.de/ias/jsc/EN/Expertise/Support/Software/SIONlib/_node.html

[GF-2008] Towards Ultra-High Resolution Models of Climate and Weather, M. Wehner, L. Oliker, J. Shalf, http://crd-legacy.lbl.gov/~oliker/papers/IJHPCA08_Wehner.pdf

[Godunov] A Difference Scheme for Numerical Solution of Discontinuous Solution of Hydrodynamic Equations, Godunov, S. K. (1959), Math.

Sbornik, 47, 271–306, translated US Joint Publ. Res. Service, JPRS 7226, 1969.

[Green500-2007] www.green500.org/lists/green200711

[Green500-2012] http://www.green500.org/

[HPCC] HPC Challenge Benchmarks, http://icl.cs.utk.edu/hpcc/

[Horowitz-2009] Scaling Power and the Future of CMOS, M Horowitz, 2009, https://www.e3s-center.org/events/09/symposium/index.htm

[Hybrid] Hybrid MPI-OpenMP Programming, P.-Fr. Lavallée and Ph. Wautelet, http://www.idris.fr/data/cours/hybride/choix_doc.html

[Hydro] Hydro Benchmark, https://github.com/HydroBench/Hydro.git

[IBM-2009] High-Temperature Cooling: Towards a Zero-Emission Datacenter, I. Meijer, T. Brunschwiler, S. Paredes and B. Michel, IBM, 1st European Workshop on HPC Centre Infrastructures, http://www.cscs.ch/events/event_list/event_detail/index.html?tx_seminars_pi1%5BshowUid%5D=7

[IBM-2010] Harnessing the Adaptive Energy Management Features of the POWER7 chip, M. Floyd, B. Brock, M. Ware, K. Rajamani, A. Drake, C. Lefurgy, and L. Pesantez, IBM, HotChips, August, 2010, http://www.hotchips.org/uploads/archive22/HC22.23.130-1-Floyd-IBM-POWER7-Power-Management.pdf

[IBM-2011] IBM Deep Computing XXL/SciComp Conference, Paris, May 11 2011, L. Brochard http://spscicomp.org/wordpress/wp-content/uploads/2011/05/brochard-Luigi_Perf-and-Power-Trade-off.pdf

[Inefficient-2007] An In-Efficient Truth, Global Action Plan, Report, December 2007, http://www.greenict.org.uk/reports-library/an-inefficient-truth

[Intel-2006] Energy per Instruction Trends in Intel® Microprocessors, E. Grochowski, M. Annavaram, Technology@Intel Magazine, March 2006, http://support.intel.co.jp/pressroom/kits/core2duo/pdf/epi-trends-final2.pdf

[Intel-2007] An 80-tile 1.28 Tflops Network-on-Chip in 65 nm CMOS, S. Vangal, J. Howard, G. Ruhl, S.Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, N. Borkar. February 11-15, 2007, pp. 98 – 99, IEEE Solid-States Circuits Conference, San Francisco, http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4242283

[Intel-2011] Power management architecture of the 2nd generation Intel® Core™ microarchitecture, formerly codenamed Sandy Bridge, E. Rotem, A. Naveh, D. Rajwan, A. Ananthakrishnan, E. Weissmann, http://www.hotchips.org/wp-content/uploads/hc_archives/hc23/HC23.19.9-Desktop-CPUs/HC23.19.921.SandyBridge_Power_10-Rotem-Intel.pdf

[Intel-2012] Intel's Near Threshold Voltage Computing and Applications, D. Kanter, September 18, 2012. http://www.realworldtech.com/near-threshold-voltage/

[Intel-clock] Intel Processor Clock Speed (MHz), http://smoothspan.files.wordpress.com/2007/09/clockspeeds.jpg

[IOR] IOR: I/O Performance Benchmark, https://asc.llnl.gov/sequoia/benchmarks/IOR_summary_v1.0.pdf

[Jovanovic-2012] FPGA Accelerator for Floating-Point Marix Multiplication, Computers & Digital techniques, vol. 6, no. 4, pp 249 – 256, July 2012, http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=6337379

[Koomey-2009a] Assessing Trends over Time in Performance, Costs, and Energy Use for Servers, J. G. Koomey, C. Belady, M. Patterson, A. Santos, K-D. Lange. August 17, 2009, Intel, http://www3.intel.com/assets/pdf/general/servertrendsreleasecomplete-v25.pdf

[Koomey-2009b] Assessing in the Trends in the Electrical Efficiency of Computation over Time, Intel, August 17, 2009. J.G. Koomey, S. Berard, M. Sanchez, H. Wong., http://download.intel.com/pressroom/pdf/computertrendsrelease.pdf

[LINPACK] HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers. September, 2008. A. Petitet, R. C. Whaley, J. Dongarra, A. Cleary. http://www.netlib.org/benchmark/hpl/

[Marvell] http://www.marvell.com

[Mills-2009] SuperComputers: Super-Polluters? Data Center Journal vol. 7, May, p. 16–18 (2008) Evan Mills, William Tschudi, John Shalf, and Horst Simon

[MKL] Intel Math Kernel Library, http://software.intel.com/en-us/intel-mkl

[Moore-1965] Cramming More Components onto Integrated Circuits, Electronics, Vol. 38, no 8. April, 1965. G. E. Moore, ftp://download.intel.com/research/silicon/moorespaper.pdf

[Mora-2010] Private Communication.

[Mora-2012] Understanding Bulldozer architecture through Linpack benchmark, J. Mora, ISC 2012, http://www.hpcadvisorycouncil.com/events/2012/European-Workshop/Presentations/12_AMD.pdf

[MPI] MPI - The Message Passing Interface Standard, http://www.mcs.anl.gov/research/projects/mpi/

[MPrime] Mersenne Prime, http://en.wikipedia.org/wiki/Prime95

[NEST] Neural Simulation technology, http://www.nest-initiative.org/index.php/index.php

[NUMAscale] http://www.numascale.com

[NVIDIA-2010] CUDA Accelerated Linpack on Clusters, E. Phillips, NVIDIA, GPU Technology Conference, September, 2010, http://www.nvidia.com/content/GTC-2010/pdfs/2057_GTC2010.pdf (Also presented at SC10)

[NYT-2012a] Power, Pollution and the Internet, New York Times, J. Glanz, 2012-09-22, http://www.nytimes.com/2012/09/23/technology/data-centers-waste-vast-amounts-of-energy-belying-industry-image.html?pagewanted=all&_r=0

[NYT-2012b] Data Barns in a Farm Town, Gobbling Power and Flexing Muscle, New York Times, J. Glanz, 2012-09-23, http://www.nytimes.com/2012/09/24/technology/data-centers-in-rural-washington-state-gobble-power.html?pagewanted=all

[OpenMP] OpenMP - Open Multi-Processing specifications, http://openmp.org/wp/openmp-specifications/

[Pollack-1999] New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies, Pollack, F., Proceedings of the 32nd Annual IEEE/ACM International Symposium on Microarchitecture, Haifa, Israel.

[QCD-2013] Computational Challenges in QCD Thermodynamics, C. DeTar and F. Karsch, February 6, 2013, http://www.usqcd.org/documents/13thermo.pdf

[QUDA] QUDA: A Library for QCD on GPUs, http://lattice.github.com/quda/

[RAMSES] The RAMSES Code, http://irfu.cea.fr/Sap/en/Phocea/Vie_des_labos/Ast/ast_visu.php?id_ast=904

[Rose] Approximate Riemann solvers, parameter vectors and difference schemes, P. L. Rose, (1981), J. Comput. Phys. 43 (2): 357–372, Bibcode 1981JCoPh..43..357R, doi:10.1016/0021-9991(81)90128-5

[RPE] Robinhood Policy Engine, http://sourceforge.net/apps/trac/robinhood

[SeaMicro] http;//www.seamicro.com

[SGIMAID] http://www.sgi.com/products/storage/maid/

[Shalf-2007] About Memory Bandwidth, J. Shalf, June 13, 2007, http://www.csm.ornl.gov/workshops/SOS11/presentations/j_shalf.pdf

[Shalf-2010] Exascale Computing Technology Challenges, John Shalf, National Energy Research Supercomputing Center, Lawrence Berkeley National Laboratory ScicomP / SP-XXL 16, San Francisco, May 12, 2010, http://www.spscicomp.org/ScicomP16/presentations/ExascaleChallenges.pdf

[SorTech] http://www.sortech.de/en/

[SPRASEPACK] Sparsepack, http://people.sc.fsu.edu/~jburkardt/f_src/sparsepak/sparsepak.html

[STREAM-1] The STREAM Benchmark: Computer Memory Bandwidth, J. McCalpin, http://www.streambench.org/

[Strumpen-2004] Stream Algorithms and Architecture, V. Strumpen, H. Hoffman and A. Agarwal, Journal of Instruction-Level Parallelism, vol. 6, September 2004 http://www.jilp.org/vol6/v6paper8.pdf

[Supermicro-2008] Private Communication.

[TI-2012] Multicore DSP+ARM KeyStone II System-on-Chip (SoC), November 2012, http://www.ti.com/lit/ds/symlink/66ak2h12.pdf

[Ung-2010] The History of a Dream: How the Ultimate PC Has Evolved In 15 Years, 2010,G. M. Ung and A. Castle, http://www.maximumpc.com/article/home/history_dream_how_ultimate_pc_has_evolved_15_years

[UPC-1] UPC: Distributed Shared Memory Programming, Book of Wiley Inter-Science (2005), W. Carlson et. al.

[UPC-2] Berkeley UPC, http://upc.lbl.gov

[Wyoming-2010] NCAR's dirty little secret, January 16, 2010, Anthony Watts, http://wattsupwiththat.com/2010/01/16/ncars-dirty-little-secret/

[Wyoming-2012] Wyoming experiences that "giant sucking sound" as new coal fired climate supercomputer is turned on, October 16, 2012 by Anthony Watts, http://wattsupwiththat.com/2012/10/16/wyoming-experiences-that-giant-sucking-sound-as-new-coal-fired-climate-supercomputer-is-turned-on/

[Xilinx-2012] Latest FPGAs Show Big Gains in Floating Point Performance, April 16, 2012 D. Strenski, C. Kulkarni, J. Cappello, and P. Sundararajan, http://www.hpcwire.com/hpcwire/2012-04-16/latest_fpgas_show_big_gains_in_floating_point_performance.html

[Xyratex] http://www.xyratex.com

List of Acronyms and Abbreviations

ACM	Association for Computing Machinery
ACML	AMD Core Math Library
ACS	Adsorption Chiller System
AMD	Advanced Micro Devices
API	Application Programming Interface
APP	Accelerated Parallel Processing
ARM	Formerly known as Advanced RISC Machine
ASIC	Application-Specific Integrated Circuit
ATI	Array Technologies Incorporated (AMD)
ATLAS	Automatically Tuned Linear Algebra Software
BAdW	Bayerischen Akademie der Wissenschaften (Germany)
BG	Blue Gene
BLAS	Basic Linear Algebra Subprograms
BSC	Barcelona Supercomputing Center (Spain)
CaSToRC	Computation-based Science and Technology Research Centre
ccNUMA	cache coherent NUMA
CEA	Commissariat à l’Energie Atomique (represented in PRACE by GENCI, France)
CINECA	Consorzio Interuniversitario, the largest Italian computing centre (Italy)
CINES	Centre Informatique National de l’Enseignement Supérieur (represented in PRACE by GENCI, France)
CMOS	Complementary Metal Oxide Semiconductor
CPU	Central Processing Unit
CRAC	Computer Room Air-Conditioning
CS	ClusterStor
CSC	Finnish IT Centre for Science (Finland)
CSCS	The Swiss National Supercomputing Centre (represented in PRACE by ETHZ, Switzerland)
CUDA	Compute Unified Device Architecture (NVIDIA)
DARPA	Defense Advanced Research Projects Agency
DAS	Direct Attached Storage
DDR	Double Data Rate
DGEMM	Double precision General Matrix Multiply
DIMM	Dual Inline Memory Module
DMA	Direct Memory Access
DMF	Data Migration Facility (SGI)
DP	Double Precision, usually 64-bit floating point numbers
DRAM	Dynamic RAM
DSP	Digital Signal Processor
EC	European Community
EDMA	Enhanced DMA
EESI	European Exascale Software Initiative
EP	Efficient Performance, e.g., Nehalem-EP (Intel)
EPA	Environmental Protection Agency (United States)
EPCC	Edinburg Parallel Computing Centre (represented in PRACE by EPSRC, United Kingdom)
EPSRC	The Engineering and Physical Sciences Research Council (United Kingdom)
eQPACE	extended QPACE, name of the FZJ WP8 prototype

ESFRI	European Strategy Forum on Research Infrastructures; created roadmap for pan-European Research Infrastructure.
ESM	Embedded Server Module
ETHZ	Eidgenössische Technische Hochschule Zuerich, ETH Zurich (Switzerland)
EX	Expandable, e.g., Nehalem-EX (Intel)
FC	Fiber Channel
FDR	Fourteen Data Rate
FFT	Fast Fourier Transform
FFTW	Fastest Fourier Transform in the West
FHPCA	FPGA HPC Alliance
FMA	Fused Multiply Add
FP	Floating-Point
FPGA	Field Programmable Gate Array
FPU	Floating-Point Unit
FZJ	Forschungszentrum Jülich (Germany)
GB	Giga (= 2^{30} ~ 10^9) Bytes (= 8 bits), also GByte
Gbps	Giga (= 10^9) bits per second, also Gbit/s, Gb/s
GB/s	Giga (= 10^9) Bytes (= 8 bits) per second, also GByte/s
GCS	Gauss Centre for Supercomputing (Germany)
GDDR	Graphic Double Data Rate memory
GEDI	Generic Diskless Installer
GEM	General Enclosure Management
GENCI	Grand Equipement National de Calcul Intensif (France)
GF/s	Giga (= 10^9) Floating point operations (usually in 64-bit, i.e. DP) per second, also GFlops/s
GHz	Giga (= 10^9) Hertz, frequency =10^9 periods or clock cycles per second
GbE	Gigabit Ethernet, also GigE
GNU	GNU's not Unix, a free OS
GPFS	General Parallel File System (IBM)
GPGPU	General Purpose GPU
GPU	Graphic Processing Unit
HBA	Host Bus Adapter
HCA	Host Channel Adapter
HDD	Hard Disk Drive
HE	High Efficiency
HMPP	Hybrid Multi-core Parallel Programming (CAPS enterprise)
HP	Hewlett-Packard
HPC	High Performance Computing; Computing at a high performance level at any given time; often used synonym with Supercomputing
HPCC	HPC Challenge benchmark, http://icl.cs.utk.edu/hpcc/
HPL	High Performance LINPACK
HSM	Hierarchical Storage Management
HT	HyperTransport channel (AMD)
HTX	HyperTransport Expansion
IB	InfiniBand
IBA	IB Architecture
IBM	Formerly known as International Business Machines
ICC	Intel C Compiler
IDMA	Internal DMA

IDRIS	Institut du Développement et des Ressources en Informatique Scientifique (represented in PRACE by GENCI, France)
IEEE	Institute of Electrical and Electronic Engineers
IESP	International Exascale Project
I/O	Input/Output
IOR	Interleaved Or Random
IP	Internet Protocol
IPC	Inter-Process Communication
IPMI	Intelligent Platform Management Interface
ISC	International Supercomputing Conference; European equivalent to the US based SC0x conference. Held annually in Germany.
JBOD	Just a Bunch of Disks
JSC	Jülich Supercomputing Centre (FZJ, Germany)
kB	Kilo (= 2^{10} ~10^3) Bytes (= 8 bits), also KByte
KTH	Kungliga Tekniska Högskolan (represented in PRACE by SNIC, Sweden)
kW	Kilo Watt
LINPACK	Software library for Linear Algebra
LLNL	Laurence Livermore National Laboratory, Livermore, California (USA)
LMT	Lustre Monitoring Tool
LQCD	Lattice QCD
LRZ	Leibniz Supercomputing Centre (Garching, Germany)
MAC	Media Access Control
MAID	Massive Array of Idle Disks
MB	Mega (= 2^{20} ~ 10^6) Bytes (= 8 bits), also MByte
MB/s	Mega (= 10^6) Bytes (= 8 bits) per second, also MByte/s
MDT	MetaData Target
MFC	Memory Flow Controller
MF/s	Mega (= 10^6) Floating point operations (usually in 64-bit, i.e. DP) per second, also MFlops/s
MHz	Mega (= 10^6) Hertz, frequency =10^6 periods or clock cycles per second
MIPS	Originally Microprocessor without Interlocked Pipeline Stages; a RISC processor architecture developed by MIPS Technology
MKL	Math Kernel Library (Intel)
Mop/s	Mega (= 10^6) operations per second (usually integer or logic operations)
MPI	Message Passing Interface
MPICH	MPI CHameleon
MPP	Massively Parallel Processing (or Processor)
µSD	Micro SD
MVAPICH	MPICH over Verbs API
MW	Mega Watt
NAS	Network-Attached Storage
NCF	Netherlands Computing Facilities (Netherlands)
NCPUM	NumaConnect Pick-Up Module
NFS	Network File System
NI	National Instruments
NIC	Network Interface Controller
NSD	Network Shared Disk (IBM/GPFS)
NUMA	Non-Uniform Memory Access or Architecture
OpenCL	Open Computing Language
OpenMP	Open Multi-Processing

OpenMPI	Open MPI
OS	Operating System
OSS	Object Storage Server
OST	Object Storage Target
PCIe	Peripheral Component Interconnect express, also PCI-Express
PDU	Power Distribution Unit
POSIX	Portable OS Interface for Unix
PRACE	Partnership for Advanced Computing in Europe; Project Acronym
PSNC	Poznan Supercomputing and Networking Centre (Poland)
QCD	Quantum Chromodynamics
QDR	Quad Data Rate
QFED	OpenFabrics Enterprise Distribution
QPACE	QCD Parallel Computing on the Cell
RAID	Redundant Array of Inexpensive Disks
RAM	Random Access Memory
RDMA	Remote Data Memory Access
RI	Research Infrastructure
RISC	Reduce Instruction Set Computer
RNG	Random Number Generator
RPM	Revolution per Minute
SAN	Storage Area Network
SARA	Stichting Academisch Rekencentrum Amsterdam (Netherlands)
SAS	Serial Attached SCSI
SATA	Serial Advanced Technology Attachment (bus)
SD	Secure Digital
SDK	Software Development Kit
SFP	Small Form-factor Pluggable
SGEMM	Single precision General Matrix Multiply, subroutine in the BLAS
SGI	Silicon Graphics, Inc.
SIMD	Single Instruction Multiple Data
SLURM	Simple Linux Utility for Resource Management
SoC	System on Chip
SM	Streaming Multiprocessor, also Subnet Manager
SMP	Symmetric MultiProcessing
SNIC	Swedish National Infrastructure for Computing (Sweden)
SP	Single Precision, usually 32-bit floating point numbers
SRAM	Static RAM
SRIO	Serial Rapid I/O
SSD	Solid State Disk or Drive
SSU	Scalable Storage Unit (Xyratex)
STFC	Science and Technology Facilities Council (represented in PRACE by EPSRC, United Kingdom)
TB	Tera (= 240 ~ 1012) Bytes (= 8 bits), also TByte
TCO	Total Cost of Ownership. Includes the costs (personnel, power, cooling, maintenance, ...) in addition to the purchase cost of a system.
TCP	Transmission Control Protocol
TDP	Thermal Design Power
TF/s	Tera (= 1012) Floating-point operations (usually in 64-bit, i.e. DP) per second, also TFlops/s
TI	Texas Instruments
UiO	Universitetet i Oslo (University of Oslo, Norway)

UPC	Unified Parallel C
USB	Universal Serial Bus
UV	Ultra Violet (SGI)
UVA	Unified Virtual Addressing (NVIDIA)
VTK	Visualization Tool-Kit
VTL	Virtual Tape Library
W	Watt

Executive Summary

The objective of Work Package 9 task 3 is to assess and make recommendations to the PRACE RI for joint developments with industrial partners to develop highly energy efficient HPC components and systems, as well as power and cooling technologies. WP9 has carried out this task through evaluation of a number of prototypes targeting novel approaches to HPC server and system design with many prototypes having some degree of direct industry involvement or support.

Prototype efforts assessed the use of FPGAs for function acceleration, the use of CPUs for the mobile market and with a TDP about two orders of magnitude less than typical x86 CPUs for the HPC market, DSPs common for embedded systems and with a TDP about one order of magnitude less than x86 CPUs, the emerging heterogeneous CPUs integrating x86 and GPU cores, and traditional GPUs with a novel direct communication between GPUs via Infiniband between nodes. Two prototypes focused on novel approaches to scalability of I/O systems in support of Exascale systems and their energy efficiency. Technologies assessed included integration of I/O nodes into the MPP or cluster interconnect fabric, the use of flash technology, scalable disk systems and virtual tape libraries based on disk systems with spun down idle disks. Data management in file systems, in particular the management of large numbers of small files, was also addressed with the I/O-prototypes. One prototype evaluation assessed the issues and benefits of integrated cooling solutions for hot water cooling.

The findings of the evaluations of prototypes looking at HPC server architectures is that 1) an optimized FPGA implementation of matrix-multiplication can offer a 5 – 10 times higher energy efficiency than an x86 software solution, 2) an optimized implementation of matrix multiplication on the DSP can yield about half the energy efficiency gain of an FPGA implementation, 3) the first and second generation x86+GPU CPUs are not competitive in regards to energy efficiency even with standard x86 CPUs for the functions investigated, and 4) that good scalability can be achieved for clusters based on nodes using mobile CPUs though the floating-point capabilities of the current generation mobile CPUs is insufficient to be competitive in terms of energy efficiency.

Software optimization can make an order of magnitude or more difference in efficiency and energy efficiency for kernels frequently used for HPC benchmarking even for HPC established architectures, such as the x86. Further efforts are required to better understand the energy advantages of novel approaches to HPC server designs.

The evaluation of the I/O prototypes largely gave the expected outcomes, but did expose the dependence on good software implementations and the efficiency and scalability issues related to large file systems in which a large fraction of the files are small.

The integrated cooling with hot water prototype demonstrated that the technology is viable, but that further study is necessary to assess the economic benefits, and that those likely depend on the local situation. That could however change if technology would become available to convert heat into electricity in a cost competitive way.

1 Introduction

The main objective of PRACE is to assure the availability of Leadership Class systems and associated software and services for the Computation and Data Treatment needs of European researchers as outlined in the 2006 ESFRI report [ESFRI-2006] to strengthen European competitiveness for the benefit of the society. The particular aspect of this broad objective addressed by Work Package 9 Task 3 in the First PRACE Implementation Phase project (PRACE-1IP) is energy efficiency of systems and their operation.

Large computing systems have become large consumers of electricity for operations and cooling to the point that the electricity demands may be the limiting factor for the size of the systems that can be installed, dominate the life-time cost of systems and their operation, or what is socially and environmentally acceptable. For this reason, in particular in regards to reaching the next major performance goal of large scale systems, Exascale systems, i.e., systems with a peak performance of 10^{18} floating point operations per second and beyond, business as usual is generally not viewed as feasible. A business-as-usual approach has been estimated to lead to power requirements of about 200 MW or more for such a system. Innovations beyond the normal are necessary in every aspect: device technology, architecture, software, algorithms, cooling solutions, and operations. For cost and environmental reasons, solutions that enable energy recovery are also of great importance.

HPC systems solutions have over the last decade come to be dominated by designs based on processors using the x86 instruction set and CPUs targeted for a broad market. This target has enabled very cost competitive CPUs, but in part has been realized through a rich set of features, some of which are of little or no interest for HPC applications. CPUs designed for the mobile and embedded market have traditionally had power and energy consumption as design constraints since in many applications they are powered by batteries of very limited capability, or placed in space with limited power and cooling. For these reasons they have had a much more restricted feature set, and quite refined power management compared to server class CPUs. The power consumption of CPUs for the mobile and embedded markets has traditionally been in the tens of milli-watts to a few watts range whereas server class CPUs power range has been 50 – 200+ W. This huge difference in power demand is one reason why mobile and embedded CPUs have gathered strong interest as candidates for Exascale systems [DARPA-2008], in particular since recently 64-bit floating-point has become of interest also for the mobile and embedded markets.

How the difference in processor and memory system architecture, performance enhancing features such as out-of-order instruction execution and prefetching, reliability and availability features, and communication capabilities between traditional server class CPUs and mobile and embedded CPUs affect the energy efficiency of HPC applications is not yet well understood. Therefore, WP9, Task 3, has been undertaking a few prototype activities in an attempt to develop an understanding of the benefits and drawbacks for the type of HPC applications the PRACE RI supports and is expected to support in the future. Prototypes were constructed or acquired for assessment of the energy efficiency of mobile CPUs (ARM), embedded CPUs (DSP), integrated x86 and streaming cores (APU), "traditional" accelerated nodes of x86 CPUs with GPUs, and clusters built with mobile CPUs (ARM) with Gigabit Ethernet and GPU accelerated nodes with QDR Infiniband. The energy efficiency of function acceleration through the use of FPGAs, that in recent years have reached significant capabilities, was also investigated.

I/O systems are a very important part of HPC systems, in particular scalability to Exascale systems is a concern. One prototype was constructed to investigate the use of flash memory and direct integration of the I/O system in MMP and cluster interconnection fabrics for performance enhancement and reduced energy requirements. The use of MAID (Massive Arrays of Idle Disks) technology for back-up and archival storage was investigated in another prototype for which the management issues of large file systems dominated by small files also were considered.

Server power consumption has been increasing from typically less than 100 W a decade or so ago, to typically about 300 W today and close to 1 kW for GPU accelerated dual socket nodes. Packaging and cooling technologies have also evolved so that the power consumption of racks for HPC servers has increased from a few kWs to typically 30 – 50 kW today and well over 100 kW for the most power dense designs. With the increased power and heat density the use of air as cooling medium may not be the most economical, or in case of the higher densities not even technically feasible. Some form of liquid cooling may be preferred or necessary. Integrated liquid cooling presents several technical challenges, but also potentially large benefits in enabling coolant outlet temperatures at about 60 ^{o}C that in many situations make energy recovery more efficient without adversely affecting component operating temperatures and life time. One of the prototypes was constructed to assess the benefits and challenges from a holistic approach to the use of direct warm/hot water cooling.

The outline of this report is as follows:

Chapter 2 provides a background on both historic trends with regard to computer performance and energy consumption as well as an overview over recent developments in computer technologies and their implications on energy efficiency.

Chapter 3 describes the prototypes used by WP9 to evaluate future HPC technologies and trends. Detailed descriptions of the hardware, software and system level aspects of the prototypes are included.

Chapter 4 provides details on the instrumentation used by the different prototypes to measure power consumption, including information on what parts of the prototype were instrumented and expected measurement accuracy.

Chapter 5 introduces the benchmark workloads used to carry out the energy efficiency and performance assessments on the prototypes, including key characteristics of the benchmarks important for the evaluation.

Chapter 6 presents results of the energy efficiency and performance measurements of the presented benchmark workloads on the prototypes. More in-depth results as well as background information on the processing of the actual measured data are available in the annex (Chapter 8).

Chapter 7 gives the conclusions drawn from the evaluation and recommendations for future prototyping activities.

2 Technology Background

Large computing systems have become large consumers of electricity for operations and cooling to the point that the electricity demands may be the limiting factor for the size of the systems that can be installed, or what is socially and environmentally acceptable. The latter issue is raised in, e.g., [Mills-2008, Daily-2009, Inefficient-2007, Wyoming-2010, Wyoming-2012, NYT-2012a, NYT-2012b].

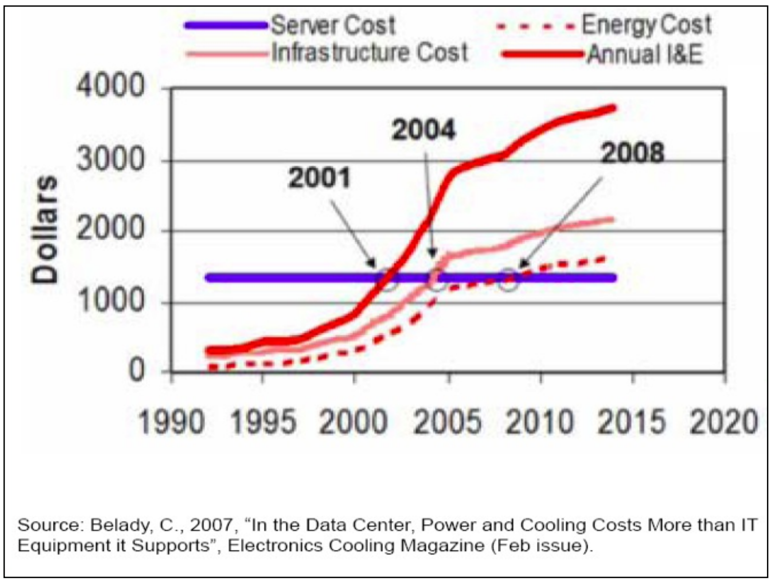


Figure 1 Evolution of US power and cooling costs for standard x86 servers.

The increased electric energy demand has also resulted in rapidly increased cost for power and cooling to the point that, since a few years back, in many locations energy costs during the life-time of a computer system exceeds the computer system capital cost [Belady-2007], Figure 1, and today may in fact amount to almost twice the capital cost of the system. The reason for this development is in part that server costs have remained fairly constant, or even declined slightly [Koomey-2009a], while the cost of electricity has been rising. Electric energy costs in Europe vary greatly with country and so do historic cost trends, but for the larger economies of Germany, France, Italy, Spain and the United Kingdom the cost on average has risen about 57% from 2001 to 2012 according to Eurostat [Eurostat-2013], or on average about 4.3% per year. With the worldwide increasing needs for energy, in particular in developing countries, and the drive towards clean energy sources the electric energy cost is generally expected to rise faster in the future rather than slower.

Though semiconductor technology has realized unparalleled exponential gains in capabilities of integrated circuits leading to a reduction in switching energy of transistors by more than a factor of a million over a 30-year period, Figure 2, HPC system energy requirements have been increasing. The reduction in transistor power needs have resulted in a doubling of the energy efficiency of servers about every 18.8 months, Figure 3 [Koomey-2009b] in line with "Moore's Law" [Moore-1965]. However, the power demand of a standard server has increased over time despite this remarkable improvement in transistor power needs from less than 100 W on average to about 200 – 300 W during the last 10 – 15 years [Koomey-2009a]. For large HPC clusters, as represented by systems on the Top500 list based on the system performance for the High Performance Linpack (HPL) benchmark [Linpack], the capabilities have increased at a rate of doubling about every 13 months, Figure 4, resulting in a growth rate of the energy needs of large data centers of about 20% per year [Brill-2008], which exceeds the predictions of 14 – 17% per year in the EPA report to Congress [EPA-2007].

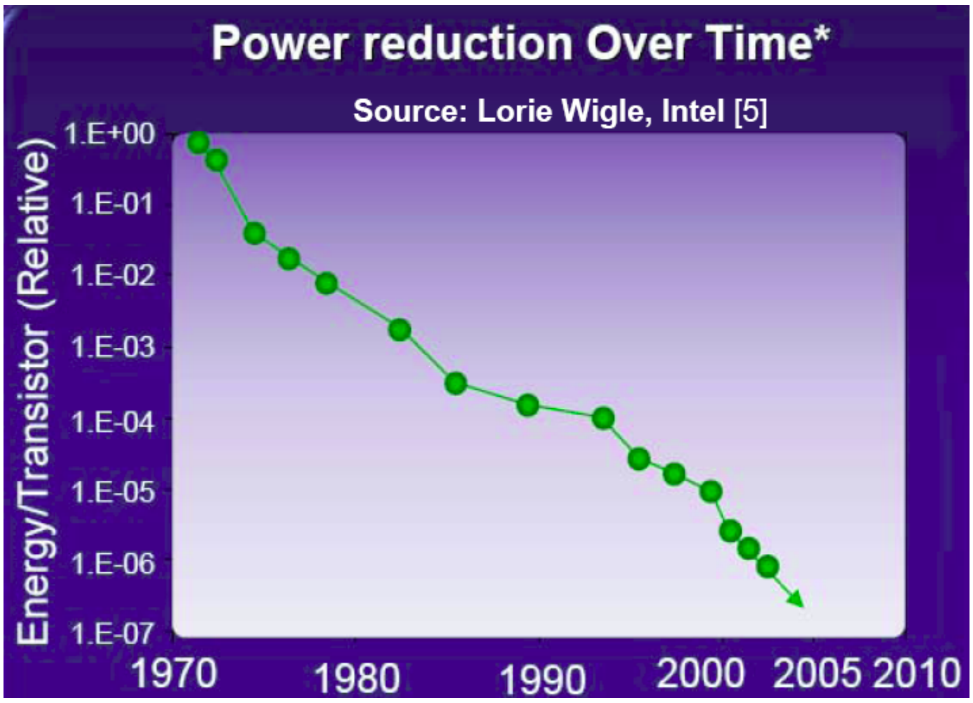


Figure 2 Transistor power reduction over time. [Curley]

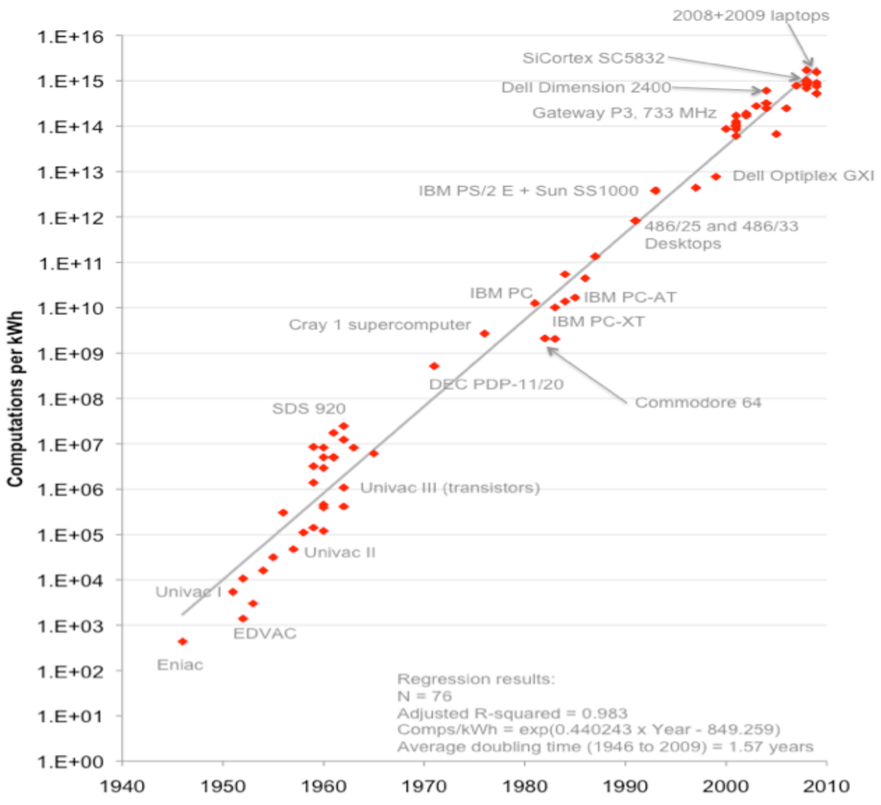


Figure 3 Energy efficiency evolution of PCs. [Koomey-2009b]

The improvement in energy efficiency for systems on the Top500 list is exemplified in Table 1, showing the most energy efficient systems of a few types on the first Green500 list [Green500-2007] created in 2007 and the most recent list [Green500-2012]. It is interesting to note that on the 2007 list there was no system that used GPUs for acceleration and that the PowerPC-based Blue Gene systems designed for energy efficiency and scalability topped the list. In 2012, the most energy efficient system for the HPL benchmark used for the Top500 and the Green500 ranking was an x86 based system using Xeon Phi's for acceleration. Over the five years from 2007 to 2012 the improvement in energy efficiency of the Blue Gene design (by a factor of 5.9) and that of x86 based systems (by a factor of 6.5) are, though quite impressive, in line with the Uptime institute observations [Brill-2008].

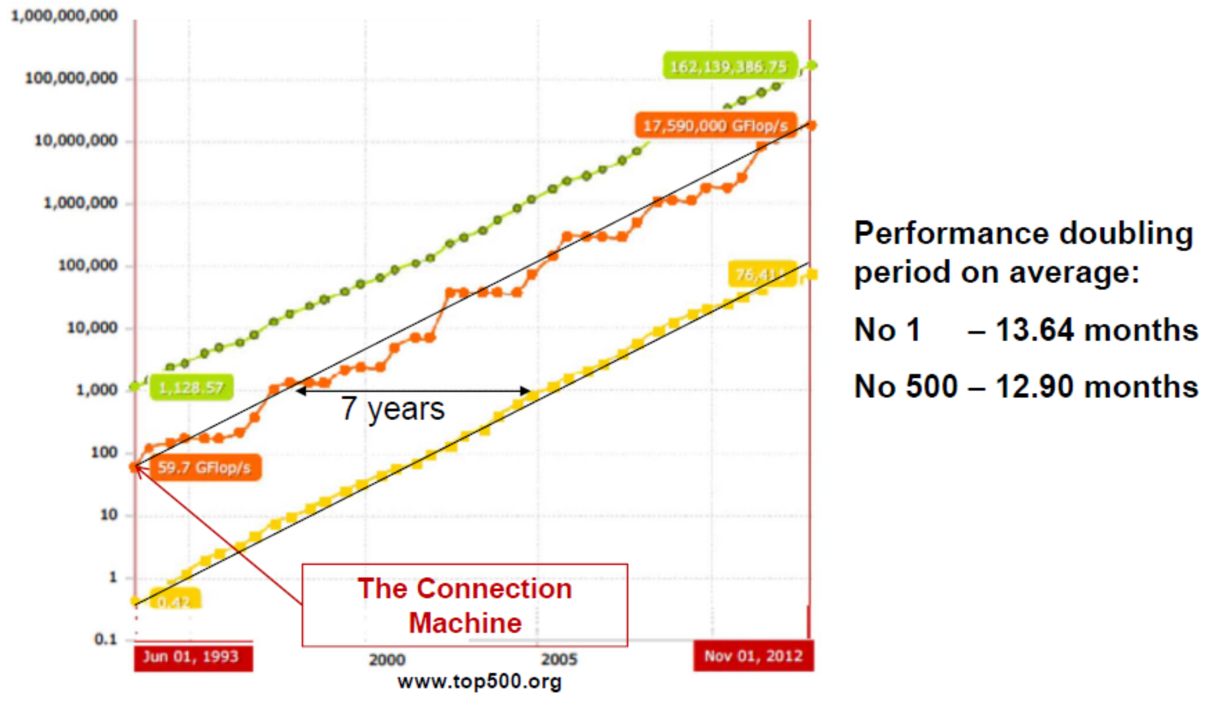


Figure 4 Performance evolution of large HPC systems as measured by the Linpack (HPL) benchmark.

	November 2007	November 2012	Ratio 2012/2007
No 1 Green 500	357	2499	7.0
Best non-accelerated (Blue Gene)	357	2102	5.9
Best x 86 non-accelerated	155	1010	6.5
Best x86 with Accelerator		2499	

Table 1 Energy efficiency for Top500 systems in MF/J for the HPL benchmark.

Despite the exceptional reduction in transistor power demands as depicted in Figure 2, power demand at the processor level grew, in part due to the inclusion of various features aimed at enhancing the performance for many applications, such as out-of-order execution and speculative execution. Figure 5 and Table 2, in which the impact of different process technologies has been factored out [Intel-2006], show how the energy per instruction increased by about a factor of five from the i486 to the Pentium-4. The ambition by the industry to have application performance grow at a rate comparable to the increased transistor density following Moore's Law resulted in an increased energy per instruction and increased heat density, as illustrated in Figure 6. This scaling behavior caused Fred Pollack of Intel to make the "famous" remark "We are on the wrong side of a square law!" at the 32nd Annual IEEE/ACM International Symposium on Microarchitecture, 1999 [Pollack-1999] and propose a new goal "Double valued performance every 18 months, at the same power level".

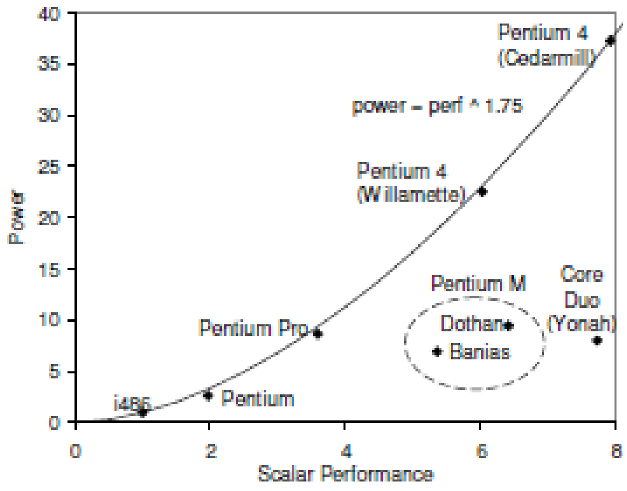


Figure 5 Normalized power versus normalized scalar performance for multiple generations of Intel microprocessors. [Intel-2006]

Product	Normalized Performance	Normalized Power	EPI on 65 nm at 1.33 volts (nJ)
i486	1.0	1.0	10
Pentium	2.0	2.7	14
Pentium Pro	3.6	9	24
Pentium 4 (Willamette)	6.0	23	38
Pentium 4 (Cedarmill)	7.9	38	48
Pentium M (Dothan)	5.4	7	15
Core Duo (Yonah)	7.7	8	11

Table 2 Energy per instruction (EPI) of Intel microprocessors. [Intel-2006]

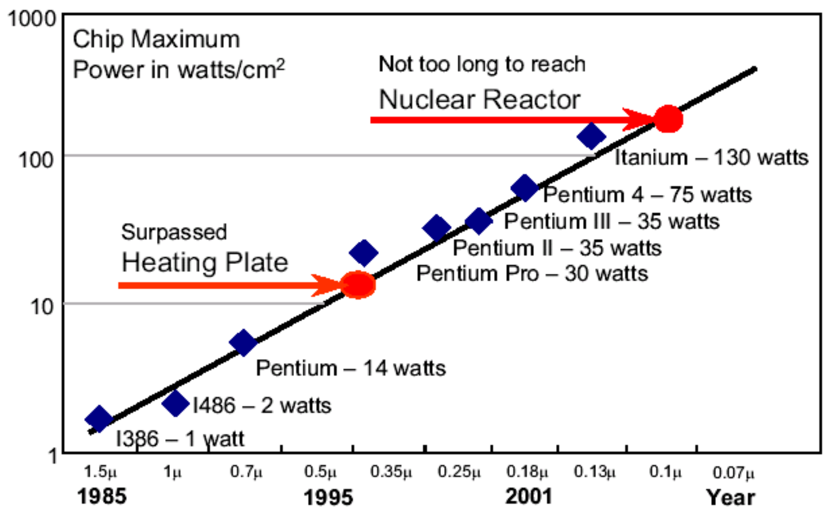


Figure 6 Heat density of Intel microprocessors. Source: Shenkhar Borkar, Intel

The square law Pollack was referring to is the relationship between power (P), voltage (V) and frequency (f) of CMOS: $P = c_1 V^2 f + c_2 V + c_3 + O(V^4)$, where the first term represent dynamic (switching) power, the second the leakage and board power and the third term represent fan power. The formula is illustrated for the HPL [Linpack] benchmark in Figure 7. Furthermore, voltage and frequency are also related as shown in Figure 8.

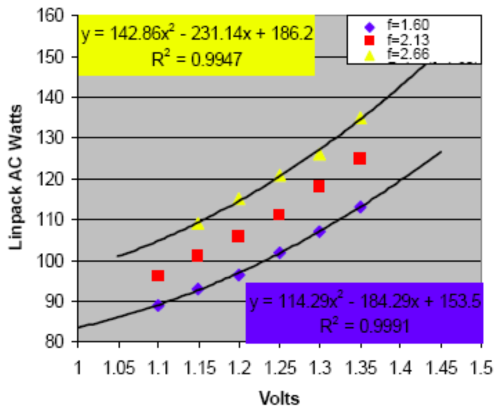


Figure 7 Power requirements as a function of voltage and frequency. [Supermicro-2008]

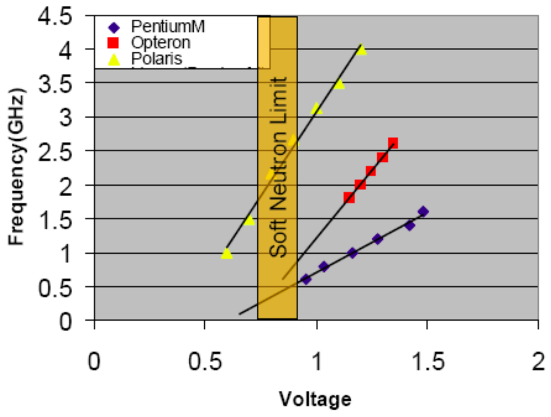


Figure 8 Relationship between frequency and voltage. [Supermicro-2008]

The heat density issues have made the industry largely keep the Thermal Design Power (TDP) constant the last several years thereby limiting the clock frequency to be fairly constant, as seen in Figure 9, Figure 10 and Figure 11.

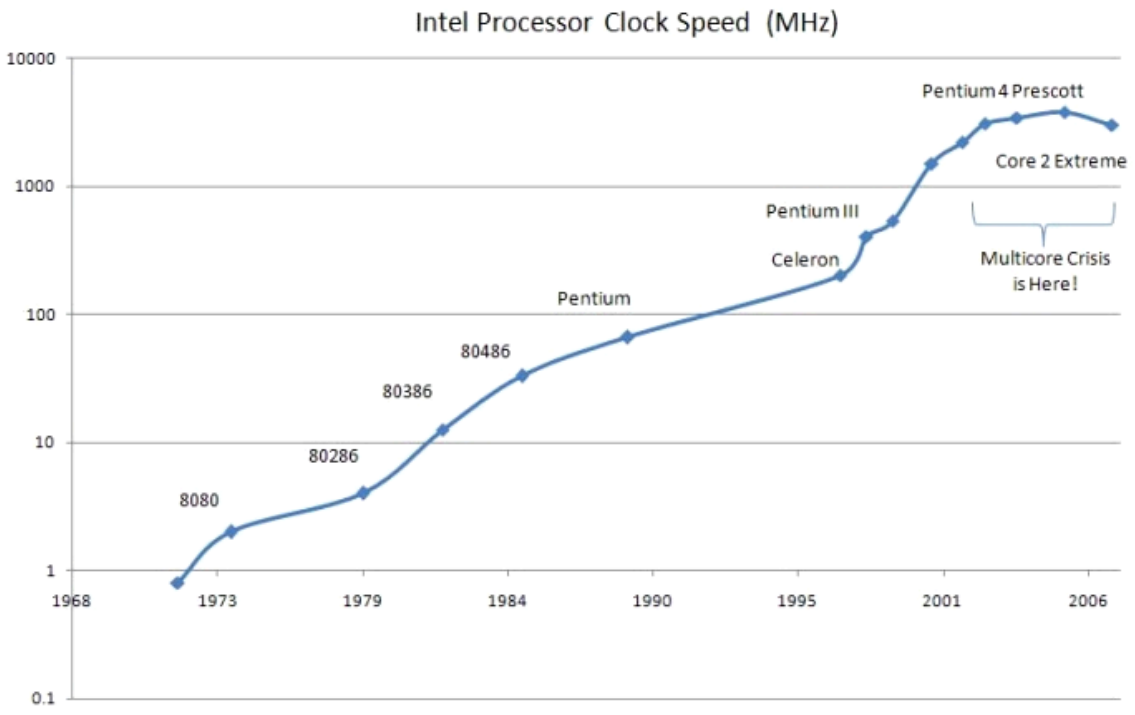


Figure 9 Historic clock frequency of Intel CPUs (log scale). [Intel-clock]

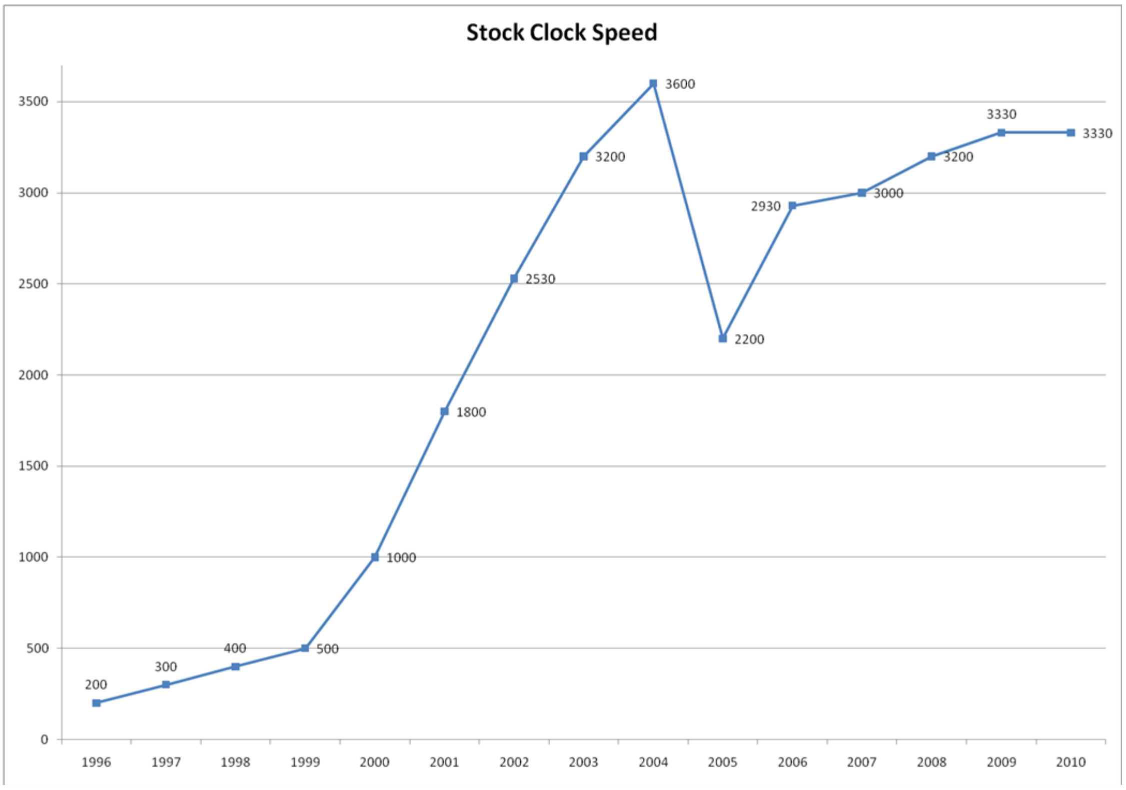


Figure 10 Clock frequency of recent Intel CPUs without turbo boost. [ung-2010]

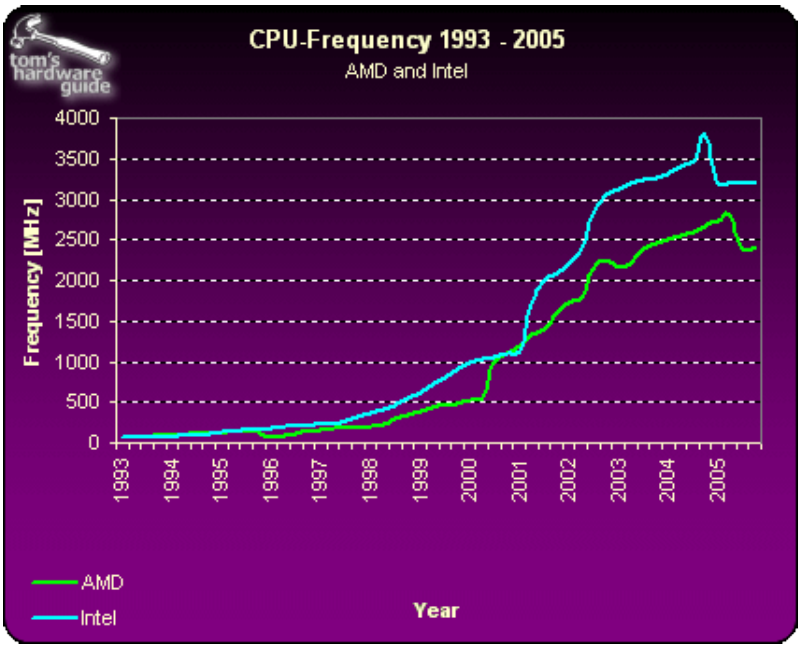
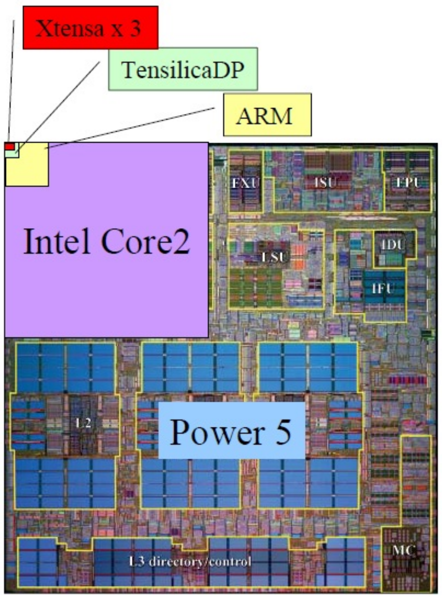


Figure 11 Clock frequencies of AMD and Intel CPUs. [AMD-Intel]

The industry response to seek to achieve continued exponential increase in capabilities at constant power has been multi-core CPUs with IBM introducing a dual core Power4 CPU in 2001 and AMD a dual core x86 CPU in 2004 and Intel a dual core x86 CPU in 2005. Since then the number of cores has increased with, e.g., IBMs Power7 having 8 cores, its Blue Gene/Q having 18 cores, both using 45 nm technology, AMDs Interlagos having 16 cores and Intel's Sandy Bridge having up to 10 cores, both using 32 nm technology. Graphics Processing Units (GPUs), having less complex and smaller processing cores, have a significantly larger number of cores with AMD's FirePro V7900 having 1280 processing cores in 40 nm technology (FirePro S9000, the next generation AMD GPU has 1792 processing cores in 28 nm technology) and NVIDIA's Fermi having 512 processing cores also using a 40 nm process (Kepler K20x due out this year in 28 nm technology has 2688 processing cores). Figure 12 illustrates the significant difference in size and power of different processor designs [Shalf-2007]. New approaches to processor designs may indeed be required to manage power issues. [Horowitz-2009, Dally-2008, Intel-2012]



- Power5 (server)
  - 389mm^2
  - 120W@1900MHz
- Intel Core2 sc (laptop)
  - 130mm^2
  - 15W@1000MHz
- ARM Cortex A8 (toaster oven)
  - 5mm^2
  - 0.8W@800MHz
- Tensilica DP (cell phones)
  - 0.8mm^2
  - 0.09W@600MHz
- Tensilica Xtensa (Cisco Rtr)
  - 0.32mm^2 for 3!
  - 0.05W@600MHz

Figure 12 Illustration of the large difference in size and power consumption of CPUs about 2005. [Shalf-2007]

The architecture and size of a processing core has a significant impact on the energy consumption, and its capabilities. An x86 core for performance oriented CPUs typically may consume 5 – 10 W in 32 nm technology, while a GPU core may consume about 0.1 – 1 W in 40 nm technology, a difference by up to one order of magnitude. However, the capabilities of these two types of cores as measured by double precision floating-point operations is quite large as well. The theoretical energy efficiency for an x86 AMD Interlagos CPU is about 1.3 GF/J and that of the FirePro V7900 3 GF/J and Fermi 2.3 GF/J (AMD S9000 3.6 GF/J, Kepler 20X 5.5 GF/J). Thus, the 40 nm GPUs have a nominal energy efficiency about 2 – 3 times that of an x86 CPU and the 28 nm GPUs a nominal energy efficiency for double precision up to four times that of a 32 nm x86 CPU.

But, there are also other types of cores in existence representing other trade-offs. Figure 5 included three data points that did not fall on the power-performance curve. Those are Intel mobile CPUs which are optimized for energy consumption and have simpler cores (for instance, in regards to out-of-order execution, speculative execution and pipeline depths). ARM has had a dominating position in the mobile market with their CPUs typically requiring less than 1 W and performance-optimized CPUs requiring up to 2 W, as opposed to x86 performance CPUs requiring 130 – 150 W and GPUs about 250 W. But, until recently the floating-point performance of ARM CPUs, in particular double-precision floating-point performance, has been quite poor, in fact so poor that despite the lower power requirements the energy efficiency for double precision floating-point has not been competitive. However, ARM has announced a 64-bit performance CPU, Cortex A57, due out this year with a TDP comparable to ARMs previous generations of performance CPUs. A number of server designs targeting non-floating-point intensive workloads have been introduced since 2011, for example by Calxeda (ARM) [Calxeda], Marvell (ARM) [Marvell] and SeaMicro (Intel ATOM) [SeaMicro], with designs targeting HPC applications expected to become available in 2014.

Processors targeting the embedded market have traditionally also been designed with much more stringent limits on power demands than x86 designs with a TDP of about 10 W and considerably better floating-point performance than mobile CPUs, like previous generations ARM. For instance, the Texas Instruments TMS320C6678 DSP has a nominal double-precision floating-point performance of 4 GF/J. Thus, in regards to nominal double-precision floating-point energy efficiency the 40 nm TI DSP is somewhat more efficient than the 45 nm BG/Q CPU, 30 – 80% more efficient than 40 nm GPUs, and 3 times more energy efficient than 32 nm x86 CPUs. The memory hierarchy and flexibility of the DSP is closer to that of an x86 CPUs than that of a GPU. The design principles and architectures of mobile CPUs and DSPs have been viewed as potential good starting points for the design of CPUs for Exascale systems [DARPA-2008].

Finally, FPGAs have grown significantly in capabilities and offer a highly flexible medium for the implementation of processing logic, e.g., allowing for the implementation of very energy efficient floating-point engines, with a delivered energy efficiency potentially better than that of DSPs. Given that for FPGAs many implementations of floating-point arithmetic are possible there isn't any nominal peak performance number readily available. Our estimate for the Xilinx Vertex-6 is that a nominal double precision floating-point performance in the 5 – 10 GF/J range should be achievable based on our prototype results and [Jovanovic-2012]. The Xilinx Vertex-7 produced in 28 nm technology (not available in time for this WP9 initiative) is expected to realize an energy-efficiency of 10+ GF/J for matrix-multiplication [Xilinx-2012], or 2 – 3 times better than that of 28 nm GPUs.

With good floating-point capabilities available in several types of designs and the significant difference in size and power consumption the merits of a relatively complex instruction set, such as the x86 instruction set, and many of the features in modern x86 cores, have been investigated [Shalf-2010]. The conclusion was that the majority of the Livermore application codes only needed 80 out of the 300+ instructions in the x86 instruction set. This conclusion was the basis for the Tensilica based Green Flash design [GF-2008]. Furthermore, some of the RAS (Reliability, Availability, Serviceability) features common in x86 CPUs are not viewed as essential in many HPC scenarios, and are not present in mobile or embedded CPUs giving them an inherent energy efficiency advantage.

With the success of streaming architectures (GPUs) for many applications and the increased capabilities in the form of billions of transistors on a single die, heterogeneous CPUs integrating streaming (GPU) and x86 cores on a single die for the PC and server market have appeared, such as the AMD Fusion (Brazos, Llano and Trinity) Application Processing Units (APUs) and Intel's Sandy and Ivy Bridge. For the mobile market CPUs with ARM cores and streaming processors have also appeared as feature sizes have reached about 30 nm or less, such as NVIDIA's Tegra2 and Tegra3. Similarly, dies integrating ARM and DSP cores have been announced, such as Texas Instruments 66AK2H12 with 4 ARM Cortex A15 and 8 C66x cores in 28 nm technology [TI-2012]. The new heterogeneous CPUs enable the streaming or DSP cores to share memory with the x86 or ARM cores without the involvement of an I/O bus, thus avoiding a potential bottleneck. However, for the integrated CPUs targeting the PC and server market the memory bandwidth is in line with that of typical x86 CPUs and a factor of five or more less than that of dedicated GPUs with their own DDR memory.

Table 3 shows the nominal double precision floating-point performance of CPUs that either are used for HPC servers or are of potential interest for HPC servers with processors available for assessments in WP9 prototypes marked in boldface. The table also include some "next generation" CPUs that in some cases have become available now.

The server CPU industry has also adopted many features introduced earlier in the mobile market, such as clock and power gating to reduce power consumption of logic that is either not active, or that can operate in a lower performance state (lower frequency) without (too much) adverse impact on performance, or that need to be operated in a lower performance state for the chip not to overheat. As technology has been developed to do on-chip voltage control the power control of logic has been increasingly refined [IBM-2010, Intel-2011, AMD-2011] with, e.g., Intel's Sandy Bridge having 100 microarchitecture event counters for power control. The number of temperature sensors has increased as well with e.g. IBM's Power 7 having 44 temperature sensors used in refined power control. The dynamic power management has so far mostly been assigned to the OS, but with the Sandy Bridge Intel has started to expose some of these features to users/applications. Some studies have been carried out to assess the energy-performance tradeoffs for HPC applications, see e.g. [Feng-2005, IBM-2011] showing measurable benefits. Some of the prototype efforts also undertook an assessment of performance vs. energy efficiency trade-offs.

CPU	**"Current"**			**"Next Generation"**		
	Feature size	**GF/J**	**GF/mm2**	**Feature size**	**GF/J**	**GF/mm2**
AMD Interlagos (16C, 2.7 GHz)	32	1.3	0.55			
AMD FirePro 7900	40	3.0	1.19			
AMD S9000				28	3.6	2.21
AMD Brazos	**40**					
AMD Llano	**32**					
AMD Trinity	**32**	**1.0**	**0.40**			
IBM Blue Gene/Q	45	3.7	0.57			
IBM Power7	45	1.4	0.48			
Intel Sandy Bridge (8C, 3.1 GHz)	**32**	**1.3**	**0.46**			
Intel Ivy Bridge (4C,3.5 GHz)				**22**	**1.45**	**0.70**
Nvidia Fermi	**40**	**2.66**	**1.26**			
Nvidia Kepler 20x				28	5.2	2.36
Nvidia Tegra2	**40**	**~1**	**0.04**			
Nvidia Tegra3	40					
TI TMS320C6678	**40**	**4**	**~3**			
TI 66AK2Hx				28		
Xilinx Vertex-6	**40**	**5-10**				
Xilinx Vertex-7				28	~13	0.236

Table 3 Summary of the nominal double precision floating-point performance.

High performance scalable I/O and storage systems for Leadership Class systems do represent a significant challenge that most likely will require both system architectural changes as well as introduction of storage technologies that are not common in today's HPC systems. For instance, Solid State Disks (SSDs) can be introduced in several different ways to potentially enhance performance, and a tighter integration of the I/O and storage systems into a HPC cluster or MPP can potentially significantly enhance performance. For archival or near-line storage, disk systems in which disks are spun down when not in active read/write mode could enhance performance compared to tape libraries without a significant penalty in energy consumption. Two prototype efforts were undertaken to assess benefits and challenges in using these technologies and their expected scalability challenges.

Most of the energy required by computing systems turns into heat that must be removed through various technologies. In the early days of supercomputing exemplified by the Cray-1 liquid cooling was used with components submerged in cooling liquid. With the emergence of the microprocessor as the dominating processor heat densities became sufficiently low that air cooling became dominant by far, with heat sinks, fans and Computer Room Air Conditioning (CRAC) being the norm. However, as the power requirements of microprocessors have increased and packing technologies have evolved heat densities of HPC systems have increased significantly and in many cases to the point that liquid cooling is a necessity. The heat capacity of air is simply not sufficient for proper cooling. Liquid cooling offers opportunities for energy recovery that are not practical for traditional air cooling with CRAC units. In the latter case the outlet water temperatures are much too low for an effective use of the energy in the outlet water for other purposes. With liquid outlet temperatures of at least 50 ^{o}C the opportunities for effective use of the energy in the cooling liquid increases dramatically [IBM-2009], as seen in Figure 13. Direct liquid cooling of components enables outlet temperatures in the desired range without adverse effects on component operation or lifetime. One prototype was constructed to assess the issues with warm/hot water cooling and the integration of such cooling into a campus system for heating and air conditioning.

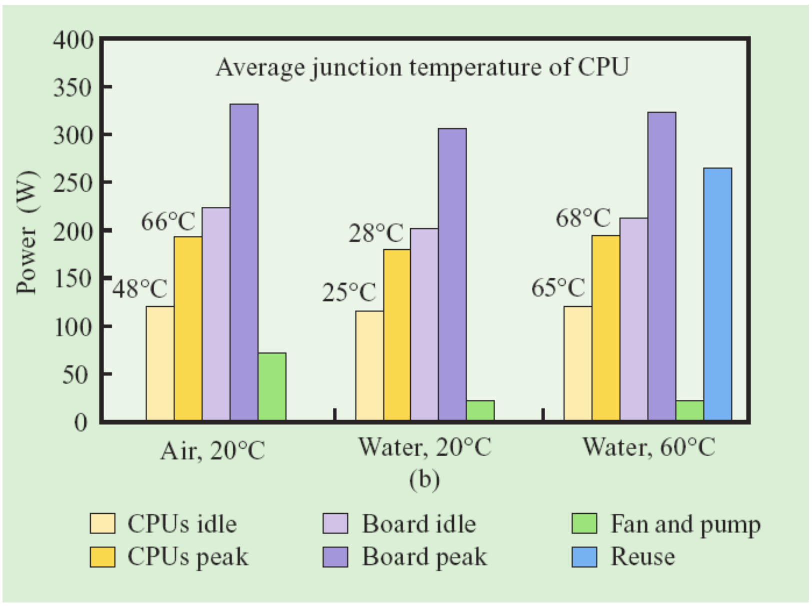


Figure 13 Hot water cooling and heat reuse [IBM-2009].

3 Prototypes

The objective of the PRACE 1IP Future Technologies WP9 prototypes is to evaluate and influence the development of new emerging technologies for future leadership and Tier-0 class systems. The focus of Task 9.3 is assessment of technologies for highly-energy-efficient HPC components and systems including integrated cooling and energy recovery technologies and, based on assessment outcomes, recommendations to the RI for joint developments with industrial partners towards future highly energy efficient multi-Petascale system architectures and technologies for highly energy efficient supercomputing centre design and operation. To achieve this objective the WP9 prototypes were intended to explore a diverse set of "high-risk" technologies (within the very limited budget with this ambitious goal) that are likely to have a big impact on leadership class HPC systems and associated infrastructure in the next 3-10 years. It was anticipated that by targeting "high-risk" technologies for assessment, prototypes might be (heavily) delayed, possibly to such an extent that prototypes might not be available within the project timeline due to major changes in vendor roadmaps, un-foreseeable technical problems, etc. This did indeed happen and the WP9 has been extended.

The WP9 prototypes cover three areas: compute node architecture and integration into clusters, I/O system architecture, and integrated cooling and energy recovery. Energy efficiency was assessed for all prototypes. For node architectures, three prototypes assessed the use of very low power HPC node designs based on architectures common in mobile and embedded markets and one prototype explored the use of FPGAs for acceleration of floating-point intensive compute tasks. Of the other two node architecture prototypes, one assessed the energy aspects of communication technologies and the other the scalability achievable through the use of a novel add-on node board for the creation of large shared memory systems based on common high-volume servers with a limited number of sockets. Two partners assessed I/O system scalability by integrating storage into the interconnect fabric of a cluster or MPP. One of the prototypes also explored the use of flash memory technology for energy efficiency as well as performance and another the use of MAID technology and the management of file systems with a large fraction of (very) small files. Finally, one prototype assessed integrated cooling and energy recovery technologies.

Device	Technology	Frequency (GHz)	Peak DP (GF/s)	Memory Technology	Channels per node	Memory Bandwidth (GB/s)
AMD Magny-Cours	45 nm	2.0	128	DDR3-1333	8	85.313
AMD Brazos E-350	40 nm	1.6	3.2/-	DDR3-1066	1	8.528
AMD Llano A8-3870	32 nm	2.4	38/-	DDR3-1333	1	10.664
AMD Trinity A10-5800K	32 nm	2.8		DDR3-1600		
Intel Sandy Bridge	32 nm	3.4-3.8	109-121	DDR3-1333	2	21.328
Intel Ivy Bridge	22 nm	3.5-3.9	111-125	DDR3-1600		
2xWestmere (6C) + 2xNvidia M2070	32/ 40 nm	2.67/ 1.15	128/ 1030	DDR3-1333/ GDDR5	6/ 6	63.984/ 150.000
Tegra2	40 nm	1	2	DDR2-667	1	5.533
TI TMS320C6678	40 nm	1.25	40	DDR3-1333[1]	1	10.664
Xilinx Vertex-6	40 nm	0.4	51	DDR3-1066	1	8.528

Table 4 Prototype processor and node characteristics.

Since energy efficiency tends to improve with reduced feature size of the technology being

[1] The EVM uses DDR3-1600 but due to limitations they operate at 1333 MHz

used, we summarize the technology and node characteristics of the different devices being assessed in Table 4.

The description below of the prototypes is organized according to targeted components or subsystems. Within each targeted area, descriptions are ordered alphabetically by responsible partner.

3.1 Compute Node Architecture Prototypes

3.1.1 *An NVIDIA Tegra 2 mobile SoC based HPC cluster, BSC*

Hardware

The prototype consists of two racks with 128 compute nodes in each rack. Each node consists of an NVIDIA Tegra 2 System on Chip (SoC), containing two ARM Cortex A9 CPUs, 1 GB of DDR2-667 DRAM memory and one 1 GbE embedded controller. The SoC, shown in Figure 14, also contains a few other units of limited or no interest for HPC applications. The ARM cores on the Tegra 2 SoC are each capable of 1 FMA every 2 cycles in double precision, or 1 GF/s/core at 1 GHz, for a total of 2 GF/s for the SoC. The SoC with associated compute node circuitry is mounted on a Q7-compliant carrier board as shown in Figure 15. Eight Q7 boards are mounted in a 1U container as shown in Figure 16

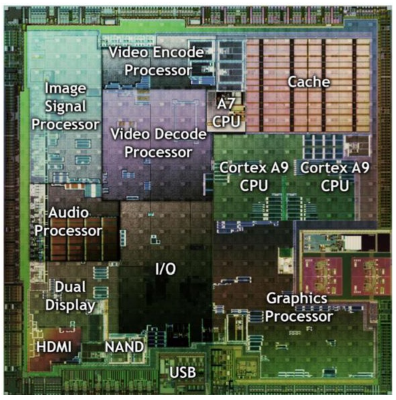


Figure 14 NVIDIA Tegra 2 System on Chip (SoC).

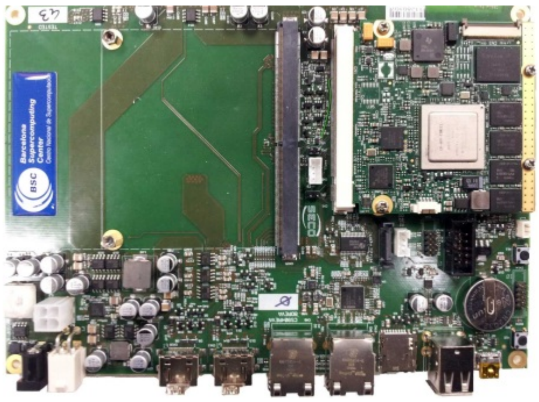


Figure 15 Q3 compute node card.

A compute node consumes about 8 W out of which the ARM cores account for 8% and the DDR2 memory for about 9%. The eight-node 1U container unit consumes about 70 W. Without any forced cooling the containers were mounted with 1U separation in a standard 19" rack to avoid overheating, resulting in a total of 128 nodes in a rack, Figure 17. GbE switches for node interconnect and service network were mounted in the space not used for compute containers. The node interconnect topology is tree-like with a 1 GbE node connection to a Cisco SG200-50 switch. The SG200-50 switches have 48 10/100/1000 TP (Twisted Pair) ports as well as two SFP (Small Form-factor Pluggable) ports and consume about 100 W. The interconnection network has a maximum distance between two nodes of four network hops, and a bisection bandwidth of 8 Gbps. The nodes are also attached to a 100 Mbps Ethernet service network (for filesystem and boot). Figure 18 shows a schematic of the interconnection network.

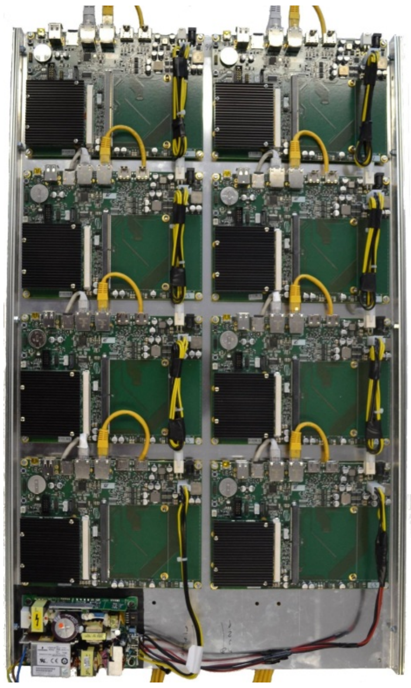


Figure 16 1U container of 8 compute cards.

Figure 17 19" rack with 16 compute containers.

The dual-core ARM Cortex A9 implements the ARMv7 instruction set with double-precision floating-point units and full cache coherency. Each ARM core has a 32 kB 4-way data cache and a 32 kB 4-way instruction cache. The two ARM cores share a 1 MB L2 cache. The power management functions are handled by a dedicated ARM7 processor, which allows the Tegra 2 to only turn on those processors that are required. The Tegra 2 also has dynamic voltage and frequency scaling capabilities. The Tegra 2 is produced using 40 nm technology. The GPU is not programmable and not used in any of the benchmarks used in our assessments.

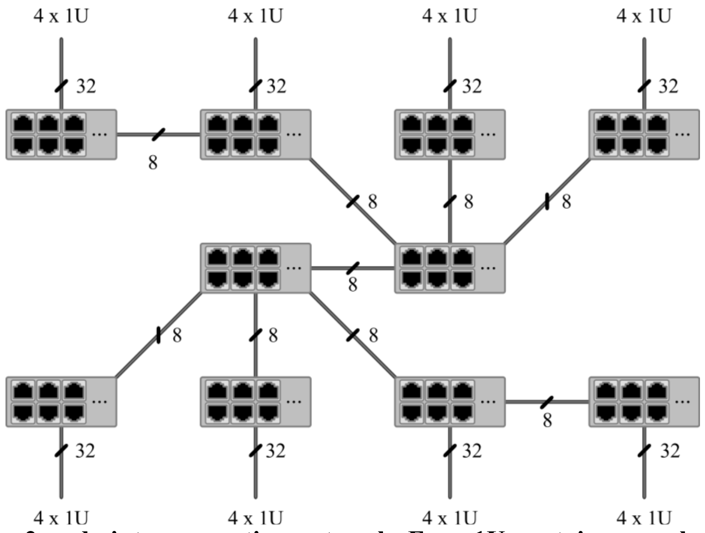


Figure 18 The Tegra 2 node interconnection network. Four 1U containers each with 8 compute nodes are connected to a 48-port leaf level switch.

Software

The Linux Kernel version 2.6.32.2 and a single Ubuntu 10.10 filesystem image are hosted on an NFSv4 server with 1 Gbps of bandwidth (each node connects to this server using the 100 Mbps service links). Each node has its own local scratch storage on a 16 GB μSD memory card. The prototype relies on version v1.4.1 of the MPICH2 MPI library, and on the SLURM job manager to manage the execution of MPI applications. For the algebraic backend we use the ATLAS [ATLAS] 3.9.51 library. This library was chosen due to the lack of a hand-optimized algebra library for the Tegra 2. Applications that need an FFT library rely on FFTW v3.3.1.

3.1.2 *GPU-GPU communication over PCIe and IB, CaSToRC*

This prototype was used for assessing the energy efficiency and performance impact of direct GPU-GPU communication over PCIe and Quad Data Rate (QDR) Infiniband connections. But, the benchmarks also reveal some aspects of the node performance and energy efficiency.

Hardware

The prototype comprises eight IBM iDataPlex servers each equipped with two NVIDIA Tesla M2070 Fermi based cards, Figure 19. The M2070 GPUs feature 150 GB/s local memory bandwidth, 6 GB of GDDR5 memory per card and a theoretical peak of 515 GF/s in double-precision. The cards use PCIe Gen 2.0 ×16 for interfacing to the host processor. Each server has six 4 GB DDR3-1333 DIMMs (one DIMM per memory channel) two 6-core Intel Xeon X5650 (Westmere) 2.67 GHz CPUs with 12 MB L3 cache, and one PCIe Gen 2.0 ×16 bus. The eight servers are interconnected to a QDR Infiniband switch using 4× QDR Mellanox MT26428 Host Control Adaptors (HCA). Each GPU is on a separate PCIe bus, and the HCA is mounted on a mezzanine card. The node configuration is shown in Figure 20. The Intel Westmere is produced in 32 nm technology and the NVIDIA Fermi in 40 nm technology.

Figure 19 IBM iDataplex dx360M3 server board and 2U chassis with two GPUs.

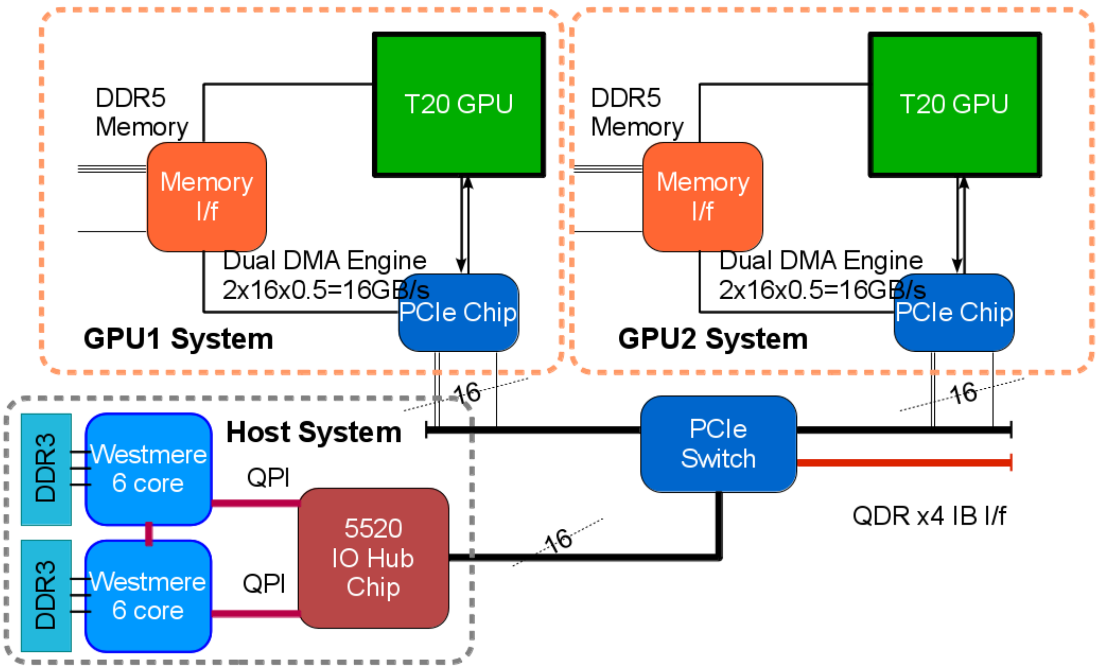


Figure 20 Node configuration for the x86+GPU prototype.

Software

The software configuration used for the benchmarks included the CUDA 4.2 toolkit (with NVIDIA driver version 285.xx.xx.xx), MVAPICH2 ver. 1.8a2 and OpenMPI ver. 1.5.4 for MPI. The MVAPICH2 library version 1.8a2 incorporates optimized support for GPU-to-GPU communications via the standard MPI interface. In particular, it includes support for point-to-point and collective operations, pipelined data transfer with automatically provided optimizations, GPUdirect and CUDA IPC. The available OpenMPI library did not support GPUdirect communication.

In the tests we used MPI, with two MPI tasks per node spawning off the CUDA kernels, such that each MPI task uses one GPU card.

For the STREAM benchmark, the benchmarks were set up as follows:

- For the GPU-local measurements, a GPU kernel program is run after the data is copied to the card so that the measurements reflects the memory bandwidth within the GPU.
- For GPU-to-GPU on the same node, peer-to-peer communication by NVIDIA was used, where data is transferred over the PCIe bus from one card to the other without going

through host memory. This transfer is initiated by the CPU, i.e. the call to the NVIDIA API is done by the CPU.
- For GPU-to-remote-GPU, communication makes use of the host memory and the CPU initiates all transfers. The CPU initiates a copy from GPU memory to host memory via an NVIDIA API call, then starts an MPI transfer from one node to another and then the CPU of the receiving node copies to the GPU memory.

For other benchmarks (Hydro, QUDA, HPL) all communications go through host memory as in the third option above. There are two MPI ranks per node, which have their own virtual address space in the host memory, and do their own copies to/from GPU and MPI Send/Receives.

3.1.3 *FPGA matrix computation acceleration, JKU*

In order to attempt to approach the potential orders of magnitude advantage of application specific integrated circuits (ASICs) compared to microprocessors [Dally-2008] a FPGA implementation of matrix multiplication was carried out as a compromise between development effort for a full custom implementation and energy efficiency. For the evaluation of the energy efficiency of FPGAs as compute node accelerators, a Xilinx ML605 Vertex-6 FPGA evaluation kit was used. The FPGA in this evaluation kit has 241 452 logic cells, 301 440 flip-flops, 768 DSP48E1 slices each with a 25×18 bit multiplier, adder and accumulator, 1.9 MB of distributed SRAM, two PCI Express interface blocks and four 10/100/1000 Ethernet MAC blocks. The Vertex-6 FPGA is produced in 40 nm technology. The Vertex-6 supports a clock rate of up to 600 MHz and has 512 MB of DDR3-1066 DRAM. Figure 21 shows a ML605 board.

In order to make effective use of the resource mix of DSP slices and logic in the FPGA the stream architecture proposed in [Strumpen-2004] specialized for the most commonly used matrix-multiplication algorithm was used. Given that FPGAs are intended for a wide range of applications and are programmable they do not come with interface and controller logic and part of the device need to be used for such functions. It was determined that the Virtex-6 XC6VLX240T-1 FPGA used on the ML605 module should be able to accommodate the necessary PCIe interface logic, other controller logic and an 8 × 8 array of tiles for the matrix-multiplication. Figure 22 shows the floorplan.

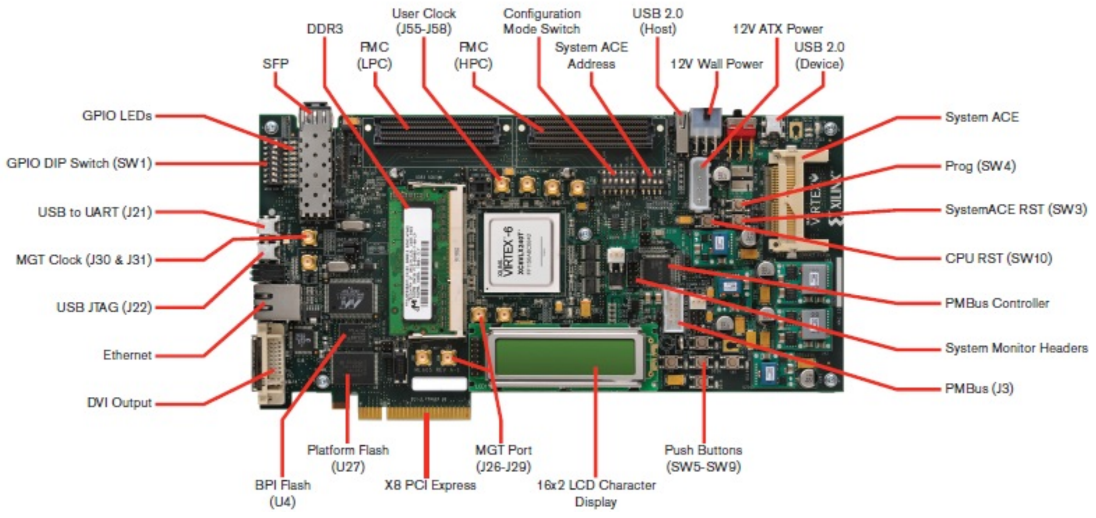


Figure 21 Xilinx Vertex-6 ML605 evaluation module.

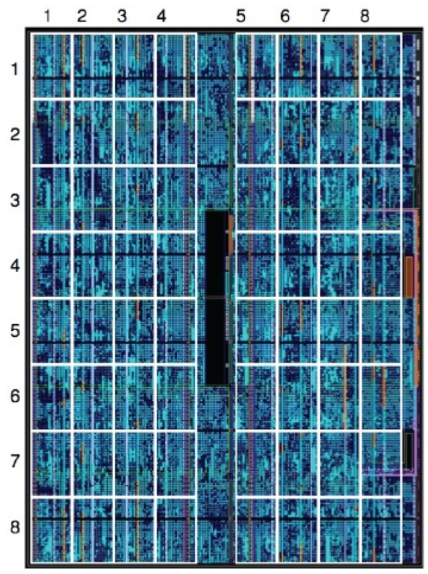


Figure 22 Floorplan of a 8 × 8 stream matrix multiplication accelerator on a Xilinx Virtex-6 FPGA. Each box outlines a tile with a double-precision floating-point multiply-and-add unit, local memory, and communication network.

Each tile contains a deeply pipelined double-precision floating-point multiply-and-add unit, memory, and a pipelined communication network, provisioned to utilize all of the FPGA's DSP slices and memory arrays within the targeted area. Within the time frame of this prototyping effort it was not possible to make the 8 × 8 design fully functional, but a fully functional 7 × 7 design was completed and evaluated for performance and energy efficiency.

The accelerator design supports rectangular matrix multiplication with parameterizable problem size. Using a cache oblivious algorithm [Frigo-1999], large matrices are partitioned into block multiplications on the host processor to minimize the total communication volume between the host and the FPGA accelerator.

3.1.4 *On die integrated CPU and GPU, PSNC*

It is well known that the I/O bus used to integrate accelerators into compute nodes can represent a serious bottleneck limiting the benefit in many applications. Therefore, with the increased ability to fit functional units on a die that comes with decreased feature sizes processor manufacturers have started to produce chips with x86 and GPU cores integrated on the same die. However, though the integration allows for x86 and GPU cores to share memory using fast on-die buses or networks that also are more energy efficient than off-chip I/O buses, one of the strong benefits of non-integrated GPUs is lost: high memory bandwidth. The memory bandwidth to graphics card memory typically is at least five times higher than that of a typical x86 multi-core processor to DRAM.

The PSNC prototypes were used for assessing the energy efficiency and traditional efficiency (fraction of peak) of five different processing chips with integrated x86 and streaming cores: Intel's 4-core Sandy and Ivy Bridge CPUs and AMD's Brazos, Llano and Trinity APUs. The Intel Sandy Bridge and the AMD Llano and Trinity processors are produced in 32 nm technology whereas the Brazos APU is produced using 40 nm technology. The Intel Ivy Bridge CPU is manufactured in 22 nm technology.

Hardware

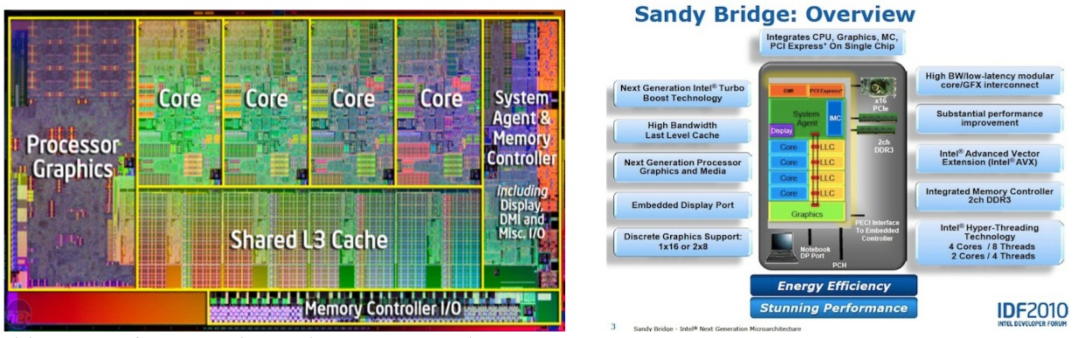

Processor Graphics
Core
Core
Core
Core
System Agent & Memory Controller
Including Display, DMI and Misc. I/O
Shared L3 Cache
Memory Controller I/O
Sandy Bridge: Overview
Integrates CPU, Graphics, MC, PCI Express* On Single Chip
Next Generation Intel® Turbo Boost Technology
High Bandwidth Last Level Cache
Next Generation Processor Graphics and Media
Embedded Display Port
Discrete Graphics Support: 1x16 or 2x8
High BW/low-latency modular core/GFX interconnect
Substantial performance improvement
Intel® Advanced Vector Extension (Intel® AVX)
Integrated Memory Controller 2ch DDR3
Intel® Hyper-Threading Technology 4 Cores / 8 Threads 2 Cores / 4 Threads
Energy Efficiency
Stunning Performance
IDF2010

Figure 23 Intel Sandy Bridge die and block diagram.

The Intel Ivy Bridge i7-3770 4-core CPU, has 32 kB L1 data and 32 kB L1 instruction cache and a unified 256 kB L2 cache per core. The Ivy Bridge processor used for the evaluation had a shared 8 MB L3 cache. The Ivy Bridge server was equipped with 8 GB of DDR3-1600 memory and the CPU clock frequency was 3.5-3.9 GHz.

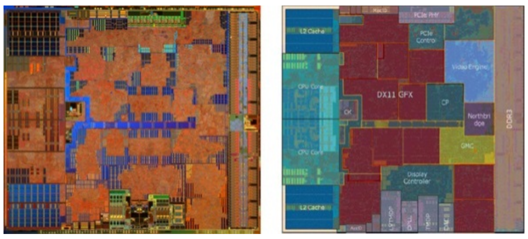


Figure 24 The AMD Brazos die and floorplan.

Figure 25 The AMD Brazos E-350 motherboard.

The AMD Brazos E-350 APU, Figure 24, has two 64-bit x86 cores each with 32 kB of L1 data and 32 kB of L1 instruction cache and a unified 512 kB L2 cache operating at 1.6 GHz alongside 80 Radeon Evergreen thread processors operating at 492 MHz. The processor has a TDP of 18 W and supports DDR3-1066. The evaluated node had 2 GB of DDR3 DRAM operating at 1066 MHz. The Brazos APUs were mounted on MiniITX motherboards in non-rack mountable chassis, Figure 25.

The AMD Llano A8-3870 APU, Figure 26, integrates four 64-bit x86 cores operating at 2.4 GHz, each core having a 64 kB L1 data and a 64 kB L1 instruction cache and 1 MB L2 cache. The die has 400 Radeon cores operating at 600 MHz. The processor supports up to 1866 MHz DDR3 memory. The evaluation server had 4 GB of DDR3-1333 memory. The A8-3870 has a TDP of 100 W.

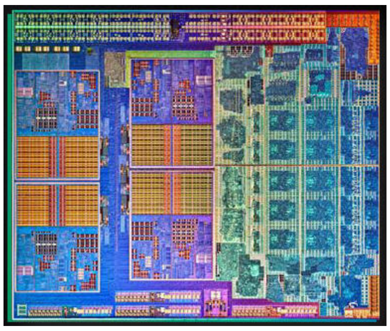


Figure 26 AMD Llano A8-3870 die.

The AMD Trinity A10-5800K APU, Figure 27, integrates two Piledriver modules which translate into four 64-bit x86 cores operating at 2.8 GHz, each core having a 64 kB L1 data and a 64 kB L1 instruction cache and 2 MB L2 cache. The die has 400 Radeon cores operating at 800 MHz. The processor supports up to 1866 MHz DDR3 memory. The prototype had 8 GB of DDR3-1600 memory. The A10-5800K has a TDP of 100 W.

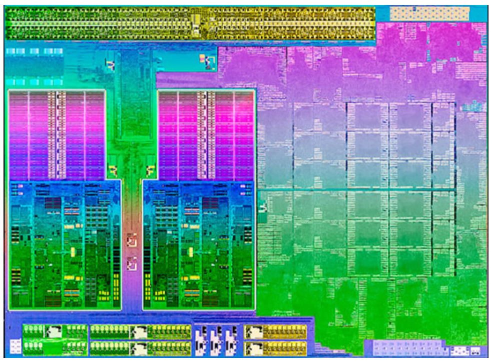


Figure 27 AMD Trinity A-10 5800k die.

Software

All machines were running Ubuntu Server 12.04 with most up-to date versions of Catalyst drivers (most recent 13.1). For AMD platforms ACML versions 5.1, 5.2 and 5.3 were used for BLAS functionality on the x86 cores. The GCC 4.7 and Open64 compilers were used for C code and the AMD APP (Accelerated Parallel Processing) SDK (Software Development Kit) version 2.7 was used to provide OpenCL code targeted for CPU cores. OpenCL 1.1/1.2 was used for the benchmarks. For the Intel platforms MKL version 11 was used to provide optimized BLAS routines and ICC and GCC 4.7 used for C code compilation. Intel OpenCL target (2012) was used for x86 cores. For OpenCL benchmarks clBlas versions 1.6 and 1.8 were used for both x86 cores and streaming cores.

3.1.5 *DSP based node for HPC, SNIC*

Digital Signal Processors (DSPs) targeting the embedded processor market are typically designed for a TDP 10 – 100 times lower than that of high-end CPUs and GPUs. In 2011 Texas Instrument (TI) re-introduced hardware support for double-precision floating-point arithmetic in an 8-core DSP, the TMS320C6678. This DSP has a TDP of 10 W and a peak double-precision floating-point performance of 60 GF/s yielding a theoretical peak energy efficiency of 6 GF/J, though for balanced multiply/add the peak is 40 GF/s and the peak energy efficiency 4 GF/J. As a comparison, the Intel Westmere CPU (the state-of-the art x86 CPU when this effort was started) has a theoretical peak energy efficiency of 0.6 GF/J. The TI DSP is produced in 40 nm technology whereas the Intel Westmere is based on 32 nm technology.

Compared to mobile processor implementations, such as ARM-based processors, the TI DSP has quite good interconnection capabilities with a 40 Gbps point-to-point link called Hyperlink, four 5 Gbps Serial Rapid I/O (SRIO) lanes, two 5 Gbps PCIe Gen 2 lanes and two GbE ports.

Given the large difference in nominal chip level energy efficiency there is a significant potential that a DSP-based HPC node could be very competitive from an energy efficiency point of view, even if the traditional efficiency (fraction of peak) for scientific and engineering applications is lower than that achieved with x86 based cores and servers. This prototype effort aimed at assessing achievable efficiency and energy efficiency of a DSP-based HPC node. This prototype work also includes the design of a FPGA-based switch interfacing to the Hyperlink for a low-latency high-bandwidth interconnect between nodes and to file systems and other services.

Hardware

For the assessment of the energy efficiency of the TI DSP for HPC applications we used an EValuation Module (EVM) including a single DSP with 512 MB of DDR3-1333 DRAM. Figure 28 shows a schematic of the DSP and Figure 29 the EVM.

Figure 30 shows a schematic of our HPC node design based on the DSP, an ARM service processor, and the FPGA based low-latency switch. The ARM processor is intended to provide general services, such as file system services, through system call forwarding, while application codes would execute on the DSPs communicating directly with each other over Hyperlink and the FPGA based network switch.

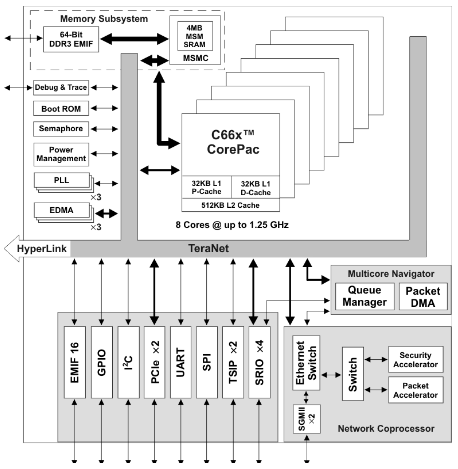


Figure 28 Texas Instruments TMS320C6678 DSP block diagram.

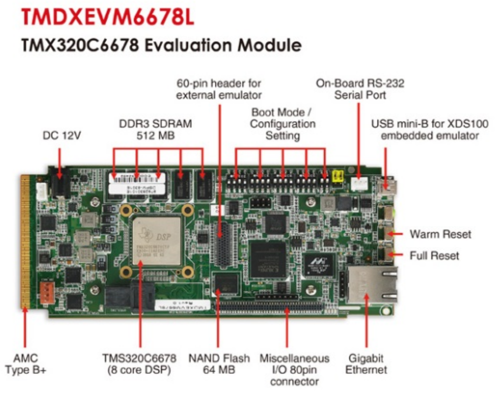

TMDXEVM6678L
TMX320C6678 Evaluation Module
60-pin header for external emulator
On-Board RS-232 Serial Port
DC 12V
DDR3 SDRAM 512 MB
Boot Mode / Configuration Setting
USB mini-B for XDS100 embedded emulator
Warm Reset
Full Reset
AMC Type B+
TMS320C6678 (8 core DSP)
NAND Flash 64 MB
Miscellaneous I/O 80pin connector
Gigabit Ethernet

Figure 29 Texas Instruments TMS320C6678 DSP evaluation module.

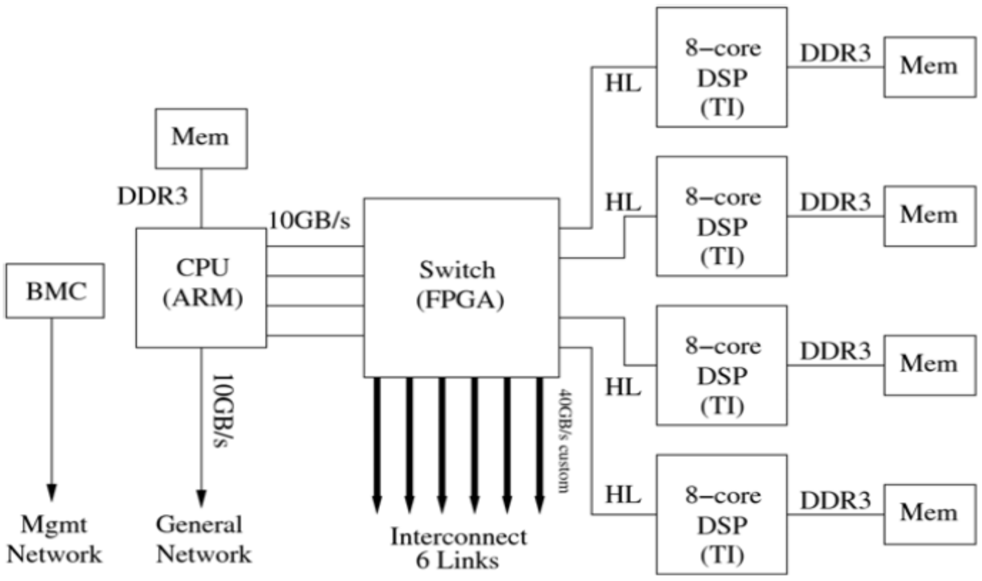


Figure 30 Schematic of HPC node based on TI TMS320C6678 DSPs.

Each DSP core has a 32 kB L1 data and a 32 kB L1 instruction SRAM and a 512 kB L2 SRAM. The SRAMs can be configured to serve entirely as fast memory, or entirely as caches, or as various combinations of cache and fast memory. In addition, there is also an on-die 4 MB shared L3 SRAM. The DSP is designed to support up to 8 GB of DDR3-1600 DRAM, but the evaluation module is equipped only with 512 MB operating at 1333 MHz. The DSP is designed for up to 1.25 GHz clock frequency.

Software

The DSP hardware is not normally used in an HPC context but rather in an embedded telecommunications context. Therefore the UNIX oriented benchmark codes were ported directly onto the "bare-metal" hardware. In the following section the unique features and shortcomings of this special purpose port are presented.

The current software stack as depicted in Figure 31 runs directly on the DSP hardware. It consists of a very thin "runtime" library that contains the standard C runtime library supplied by TI and some own developed functions for accessing hardware related parts as well as simple barrier synchronization and "system" startup code. Some of the benchmarks also use direct access to the underlying hardware during time critical sections of the code that are written in assembler.

The TI Code Composer Studio version 5.3 was used to compile the high-level language benchmarks.

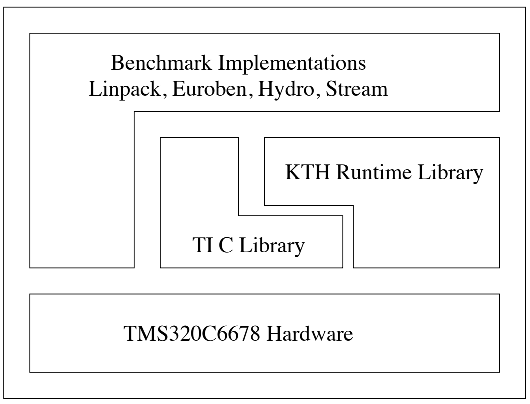


Figure 31 Software stack used to implement the benchmark kernel on the TI DSP hardware.

The six different WP9 Task 3 HPC benchmark codes were all ported to and run on the DSP, but due to time constraints only the HPL benchmark was well, though not fully, optimized. The optimizations were sufficient though to demonstrate the level of efficiency and energy efficiency achievable for HPL and DGEMM.

The STREAM benchmark was the first target for porting. It is written in pure C and uses compiler directives to optimize the actual benchmark operations. It can run concurrently on all eight cores of the DSP and the vector is split into eight consecutive parts, one for each core. Barrier synchronization ensures that the execution time is measured for the slowest core.

The Euroben mod2am benchmark is currently an initial serial port and limited to single core (thread) execution. It is a direct compilation of the Euroben C source code and as such not optimized for the DSP memory hierarchy (i.e. it uses a straightforward dot-product ”three nested loops” algorithm for the matrix multiplication).

The Euroben mod2as benchmark is a simple serial port that can only execute on a single DSP core. It also uses the unmodified Euroben C source for the timed section.

The Euroben mod2f benchmark is also based directly on the C source code which uses a radix-2 / radix-4 vectorized FFT implementation. Furthermore the FFT is parallelized across the 8 cores using the scheme that was present in the Euroben source [Argawal-1994]. Synchronization is again achieved via barriers and additional functions to flush the DSP caches to maintain coherency.

The Hydro benchmark is a straight port of the serial C source code stripped of the VTK based output file production that is not used during the floating-point performance evaluation. The benchmark is currently limited to single core execution.

For the HPL benchmark code was written from scratch and several versions of the benchmark exist. All versions utilize all the eight cores of the DSP during the dominating matrix-multiplication part. A pure C version utilizing the DSP caches (L1 and L2) as well as the fast on-chip shared memory (L3 SRAM) was able to achieve about 25% efficiency. Furthermore a more optimized version of the benchmark was created to boost efficiency to about 80%. This version uses the DSPs L1 and L2 SRAMs as fast scratch pad memory and hides the data transfer latency between those memories and the DRAM by utilizing the built in DMA units, the IDMA units (Internal DMA) for L1/L2 transfers and the EDMA3 units (Enhanced DMA3) for DRAM/L2 transfers. The inner matrix-multiply kernel was hand written in assembler to take full advantage of a software-pipelining scheme. The combination of these two techniques provided a kernel for the matrix multiplication that achieved 95% efficiency, comparable to x86 efficiencies, running on all eight cores. The other kernels of the LU decomposition, triangular solve and vector-matrix LU decomposition, are so far written in C and the latter as well as the back-substitution part are executing on a single core, accounting for the big difference between the matrix multiply efficiency and the total Linpack efficiency.

3.1.6 *Shared memory through a cache-coherency add-in card (NUMA-CIC), UiO*

Shared memory has great appeal from a program development perspective, but unfortunately has become increasingly unaffordable at a large scale. The objective of this prototype is to assess the scalability, efficiency and energy efficiency of supporting shared memory programming models through the use of a novel node board connected to the HyperTransport processor bus of standard AMD based servers.

Hardware

The NUMAscale Cache-coherent Inter-Connect (NUMA-CIC) [NUMAscale] is designed to realize a large scale SMP from common servers through the use of NumaConnect add-on-boards interfacing to a HyperTransport (HT) channel. The prototype consists of 72 IBM 3755 4-way nodes each with two AMD 12-core Opteron 6174 Magny-Cours processors and 64 GB of memory. Of the four sockets in the 3755 server one is left empty, one is used for the NumaConnect pick-up module (NCPUM), and two are used for the Opteron CPUs. The NCPUM extends the HT bus to the NumaConnect card. The NCPUM is necessary because the 3755 does not have any HyperTransport eXpansion (HTX) slots. The 6174 CPUs use HT 3.1 with a maximum bi-directional bandwidth of 51.2 GB/s. The NumaConnect card supports three 6.4 GB/s HT channels resulting in an aggregate 19.2 GB/s bandwidth between NumaConnect cards. Supporting three channels enables the construction of 3D bidirectional topologies. The NumaConnect cards support up to 256 TB of memory and up to 4096 nodes. The node by-pass delay is 53 ns. The NumaChip has a TDP of 17.5 W and the NumaConnect card (see Figure 32) a typical TDP of 25 W with 8 GB of cache memory and 4 GB of tag memory.

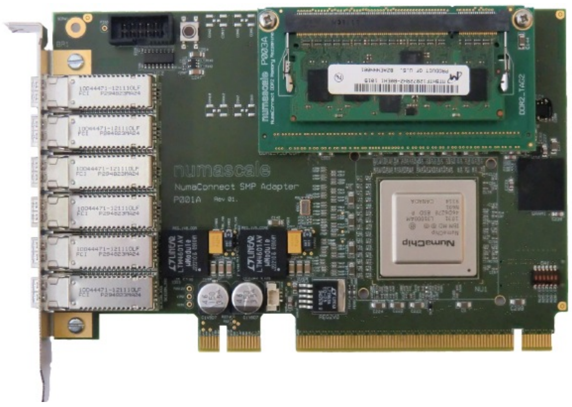


Figure 32 NumaConnect add-on HyperTransport card.

Software

The current Linux kernels (3.x.y) are NUMA aware, i.e., they provide mechanisms to schedule the many threads efficiently. Threads are distributed with tools such as **numactl** which help to launch the multi-threaded job in an efficient way and employs specific NUMA scheduling or memory placement policies.

We used Ubuntu live 10.10 with kernel version 3.4.19-ninja+SMP as operating system. Benchmark applications were compiled with Open64 5.0. The integers need to be 64 bit and the memory model needs to be medium. While this leads to larger executables (text segment), all virtual memory can be used. In practice, the processor's address size (48 bit) limits the addressable physical memory to 256 TB.

All benchmarks used OpenMP to generate threads for parallel computation.

3.2 I/O Prototypes

The objective of the prototypes focused on the I/O subsystem is scalability to Exascale systems. Assessment of energy efficiency as well as traditional efficiency was carried out in addition to evaluation of scalability; and how new storage technology, such as flash, can be incorporated beneficially; and the benefits of spinning down idle harddisks.

3.2.1 *Exascale I/O, CEA/CINES*

The prototype aimed to evaluate innovative storage hardware and software technologies towards Exascale I/O. Special attention was devoted to an efficient, scalable, high performance storage solution, but significant attention was also devoted to data management, which is a critical feature for petabyte scale file systems.

The Exascale I/O prototype was deployed as a distributed system with hardware both at CEA and CINES sites in France, connected to the PRACE network (Figure 33)

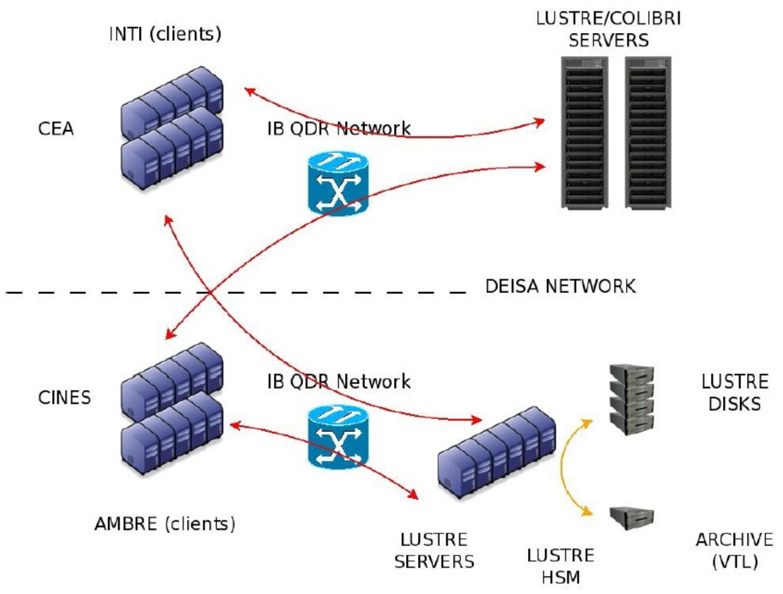


Figure 33 The distributed file system CEA/CINES prototype.

At CEA the prototype initially used Xyratex [Xyratex] ClusterStor 3000 units, Figure 34, but was later upgraded to use two Xyratex racks with nine 5U ClusterStor 6000 High Availability (HA) active/active Object Storage Server (OSS) pairs, a standard HA Metadata Data Server (MDS), a GbE management switch and redundant Infiniband switches (Mellanox IS5035). Each of the 5U HA enclosures has 84 2 TB drives. Each of the 5U Scalable Storage Units (SSU) is configured as eight RAID-6 Object Storage Targets (OSTs). Figure 34 shows the two Xyratex racks, and Figure 35 shows a schematic of the configuration of each of the SSUs. Initially, both CEA and CINES hosted Xyratex CS 3000 controllers, running on Intel Nehalem-based 4-core processors for embedded devices named Jasper-Forest (65 W TDP). One SSU features two embedded server modules, each supporting up to two QDR Infiniband connections to the fabric (only one per server module was used for this prototype). In December 2012, the system at CEA was upgraded to Xyratex CS 6000 by replacing the embedded controllers and installing the latest ClusterStor software (v1.2.1). The new controllers use 8-core Sandy Bridge processors (E5-2648L at 1.80 GHz, 70 W TDP), more memory and PCI-Express Gen 3 device support, like FDR (Fourteen Data Rate) Infiniband. The QDR Infininband network, however, has not been upgraded to FDR. Figure 36 shows the topology of this network.

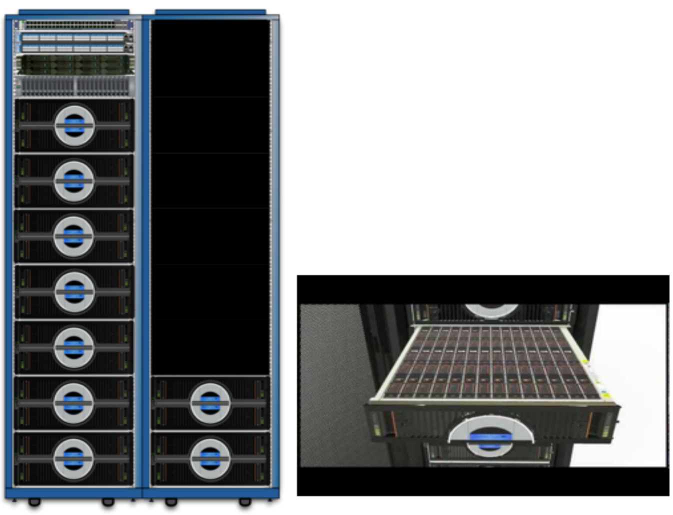


Figure 34 Xyratex racks and drawer.

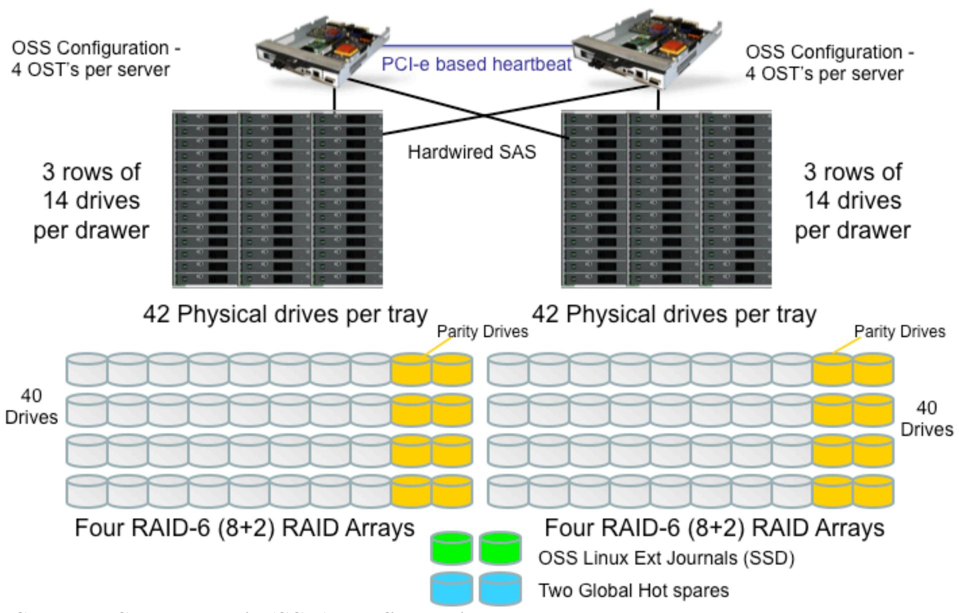

OSS Configuration - 4 OST's per server
PCI-e based heartbeat
OSS Configuration - 4 OST's per server
Hardwired SAS
3 rows of 14 drives per drawer
3 rows of 14 drives per drawer
42 Physical drives per tray
42 Physical drives per tray
Parity Drives
Parity Drives
40 Drives
40 Drives
Four RAID-6 (8+2) RAID Arrays
Four RAID-6 (8+2) RAID Arrays
OSS Linux Ext Journals (SSD)
Two Global Hot spares

Figure 35 Scalable Storage Unit (SSU) configuration.

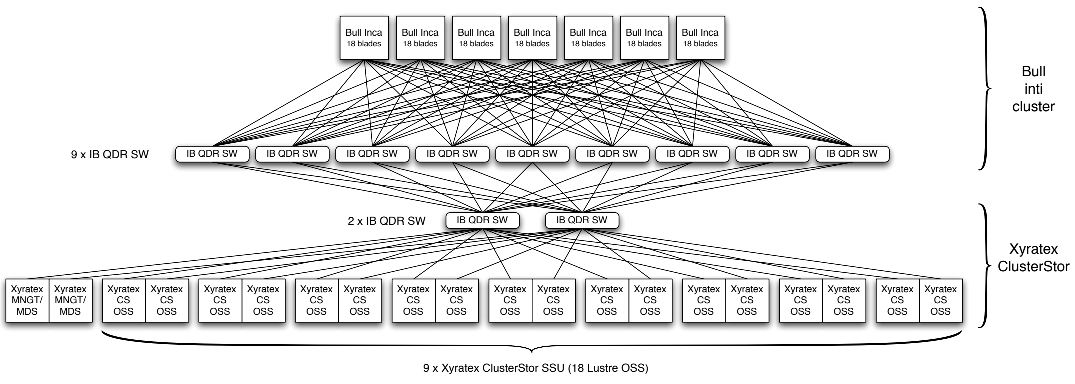


Figure 36 Topology of the Infiniband network of the CEA prototype.

The entire system is configured as a single file system as shown in Table 5.

Total usable capacity	1 152 TB
Total number of HDDs	756
Total number of OSTs	72
Specified write performance	27 GB/s
Total power requirement (under full load)	~20 kW

Table 5 Characteristics of the CEA Excascale I/O prototype file system hardware.

The CINES subsystem also consists of the SGI Data Migration Facility (DMF) for Hierarchical Storage Management (HSM) implemented as part of SGIs COPAN 400, 112 TB Massive Array of Idle Disks (MAID) Virtual Tape Library (VTL) [SGIMAID] configured as 14 virtual drives. This “archive” backend is used locally from CINES clusters and remotely from the CEA cluster. The COPAN MAID technology is very energy efficient by virtue of only powering up disks when needed for data storage or retrieval, resulting in substantial energy savings compared to always-on disks. Assessment of the energy efficiency of this technology is part of the purpose of this prototype. The CINES configuration is illustrated in Figure 37.

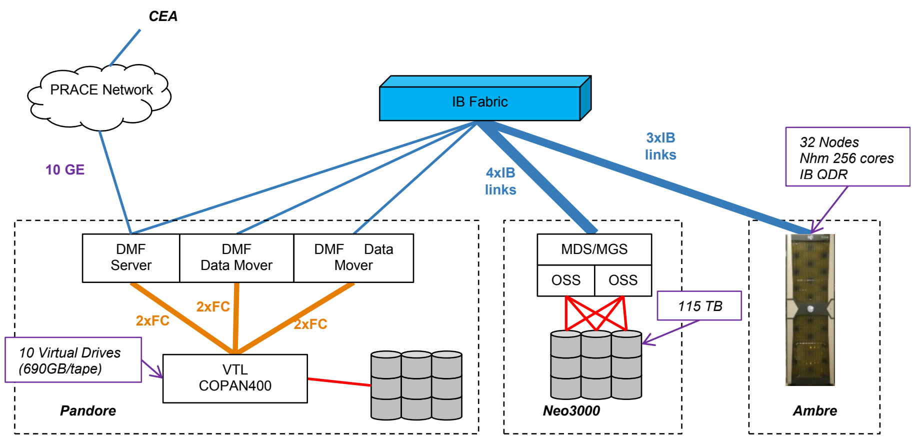


Figure 37 The CINES storage system using MAID technology.

The Xyratex hardware has controller-embedded servers in the SSUs thereby eliminating the traditional need of a storage network infrastructure, like a SAN (Storage Area Network) or DAS (Direct Attached Storage), between servers and storage. This design reduces the total data movement required between clients and disks. Furthermore, management software on each ESM (Embedded Server Module) of the SSUs, named GEM (General Enclosure Management), monitors and controls the SSU’s hardware infrastructure. It adaptively manages efficient cooling of the SSU to keep it in optimal thermal condition, using as little energy as possible.

Detailed hardware deployed on both sites is listed in the Table 6.

To accurately measure energy consumption of an entire Xyratex SSU (storage system building block), a HAMEG 8 kW Power Meter HM8115-2 was used. This instrument has an USB interface that provides data acquisition functionality.

	CEA (Bruyères-le-Châtel)	CINES (Montpellier)
December, 2011	2 x Lustre MGS/MDS (HA) • Nehalem-based 2 x Mellanox Infiniband QDR 36-port switch 9 x CS Neo 3000 SSU • 18 x Jasper Forest controller-embedded servers (OSS) • 36 SSD drives • 720 rotating 1 TB SAS2 drives	2 x Lustre MGS/MDS (HA) • Nehalem-based 1 x Mellanox Infiniband QDR 36-port switch 1 x CS Neo 3000 SSU • 2 x Jasper Forest controller-embedded servers (OSS) • 4 SSD drives • 80 rotating SAS2 drives 1 x SGI COPAN 400 VTL/MAID • 112 x 1 TB rotating drives 1 x 16-port FC Switch Brocade
December, 2012	CS Neo 6000 system upgrade kit : • 2 x Lustre MGS (HA) o Sandy Bridge • 2 x Lustre MDS (HA) o Sandy Bridge • 18 x Sandy Bridge-based controller-embedded servers (OSS)	As above.

Table 6 Detailed prototype hardware description for the CEA and CINES sites.

COPAN software

The COPAN 400T/TX server runs VTL Appliance v 4.50 and CentOS 4.8.

Xyratex software

The latest release of Xyratex software installed at CEA and CINES is ClusterStor 1.2.1. Xyratex provided a pre-installed ClusterStor v1.1 for Neo 3000 in 2011, but did a full software installation on site with the Neo 6000 upgrade at CEA.

ClusterStor 1.2.1 software is mostly built on top of open source solutions, with some Xyratex proprietary software. ClusterStor 1.2.1 uses Scientific Linux 6.1, based on Red Hat Enterprise Linux 6 Update 1. Standard Linux RAID provides Software RAID support (managed by mdadm 3.2.2), the configuration being handled by Xyratex scripts. It is worth noting the use of RAID write-intent bitmaps (WIBS) on mirrored SSD to speed up disk rebuild time in some cases. High availability software, which assembles each Software RAID and mounts Lustre targets, is based on Pacemaker v1.1.6.1 and Heartbeat v3.0.5. Diskless boot of servers is managed by a Xyratex-modified version of GeDI (GEneric Diskless Installer), while Puppet 2.6.12 is used as the configuration management tool. Lustre 2.1 (server-side) is provided and includes some Xyratex patches. Version 2.1.0.x2-74 is provided with ClusterStor 1.2.1.

The following Open Source tools are also installed by default: ClusterShell, a Python framework for cluster administration developed at CEA; PowerMan, a cluster power management utility and LMT v3 (Lustre Monitoring Tool), both developed at Lawrence Livermore National Laboratory.

Data management software components

Currently no HSM (Hierarchical Storage Management) is available natively on a stable release of Lustre, nor does any open source or proprietary software fill this void. However, the roadmap of Lustre includes HSM handling features at the Lustre file descriptor level. CEA and Intel lead the Lustre-HSM development. An early HSM integration has been performed on the CINES configuration with the help of CEA.

For the evaluation, the scenario is a wide spectrum of users accessing a “capability” HPC platform. For this scenario, it is suitable to implement data management based upon implicit service to storage targets considering the expected growth of data. In the next five years, it is likely that the data management decisions will have to use "data-centric" feature based architectures, considering the expected data flows and growing technological gaps between the computation and the storage improvements. Data management by systematic scans of a huge file system is no longer the appropriate option. This study assesses the Robinhood Policy Engine [RPE] as a tool exploiting changelogs from Lustre for data management, comparing it to a global scanning approach.

3.2.2 *Exascale Integrated I/O Subsystem, FZJ*

The objective of this prototype is to assess scalability and energy efficiency of I/O systems integrated into MPP or cluster interconnection networks by directly attaching storage to nodes serving as I/O nodes in the interconnect fabric (as shown in Figure 38). Energy efficiency is improved by eliminating the external switches used in traditional cluster I/O architectures, see Figure 39. This prototype is also used to assess benefits from the use of flash memory instead of, or in combination, with disks. For example, two Blue Gene/Q I/O node drawers including 16 Fusion-io flash drives have a nominal read and write bandwidth of 22 GB/s with a power consumption of less than 2 kW, see Table 7. For comparison, the storage part of the first PRACE Blue Gene/P system JUGENE at the Jülich Supercomputing Centre has a three times higher storage bandwidth (66 GB/s) but consumes 350x more energy.

BG/Q I/O nodes	1	16	16
Fusion-io card type	320 GB	320 GB	640 GB
NAND storage capacity [TB]	0.32	5.0	10.0
Read BW (@64KB) [GB/s]	1.42	22.66	22.4
Write BW (@64KB) [GB/s]	1.37	21.88	16.0
Read IOPS (@4KB) [Mop/s]	0.27	4.32	3.24
Write IOPS (@4KB) [MOp/s]	0.26	4.12	3.5
Power (Watts)	<125	<2000	<2000

Table 7 Energy and performance data for flash based storage.

The following scenarios were implemented:

1. Use of RAID storage controllers or JBODs directly attached to the servers.
2. Use of storage class memory inside servers as pure scratch space or in combination with disks attached to traditional storage servers.
3. Combine both types of storage (disks and storage class memory) inside the server system in a transparent way to use the flash drives as fast cache in a storage hierarchy.

Scenario 1 eliminates the external switches and storage servers. The storage devices (disks) are directly attached to the I/O servers.

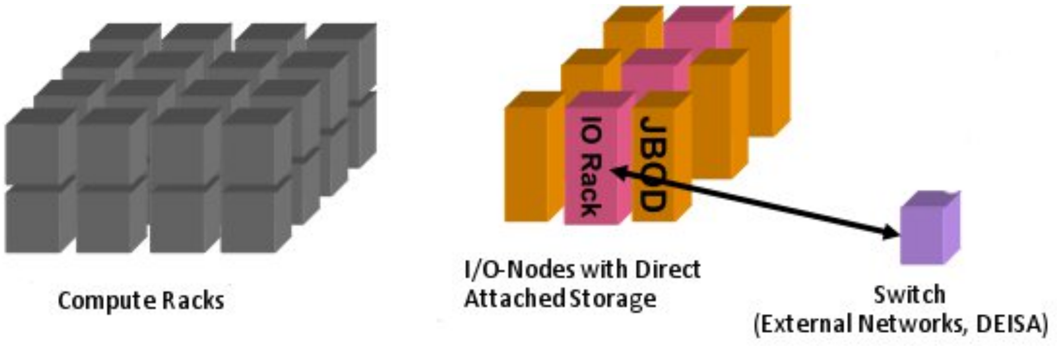

JBOD
IO Rack
Compute Racks
I/O-Nodes with Direct Attached Storage
Switch (External Networks, DEISA)

Figure 38 Storage integrated into MPP/Cluster interconnection network.

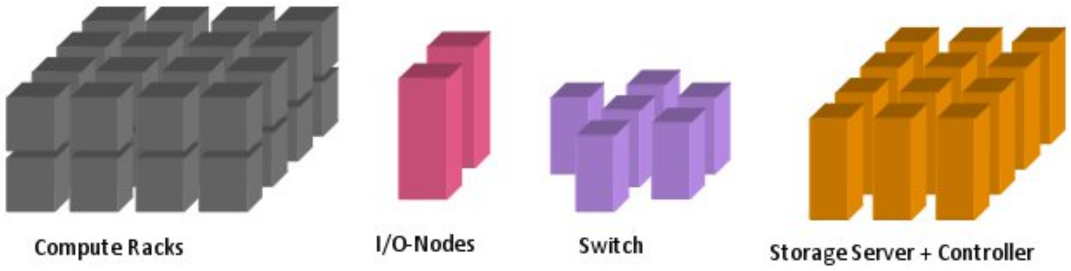


Figure 39 Traditional cluster storage architecture.

In Scenario 2, PCIe flash cards instead of disks are used to form a fast scratch file system. The flash storage is exported as Network Shared Disks (NSDs) so that a regular file system can be created. Thus, full striping over all flash drives is possible without any changes in applications or libraries. In this scenario the user has to manage the transfer of the data to a slower, disk based storage.

In Scenario 3, flash and disk storage are combined into a single file system (as different storage pools) to investigate advanced data management concepts where storage class memories are used as a fast I/O cache and a transparent migration of data to slower disks is managed by the file system. User interaction is not necessary and the data movement can take place while the storage is idle.

In all three scenarios the file system clients or servers run on MPP or cluster based nodes and connectivity to external systems like login nodes, visualization clusters and switches was established to enable data transfer to and from the system via MPP/cluster external networks.

In all cases the complete communication – including TCP/IP – is routed over the MPP or cluster interconnect fabric.

By using storage class memory for *data and metadata* – including automated system based information lifecycle management – this prototype goes clearly further than the PRACE PP-WP8 prototype XC4-IO (see PRACE PP D8.3.2) where standard SSD drives were used exclusively for metadata information.

The I/O system architecture above was assessed both for an x86 based cluster with x86 I/O nodes integrated into a 10 GbE cluster interconnect fabric, and a Blue Gene/Q MPP with BG/Q nodes in the 3-D MPP torus interconnect serving as I/O nodes. The BG/Q architecture has several hardware features that enable a significantly greater portion of I/O and analytical workloads to be shifted "on-board".

The performance was evaluated using standard parallel I/O benchmarks like IOR, mdtest and SIONlib.

Hardware

The Prototype was brought up in two phases. In phase one, x86 servers were used instead of the BG/Q, because of delivery problems. A key component for the MPP prototype is the novel BG/Q I/O node drawer. Each drawer can be equipped with up to eight I/O nodes, each of which is connected to a single standard PCIe bus enabling the use of a wide variety of PCIe based hardware (even devices not officially supported by IBM). All I/O nodes are interconnected through a 3D torus playing the role of a switch fabric.

The following storage solutions have been evaluated:

- Direct attached disks (via FC or SAS controllers)
- Storage class memory-based solutions like Fusion-io's flash memory cards

Fusion-io's revolutionary new solid-state technology dramatically increases bandwidth and reduces latency. It is based on flash memory chips and can achieve 1 million IOPS and 6 GB/s bandwidth from a single PCIe-based card with the recently announced ioDrive Octal.

The deployed hardware is listed in Table 9 and some device performance data summarized in Table 8. A schematic of the x86 cluster prototype is shown in Figure 40 and a schematic of the MPP prototype is shown in Figure 41.

	Fusion-io ioDrive Duo 320GB SLC	**TMS Ramsan80**
READ Bandwidth	1.5 GB/s	700 MB/s
WRITE Bandwidth	1.5 GB/s	700 MB/s
READ IOPS	261000	170000
WRITE IOPS	262000	80000
Access Latency	26µs	100µs (read) / 30µs (write)
Bus Interface	PCI-Express x4/x8	PCI-Express x8

Table 8 Some I/O device performance data.

Blue Gene/Q specific software

- *OFED verbs provider (RoQ device driver):*
  This driver plays the key role in enabling the BG/Q I/O torus as a switch fabric. It provides an iWARP interface that can be used by the OFED software stack to implement higher level protocols like TCP/IP.
- *Fusion-io driver:*
  The driver for the Fusion-io cards was ported to the BG/Q architecture.
- *FC HBA driver:*
  Drivers for common Fiber Channel HBAs are generally part of the Linux kernel and have been adapted and tested within the BG/Q environment.
- *GPFS (General Parallel File System) client and server on Blue Gene/Q I/O nodes:*
  GPFS was intensively tested with the new BG/Q networking infrastructure replacing standard Infiniband or Ethernet networks.

Hardware independent software (GPFS)

The *GPFS* from IBM was used for the file system on the JUNIORS prototype. It can use disks and – using a small patch – flash devices as NSD. The NSDs are used to create a GPFS file system.

The following GPFS features have been used to implement the Information Lifecycle Management (ILM) for Scenario 3:

- Define multiple storage pools (e.g. *flash* and *disk*).
- Define failure groups for redundancy.

- Use the GPFS Policy engine
  - Placement policy rules:
    Determine which storage pool to use when a file is created (setup once)
  - Migration policy rules:
    Determine the files to move from the fast flash storage pool to the slow disk storage pool by defining thresholds (high & low) and a weight function.
- Define callback:
  - Bind *LOW_SPACE* event to *mmstartpolicy* to trigger migration policy.

Step 1	4x IBM System x3650 M3 2x fusionIO ioDrive Duo 320GB SLC 2x Dual-Port ConnectX2-EN 10 GbE 1 Building Block (FC attached storage) 2x IBM System x3650 M3 2x Dual-Port ConnectX2-EN 10 GbE 1x IBM DCS3700 Storage System + Expansion Unit 10x 2 TB 7200 rpm 6 GB NL-SAS 1 Building Block (SAS attached storage) 2x IBM System x3650 M3 2x Dual-Port ConnectX2-EN 10 GbE 2x Dual-Port Qlogic 8 Gbps FC Adapter 1x IBM DCS3700 Storage System + Expansion Unit 10x 2 TB 7200 rpm 6 GB NL-SAS 1x IBM System x3650 M3 (Management-Node) 1x Cisco Catalyst 4948 (48 Port) JSC has provided the required 10 GbE ports for the I/O network
Step 2	1x Service Node POWER7 720 with dual-10 GbE 1x BG/Q IO-rack with bulk power and clock card 4x BG/Q IO-drawer @ 8 IONs, each ION with dual 10 GbE 48x BG/Q torus cables (for ION torus cabling) Storage Class Memory (TBD) Fiber Channel HBA's 10 GbE adapter (for external connectivity)

Table 9 Integrated I/O system prototype hardware.

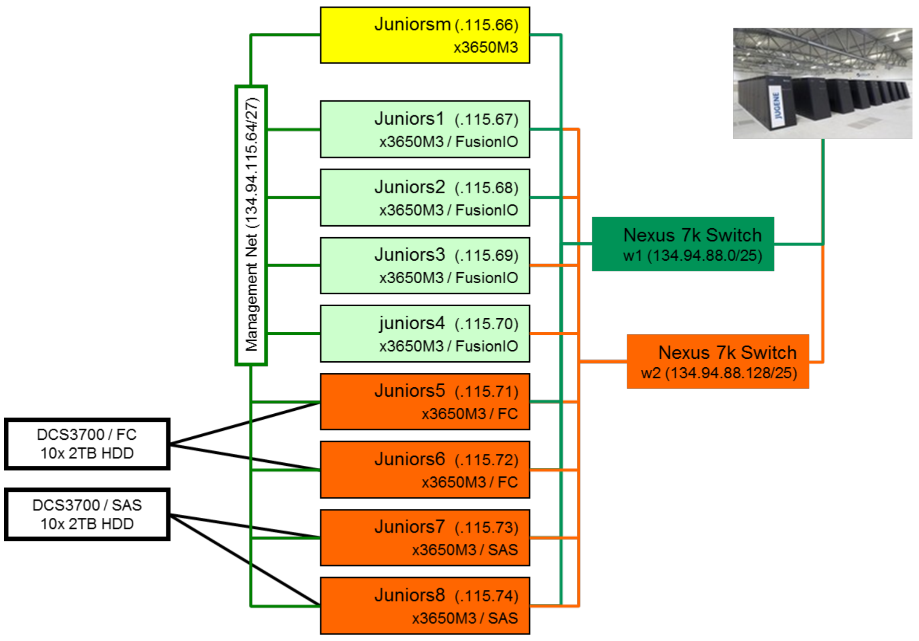

Juniorsm (.115.66) x3650M3
Management Net (134.94.115.64/27)
Juniors1 (.115.67) x3650M3 / FusionIO
Juniors2 (.115.68) x3650M3 / FusionIO
Juniors3 (.115.69) x3650M3 / FusionIO
juniors4 (.115.70) x3650M3 / FusionIO
Juniors5 (.115.71) x3650M3 / FC
Juniors6 (.115.72) x3650M3 / FC
Juniors7 (.115.73) x3650M3 / SAS
Juniors8 (.115.74) x3650M3 / SAS
DCS3700 / FC 10x 2TB HDD
DCS3700 / SAS 10x 2TB HDD
Nexus 7k Switch w1 (134.94.88.0/25)
Nexus 7k Switch w2 (134.94.88.128/25)
JUGENE

Figure 40 Schematic of the cluster I/O prototype.

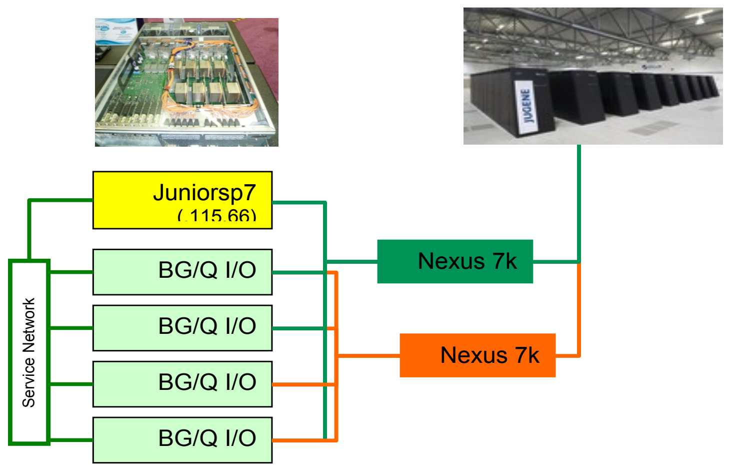


Figure 41 **Schematic of the MPP I/O prototype.**

3.3 Holistic Approach to Energy Efficiency, LRZ

The CooLMUC prototype at LRZ, (Figure 42), was built by MEGWARE Computer in collaboration with Kälte Klima Umwelt. The main goal of the prototyping effort was to evaluate the benefits of direct warm-water cooling and waste-heat reuse through adsorption refrigeration. It was first put into service in July 2011 and subsequently moved to its current location in the new building of LRZ where it is hooked up to a new warm-water cooling loop.

The following tools and technologies were assessed for the CooLMUC prototype:

- Power monitoring and management tools.
- Warm-water cooling for standard cluster architectures.
- Waste-heat reuse to drive an adsorption chiller.
- Energy-to-Solution for various workloads.

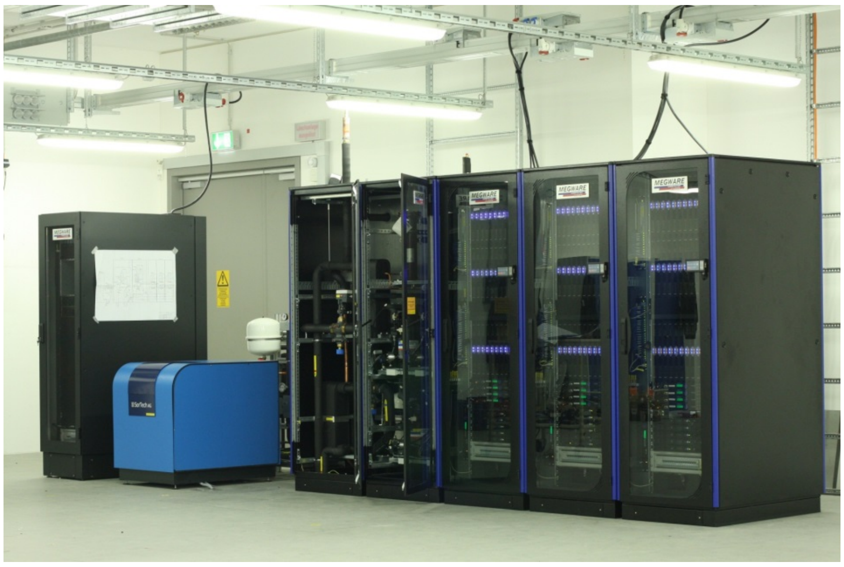


Figure 42 **CooLMUC prootype at the Leibniz Supercomputer Centre.**

Hardware

The CooLMUC cluster at LRZ consists of 178 nodes, each containing two AMD 8-core Opteron 6128HE CPUs (Magny-Cours) with 12 MB L3 cache. In their standard setting, the CPUs run at 2 GHz clock frequency. Each node is equipped with 16 GB RAM arranged in eight 2 GB DDR3 modules running at 1333 MHz.

The node interconnect is a QDR Infiniband fat tree. Each node also has two GbE ports for IPMI (Intelligent Platform Management Interface) and a service network, which is used to boot the diskless nodes and to provide the root filesystem over NFS. A network bridge component connects the Infiniband network to the upstream Ethernet services at LRZ.

A central appliance server provides cluster management functions, such as temperature/power monitoring and remote power control, and also acts as the NFS server providing the OS images to the nodes.

The cluster is arranged in five racks with the 178 cluster nodes and their interconnection network contained in three racks. The cooling components are contained in the other two racks. A SorTech [SorTech] ACS-08 adsorption chiller is used to turn the hot water emitted from the cluster into cold water. The cold water is then used to cool the rear-door heat exchanger of a sixth, otherwise unrelated rack. The setup is shown in Figure 42. Figure 43 shows the liquid node cooling.

To make the cooling independent of LRZ's CRAC infrastructure, the rack doors are solid. Two independent cooling loops are used to cool the enclosed equipment. One loop provides water at 40 ºC directly to the nodes through copper pipes connecting special heat sinks on top of CPUs, chipset, and IB HCAs. This technology completely eliminates the necessity to use air as a heat transfer medium for the corresponding components. Yet, some components remain that rely on air cooling, such as power supply units contained in each compute node, IB switches, and Power Distribution Units (PDUs) in the racks. The second cooling loop provides cooling to those components. It is based on standard compressor-based cooling technology with special 19" in-rack evaporators that generate airflow from the rear part of the racks to the front while at the same time re-cooling the air to the set temperature of 30 ºC.

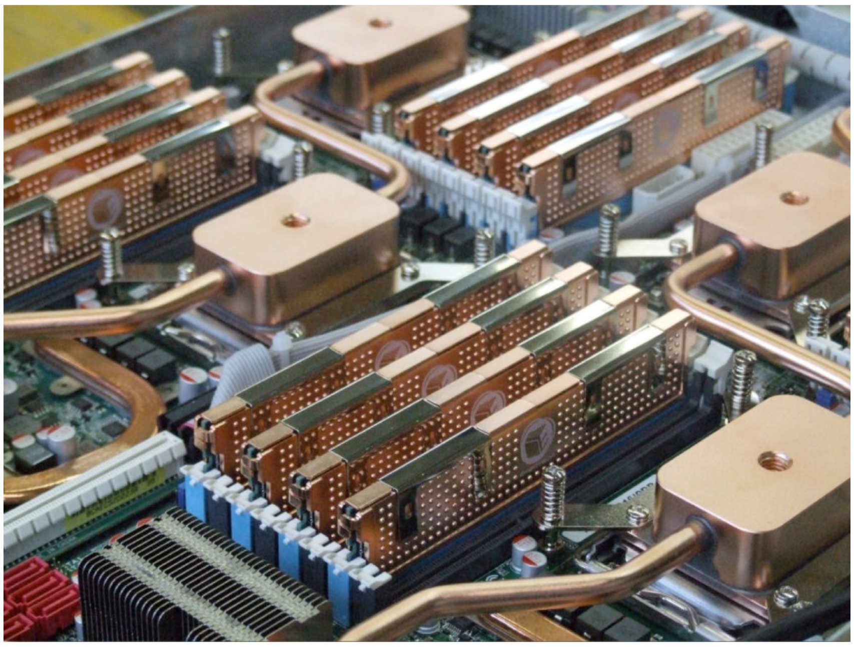


Figure 43 CoolMUC compute node with copper pipes for direct liquid cooling.

In order to use the heat collected from the two cooling circuits to drive the adsorption chiller, the condenser of the second cooling circuit is cooled with water originating from the first cooling loop's outlet. This way, the server water inlet temperature of 40 ºC can be kept while providing supply water temperatures of 60 ºC to the adsorption chiller, see Figure 44.

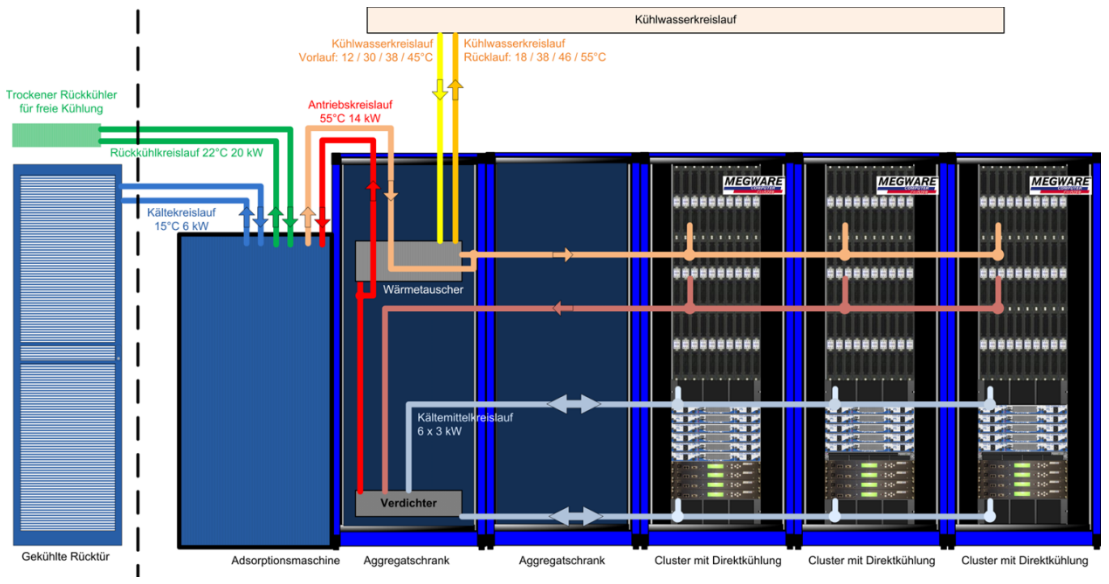


Figure 44 CoolMUC cooling system schematic.

4 Instrumentation

In this section we describe the instrumentation of the different prototypes. Energy efficiency measured as energy to solution is an objective of all the prototypes, but not the only objective. Assessing the traditional efficiency measured as fraction of peak is also an objective. For energy efficiency it is not only important to assess the energy to solution measured as electric energy consumed for computation and storage, but also energy consumed for cooling and other necessary services, and recovered energy when an energy recovery system is put in place.

Most of the prototypes focused on assessing the electric energy consumed for particular tasks/benchmarks, but the LRZ prototype had a holistic objective and also included technology and instrumentation for cooling and energy recovery.

For energy efficiency, as well as traditional efficiency focused on hardware resource utilization, measurements at different levels of detail are required depending on the objectives. Whole system efficiencies may suffice for selecting different system implementations, while component level efficiencies often are required for R&D targeting technology and/or architectural choices as well as refined operational decisions. The PSNC and SNIC prototypes were instrumented to enable study of the energy consumption of major server components during different phases of application program execution.

4.1 Node architecture prototype measurement setups

4.1.1 *An NVIDIA Tegra 2 mobile SoC based HPC cluster, BSC*

For energy to solution assessment, this prototype was instrumented for node and rack level power measurements, Table 10. The nodes use passive cooling and so does the rack. The rack level measurements do include power consumed by the interconnection network and PDUs in the rack in addition to power consumed by the nodes. No attempts were made to measure energy required for handling the heat generated by the racks, and no energy recovery was attempted specifically for these racks. Figure 45 shows a diagram of the instrumentation and instrument characteristics are listed below.

No.	Device	Scope/Purpose	Measurand	Error	Sampling
1	Integrated power meter	Power measurement of racks	Active power (W) Voltage (V) Current (A)	± 100W	1/sec
2	Yokogawa wt230	Power measurement of a subset of nodes	Active power (W) Voltage (V) Current (A)	± 0.2%	4/sec

Table 10 BSC NVIDIA Tegra 2 mobile SoC based node Prototype Characteristics.

The power meter integrated into the racks is intended for measuring multi-kW loads with a resolution of ±100 W. Due to the fact that a Tegra 2 node consumes around 8 W, and a 1U container around 70 W the rack power meter is not accurate enough for the Tegra 2 prototype and not used to report energy efficiency results.

For the energy efficiency assessments the node level measurement setup was used. For the results reported the power consumption of one 1U container (eight nodes) is measured and then scaled with the number of nodes that were included in the benchmark. Since the load per node does not vary significantly within one run, measuring a single node and then scaling the power does not introduce any major error. This was confirmed with several tests that took place before the actual measurements were carried out. The power consumption of the networking equipment was not measured.

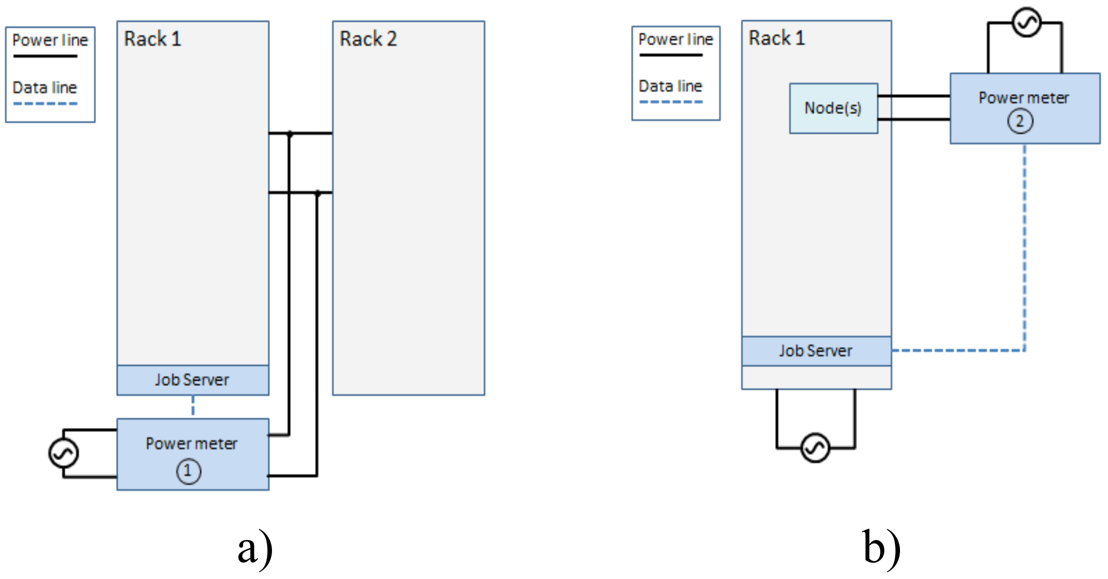


Figure 45 Rack power a) and node power b) measurements of the Tegra 2 cluster.

For both whole-rack and per-node measurements, the power meter is connected to the job server with collection of samples automated. When a job is started, the cluster management software will start the measurements, and power samples will be delivered together with output files from the job. For whole-rack measurements the power meter is connected through the network to the job server, and for single node measurements via serial cable. For the single node measurements a driver was written for the power meter to fully automate the process.

4.1.2 *GPU-GPU communication over PCIe and IB, CaSToRC*

The power supply for the motherboard of each server is separate from the power supplied to the two GPU boards in a node, allowing for separate power measurements. The server-level power consumption analysis was performed by means of xCat's rvitals utility, which reports the server power consumption. No explicit instrumentation of the servers was made.

For the energy efficiency assessment, application measurements were taken during about five minutes. An idle time of about one minute was allowed at the beginning to reliably estimate the idle power consumption. Furthermore, to determine errors, multiple measurements were made for each benchmark. Samples of power consumption where taken every 3-10 seconds and stored in text files for analysis. Figure 46 shows a typical power recording of all 16 GPUs plotted separately in the same plot. The shaded area in this plot is considered as the effective energy consumed for running the application, which divided by the wall-time yields the average effective power consumption.

The prototype is deployed in CaSToRC's data center, which provides the ability to log environmental readings. However, for the results provided here we do not use environmental readings of ambient humidity and temperature.

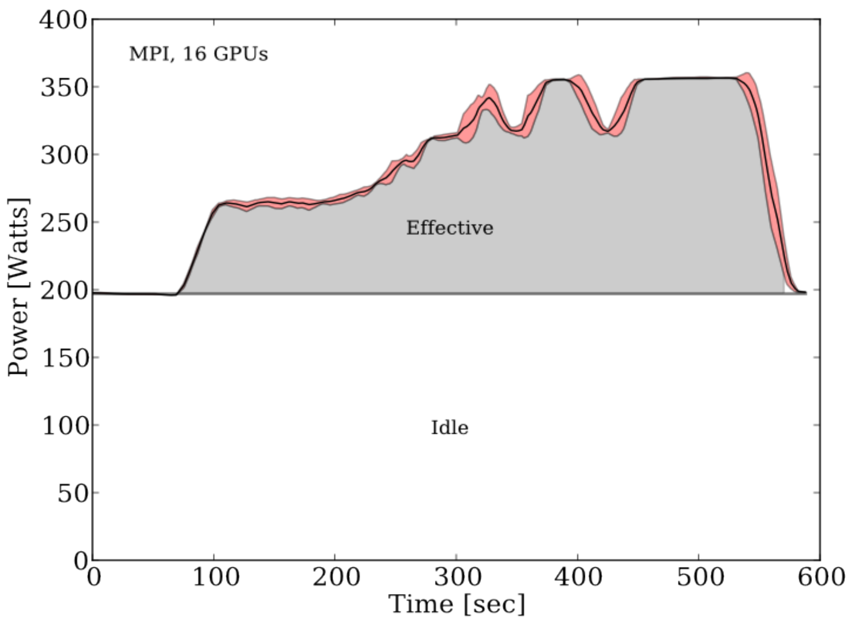


Figure 46 Example power measurement for the STREAM benchmark on all 16 GPUs of the prototype.

4.1.3 *FPGA matrix computation acceleration, JKU*

The FPGA includes system monitor hardware to manage analogue on-chip and off-chip sensor inputs. FPGA core power and ML605 board power consumption were measured with a sampling period of 5 microseconds and the Xilinx ML605 board plugged into the PCI slot of a standard PC with USB and JTAG connectivity to a separate PC for data collection.

No.	Device	Scope/Purpose	Measurand	Error	Sampling
1	Virtex 6	FPGA core temp.	Temperature	+/-	5 µs
		FPGA supply voltage	Voltage	+/-	5 µs
		FPGA supply current	Current	+/- 1%	5 µs
		ML605 supply	Voltage	+/- 1%	5 µs
		ML605 supply	Current	+/- 1%	5 µs
2	KTY81-210	External	Temperature	+/-	5 µs

Table 11 On-chip and off-chip quantities measured by FPGA system monitor.

Table 11 lists the on-chip and off-chip quantities measured by the FPGA system monitor. Furthermore, we have built an external temperature sensor to test the external sensor input capability of the system monitor.

4.1.4 *On die integrated CPU and GPU, PSNC*

For power measurements three National Instruments (NI) PXI-6255 high accuracy measurement modules were installed in a NI PXIe-1082 chassis together with a 2-port NI PXI-8433/2 serial interface module and a NI PXIe-8133 controller module. Additionally, shunt resistors were installed to enable current measurements to groups of components on the server motherboards, such as CPUs, memory and "uncore". For rack power consumption, a Lumel P43 programmable transducer was used for 3-phase power. For total power at the individual server level a 1-phase Lumel P12P programmable transducer was used. The transducers are connected to the NI controller using a RS485 communication interface. The Direct Current (DC) is measured using the NI PXI-6255 modules. The data acquisition, storage and analysis make use of National Instruments LabView software running on the NI

PXIe-8133. Data is gathered in a standard SQL database for easy access by any external software. A schematic overview of the prototype's instrumentation is shown in Figure 47. The properties of the instrumentation are summarized below in Table 12

Number	Device	Scope/Purpose	Measurand	Error	Sampling
1	(AC) Lumel P43	3-phase measurement of power consumption for whole rack	Voltage (V)	0,2%	3/sec
			Active Power (W)	0,5%	
			Current (A)	0,2%	
2	(AC) Lumel P12P	1-phase power consumption per node / switch	Voltage (V)	0,2%	3/sec
			Active Power (W)	0,5%	
			Current (A)	0,2%	
3	(DC) NI PXI-6255	Fine grained measurement of node internal power lines	Voltage (V)	<1%[2]	5-9k/sec
			Active Power (W)	<1%[2]	
			Current (A)	<1%[2]	

Table 12 Instrumentation properties.

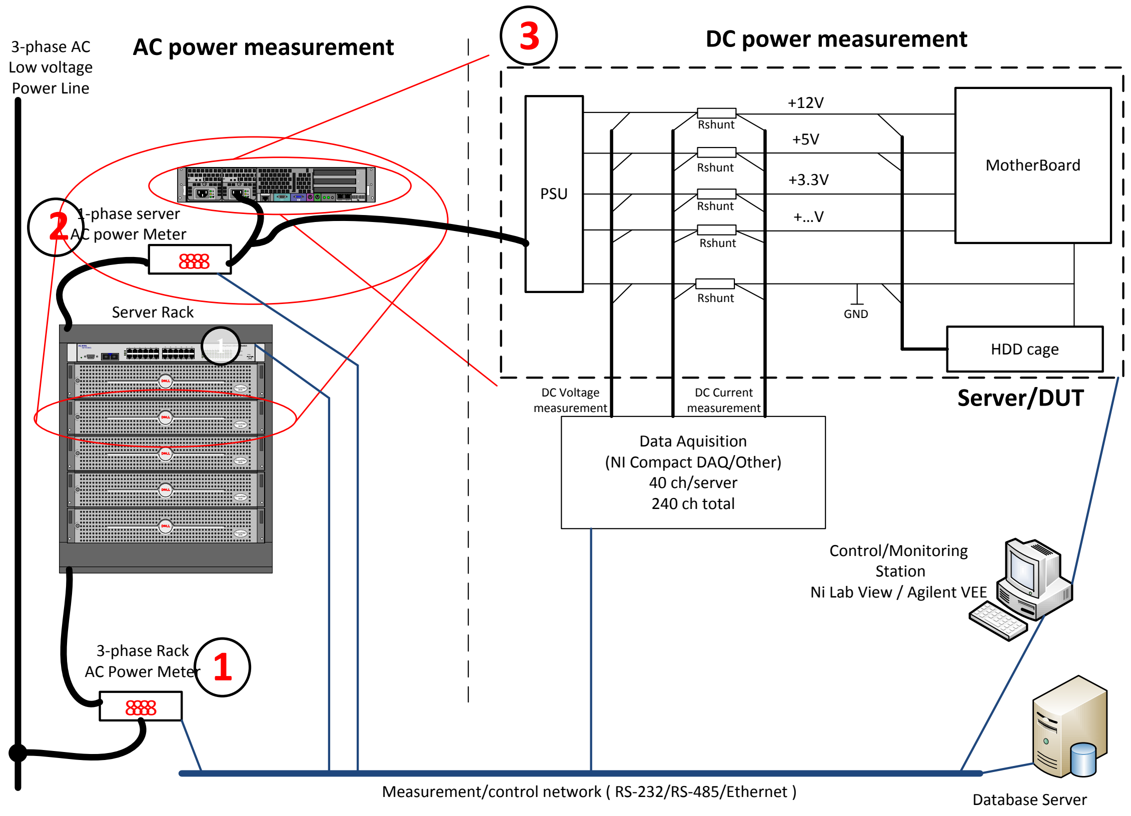

3-phase AC Low voltage Power Line
AC power measurement
3
DC power measurement
2
1-phase server AC power Meter
Server Rack
3-phase Rack AC Power Meter
1
PSU
+12V
+5V
+3.3V
+...V
Rshunt
GND
MotherBoard
HDD cage
Server/DUT
DC Voltage measurement
DC Current measurement
Data Aquisition (NI Compact DAQ/Other) 40 ch/server 240 ch total
Control/Monitoring Station Ni Lab View / Agilent VEE
Measurement/control network (RS-232/RS-485/Ethernet)
Database Server

Figure 47 Schematic overview of the power measurement system for the prototype.

4.1.5 *DSP based node for HPC, SNIC*

The main goal of the instrumentation of this prototype was to allow meaningful prediction of the energy efficiency of a real world HPC node deployment scenario using the DSP processors. The challenge in achieving this consisted in the fact that, as in the case of the ARM based nodes studied at BSC, the EVMs were not intended for HPC use or as a development environment for HPC applications. In fact, the modules contained very little memory and included a number of debug and I/O features not needed in a HPC node. Therefore, a rather detailed, fast instrumentation was designed and implemented that enables quantifying the power consumption during application execution for the two main power

[2] The real accuracy of the NI PXI-6255 device is 0,2% but we have to account for additional error margin introduced by the shunt resistors

consumers that would also be present in a HPC node design:

- The DSP cores.
- The DRAM memory subsystem.

Figure 48 shows the instrumentation of an EVM. Small shunt resistors were added at four points to allow sensing of the current delivered to major components of the EVM. Sensing was also added to measure the voltage present on the rails fed by the shunt resistors to be able to accurately calculate the power. The sensed signals were fed to an analogue front-end consisting of:

- Amplifiers to convert and scale the differential input signals and remove common mode noise.
- A 9th order Butterworth anti-alias filter to remove high frequency components for each channel.

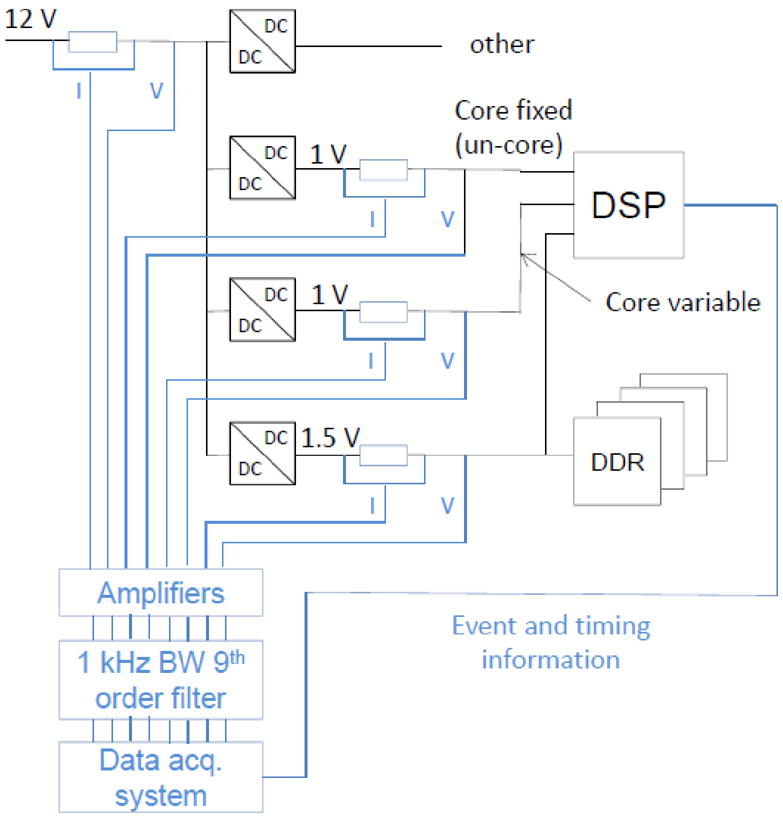


Figure 48 The power measurement instrumentation of the DSP EVM prototype.

The filtered signals were digitized by a National Instruments cRIO-9074 based system using NI-9205 ADC (Analog to Digital Converter) modules. The digitized data is sent via Ethernet using UDP (User Datagram Protocol) to the acquisition host. There it is time-stamped and correlated with benchmark progress information received from the DSP via a RS232 line. Figure 49 shows instantaneous power measurements during HPL benchmark runs (looped) displayed using an in-house developed real-time viewer for demonstration and verification purposes.

For the Euroben, STREAM and Hydro, benchmarks that were carried out before the just described instrumentation was implemented, a simpler instrumentation as shown in Figure 50 based on bench top power meters (Instek GPM8212) and multimeters (Tektronix DWM 4050) as well as the built-in sensors in the DC/DC converters (PMBUS) was used. Accuracy for the GPM8212 is 0.2%, the DWM 4050 plus shut resistor has 1% accuracy due to the shunt resistor but only senses current. Voltage is regulated to within 2.5% by the DC/DC converter.

The PMBus (Power Management Bus) information has about 5% accuracy according to datasheet specifications. All sources deliver between 1 and 3 samples per second.

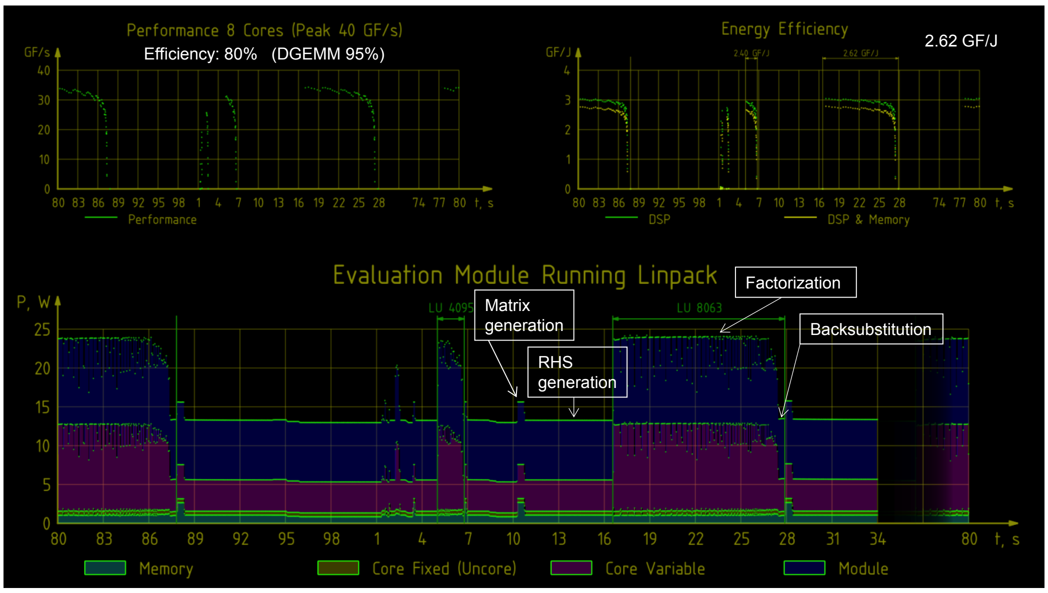


Figure 49 Illustration of measured EVM instantaneous power for HPL runs (looped).

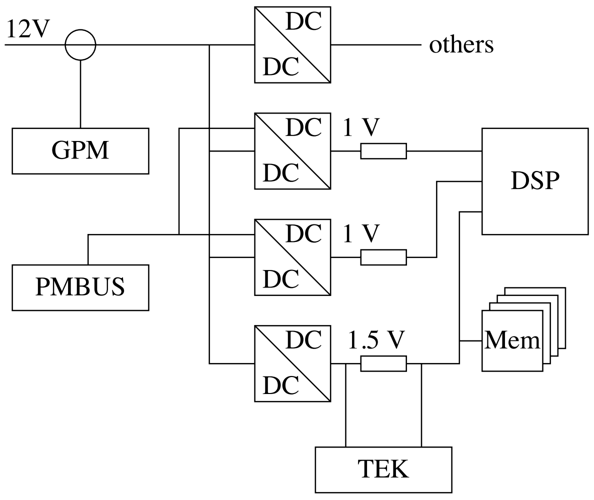


Figure 50 Older instrumentation of the DSP EVM using build power sensors and multimeters.

Accuracy

The overall accuracy of the measurement system is influenced by the following factors listed in Table 13.

The error estimation in Table 13 covers only the actual measurement error for each of the sensed channels. By far the biggest contribution (~5 %) comes from the uncertainty of the delineation of the system from the surrounding components. For instance, the 1.8 V DSP I/O rail is not measured, the DSP may consume up to 0.6 W on this power supply according to design documentation. The sampling rate is 7000 samples per second and the cut-off frequency of the anti-alias filter is 1000 Hz. The stop-band rejection at the Nyquist limit of 3500 Hz is better than 93 dB resulting in a maximum alias error of 2 LSB (Least Significant Bit) (25 ppm).

Data Processing

Most of the processing of measurement data is done offline. A small utility captures the time-stamped measurements and stores them in standard HDF5 formatted files. Offline programs are used to calculate instantaneous power and integrate this into energy values using a normal trapezoid rule.

Component Error	Relative Error	Contribution
Shunt Resistor (ΔT ± 20K)	0.1% + 20 ppm/K	1800 ppm
Front End Offset Error	0.01%	2 mV
Front End Gain Error	0.1%	1000 ppm
Front End Passband Flatness	0.1%	1000 ppm
Front End Stopband Rejection	25 ppm	25 ppm
ADC Offset Error	140 ppm	2.4 mV
ADC Gain Error	476 ppm	476 ppm
Total Voltage Error		*< 1421 ppm + 9.6 mV*
Total Current Error		*< 2294 ppm + 19.2 mA*
Total Power Error		*< 9500 ppm*
Marker Jitter (1000s run-time)	4 ppm	4 ppm
Total Energy Error		*< 9500 ppm*

Table 13 Error budget for the DSP measurement system. The total errors are dependent on the sensing channel in the table the worst case is specified.

4.1.6 *Shared memory through a cache-coherency add-in card (NUMA-CIC), UiO*

Each of the IBM servers is connected to an intelligent PDU (IBM Switched C19/C13 PDU, 32A) that provides information about the common electrical parameters like voltage, current, phase etc. These PDUs have a web interface that provides the user with full control over each single outlet, including the above parameters. Each of the servers has only one power supply unit.

The PDU does not provide a scriptable interface or suitable logging facility. It can produce a Java based plot, but it was found to be of limited value. For power measurements a command line script was written that extracts information from the web interface for each power outlet and provides a power value. All outlets belonging to all servers in a Numascale system are lumped together to provide a common total power value for the system. The values logged are active power which is the actual/real power used.

Unfortunately the time it takes to extract the data from the PDUs is rather long. It takes many seconds to get numbers for the system, usually about 10 seconds. The resolution is hence rather coarse. The monitoring script is providing a time stamp and a wattage number for about every 10 seconds. An average value from a run can then easily be calculated as a benchmark run usually takes from many minutes to many hours or even days when accessing large memory footprints.

A typical power monitoring script output looks like this:

```
Wed Feb 27 15:37:05 CET 2013 : 1361975825  : 3057  : Watt
Wed Feb 27 15:37:15 CET 2013 : 1361975835  : 3179  : Watt
Wed Feb 27 15:37:27 CET 2013 : 1361975847  : 4063  : Watt
Wed Feb 27 15:37:38 CET 2013 : 1361975858  : 5037  : Watt
Wed Feb 27 15:37:48 CET 2013 : 1361975868  : 5057  : Watt
Wed Feb 27 15:38:00 CET 2013 : 1361975880  : 5059  : Watt
```

The time stamp is given in two forms one for humans to read and one Unix epoch time in seconds. As the script is looping as fast as it can the time intervals are not regular. Providing as many samples as possible has been given priority as post-processing can easily transform data to regular intervals or averages if needed. Power monitoring samples are gathered before, while and after running the benchmarks and hence provide a fairly good overview of the power consumption.

4.2 I/O Prototype instrumentation

4.2.1 *Exascale I/O, CEA/CINES*

All the prototype hardware is connected to "intelligent" PDUs that are able to measure power consumption globally and on an outlet basis. The prototype hardware components, like the nodes and the storage controllers, are also able to measure their own power consumption through internal sensors. The accuracy of these sensors is unfortunately not guaranteed. We used an external HAMEG HM8115-2 power meter to measure the active power of the Xyratex Scalable Storage Unit. The accuracy of the meter is known to be +/-1 W for active power measurements up to 2400 W. Table 14 summarizes the instrumentation.

Number	Device	Scope/Purpose	Measurand	Error	Sampling
1	CSG-24VD/Y	PDU	Active power (W) Voltage (V) Current (A)	+/-2%	<10 sec
2	Node sensor	Power supply	Voltage (V) Current (A)		1 sec
3	HAMEG 8115-2	Power Meter	Active power (W)	+/-1 W	1 sec

Table 14 Summary of the instrumentation.

4.2.2 *Exascale integrated I/O subsytem, FZJ*

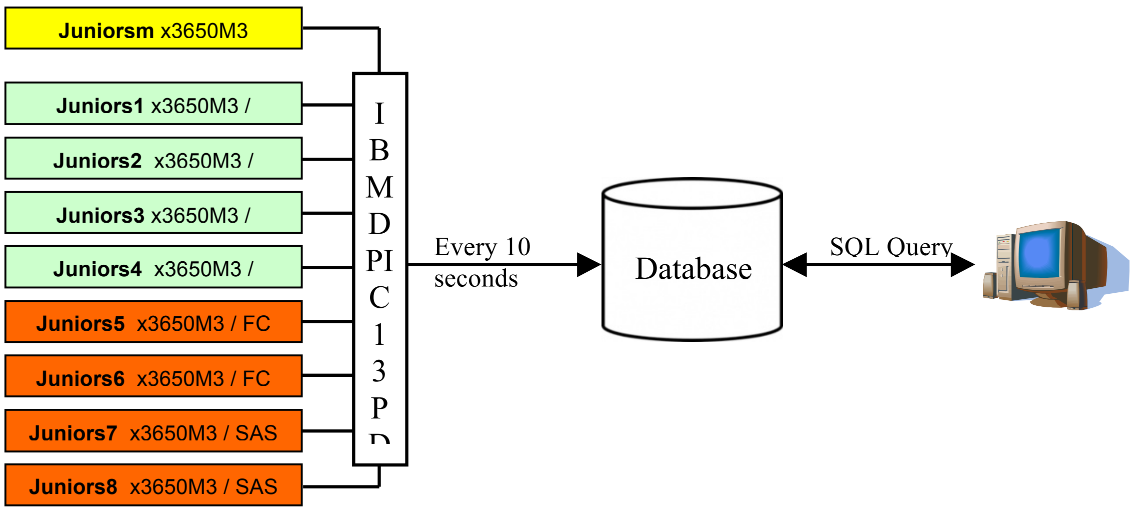


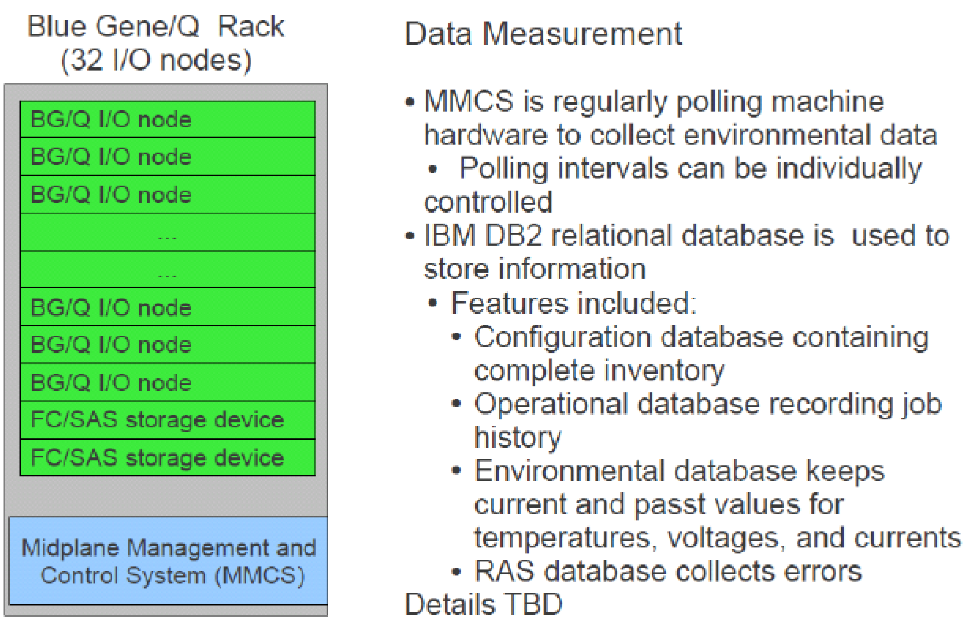
Figure 51 Schematic overview of the instrumentation of the x86 prototype.

The x86-based prototype has eight storage servers, each with eight Fusion-io cards and two FC/SAS attached storage devices as shown in Figure 51. Each node is separately connected to an IBM DPI C13 PDU enabling readout of the power data of a single node. This PDU is connected over an Ethernet connection to the local environment. A Perl script reads out the data of the PDU every 10 seconds and stores it in a database. The required data can be queried via an SQL script.

Number	Device	Scope/Purpose	Measurand	Error	Sampling

1	Raritan PDU DPXS12A-16	Measurement of power consumption for rack devices	Voltage(V)
			Active Power(W)
			Current(A)
					Interval 3 sec

Table 15 Properties of the Exascale integrated I/O subsytem instrumentation.



Data Measurement

- MMCS is regularly polling machine hardware to collect environmental data
  - Polling intervals can be individually controlled
- IBM DB2 relational database is used to store information
  - Features included:
    - Configuration database containing complete inventory
    - Operational database recording job history
    - Environmental database keeps current and passt values for temperatures, voltages, and currents
    - RAS database collects errors

Details TBD

Figure 52 Schematic overview of the instrumentation of the MPP prototype.

The instrumentation of the MPP BG/Q prototype, with 32 I/O nodes with flash drives and two FC/SAS attached storage devices, is shown in Figure 52. The specification of the intelligent PDU to measure the x86-based prototype is given in Table 15.

The database is a SQL database and contains the data described in Figure 53.

```
mysql> describe powertable;
+-------------+------------+------+-----+-------------------+-----------------------------+
| Field       | Type       | Null | Key | Default           | Extra                       |
+-------------+------------+------+-----+-------------------+-----------------------------+
| time_stamp  | timestamp  | NO   |     | CURRENT_TIMESTAMP | on update CURRENT_TIMESTAMP |
| pdu_nr      | int(1)     | YES  |     | NULL              |                             |
| load_group  | int(1)     | YES  |     | NULL              |                             |
| voltage     | varchar(9) | YES  |     | NULL              |                             |
| current     | varchar(9) | YES  |     | NULL              |                             |
| powerfactor | varchar(9) | YES  |     | NULL              |                             |
| power       | varchar(9) | YES  |     | NULL              |                             |
| watts_hours | varchar(9) | YES  |     | NULL              |                             |
+-------------+------------+------+-----+-------------------+-----------------------------+
```

Figure 53 SQL table to store PDU power monitoring information.

An additional way to measure power is to use the built-in sensors of the Fusion-io devices. Numerous data can be read out with the **fio-status** command. This command optionally returns an XML file. We use this feature to measure the power drawn from the PCIe bus.

The six bulk power supplies of the MPP IO-racks are read out every five minutes and stored into a database. The database connection is done via a bgqdb2 script. It includes the collecting of the data described in Figure 54.

```
+---------------+------------------+----------------+---------------+--------+----------+
| Column name   | Data type schema | Data type name | Column Length | Scale  | Nulls    |
|---------------+------------------+----------------+---------------+--------+----------+
| LOCATION      | SYSIBM           | CHARACTER      |            10 |      0 | No       |
| TIME          | SYSIBM           | TIMESTAMP      |            10 |      0 | No       |
| INPUTVOLTAGE  | SYSIBM           | DOUBLE         |             8 |      0 | No       |
| INPUTCURRENT  | SYSIBM           | DOUBLE         |             8 |      0 | No       |
| OUTPUTVOLTAGE | SYSIBM           | DOUBLE         |             8 |      0 | No       |
| OUTPUTCURRENT | SYSIBM           | DOUBLE         |             8 |      0 | No       |
+---------------+------------------+----------------+---------------+--------+----------+
```

Figure 54 SQL table holding the BG/Q power measurements.

The database collection is configured to store the power consumption of the MPP IO-racks every two seconds.

4.3 Holistic approach to energy efficiency, LRZ

A schematic overview of the prototype's cooling and power distribution systems can be seen in Figure 55. The three measuring points important for the Energy-to-Solution evaluation are:

1. The sensor measuring the power consumption of the internal cooling infrastructure.
2. The sensors (PDUs) measuring the power consumption for each node and environmental condition inside each rack, e.g., temperature and humidity.
3. The sensors providing information concerning the RCL (Refrigeration Coolant Loop) evaporators.

The instrumentation, summarized in Table 16, enables measurement of compute power consumption at the node level. Power consumption on a rack or the entire system level can be achieved by summing up the node power consumption. Cooling power is measured for the entire system and thus cannot be attributed to single nodes.

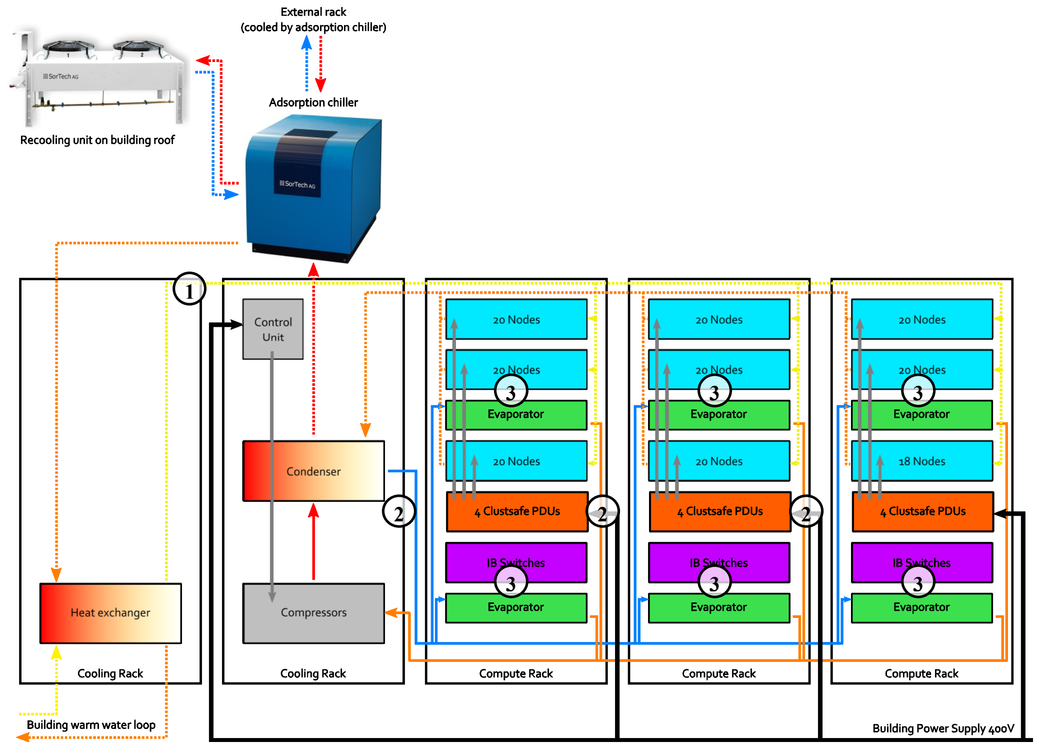


Figure 55 Schematic overview of the power distribution and cooling measurement system.

Number	Device	Scope/Purpose	Measurand	Error	Sampling
1	WBZ-80 + LAN TCP/IP module	Measurement of entire cooling power	Voltage (V)	< 2%	1/min
			Active Power (W)	-	
			Current (A)	< 2.5%	
2	Per rack 4x Megware Clustsafe PDUs	Power consumption per node & environment conditions	Voltage (V)	[3]	1/min
			Active Power (W)	[3]	
			Current (A)	[3]	
			Temperature (C)	[3]	
			Rel. Humidity (%)	[3]	
3	Per rack 2x KKU RCL Evaporator	Monitoring of cooling activity & measurement of environment conditions	Temp. inlet (C)	[3]	1/min
			Temp. outlet (C)	[3]	
			Rel. Humidity (%)	[3]	
			Fan speed (RPM)	[3]	

Table 16 CoolMUC internal sensor description.

[3] Sensor accuracy is still to be checked with device vendor

5 Benchmarks for Measurements

5.1 Overview Matrix

Table 17 gives an overview of what benchmarks focused on performance were run on which prototype for assessment of energy efficiency and traditional efficiency (fraction of peak capabilities). The first six benchmarks listed were selected as common computational benchmarks and the intention was that all computational prototypes carry out measurements, and to the extent time permitted optimized the code for the respective prototypes. The last three benchmarks were selected by individual parterners for their prototypes to highlight specific capabilities or to serve a clearly defined special purpose, for example the Dhrystone benchmark was used by BSC to validate the Tegra 2 performance against publicized results for the ARM cores, and the Mprime benchmark was used by LRZ as a simple "thermal virus" benchmark to generate a high heat load. The FPGA prototype (JKU) was limited to a single benchmark due to implementation complexity issues. The Numascale prototype (UiO) was delayed due to stability problems and only the STREAM benchmark could be run in time for this report. Unfortunate stability problems prevented a successful FFT run on the Tegra 2 prototype (BSC-1) and instrumentation problems limited the power and energy measurements available for the PSNC prototypes.

Table 18 gives an overview of what I/O benchmarks were carried out by which partner. The Hydro, IOR and SIONlib benchmarks were selected as common benchmarks for the I/O prototypes. HLRS kindly carried out benchmarks on the CEA/CINES prototype using their in-house developed metadata benchmark.

The Hydro application oriented benchmark is listed in both tables. It can be executed with the option to write snapshot files. This option was used by the I/O prototypes to evaluate application level performance while it was completely turned off on the computational prototypes.

Table 17 and Table 18 identify the prototypes by the partner site at which the prototype was installed. The same method is used in the abbreviations given Table 21 that were used in the graphs later in the report.

	STREAM	Dense matrix mult	Sparse MV	FFT	HPL	Hydro (Comp)	QCD	Dhrystone	Mprime
BSC-1	Y	Y	Y		Y	Y		Y	
CaSToRC	Y			Y	Y	Y	Y		
JKU		Y							
PSNC	Y	Y	Y	Y	Y	Y			
SNIC	Y	Y	Y	Y	Y	Y			
UiO	Y								
LRZ	Y	Y	Y	Y	Y	Y			Y

Table 17 Summarizes what computational benchmarks were carried out on the different prototypes.

	Hydro (I/O)	IOR	SIONlib	HLRS Metadata
CEA		Y	Y	Y
FZJ	Y	Y	Y	

Table 18 Summarizes what I/O benchmarks were carried out on the different prototypes.

5.2 Brief benchmark descriptions

5.2.1 *STREAM*

The STREAM benchmark [STREAM-1] includes four simple operations that stress the coherent memory system using strided access. The operations are:

COPY:	a(i)=b(i),
SCALE:	a(i)=q*b(i),
SUM:	a(i)=b(i)+c(i), and
TRIAD:	a(i)=b(i)+q*c(i).

There is no data reuse, and in case of SUM and TRIAD 24 bytes of data are involved for each execution of the function. Hence, STREAM is a good memory system benchmark for applications dominated by regular memory accesses.

STREAM dates back to a time when floating-point arithmetic was comparable in cost to memory accesses, so that the copy test was significantly faster than the others. This is no longer the case on any machines of interest for HPC, and the four STREAM bandwidth values are typically quite close to each other. It should be noted that in the memory bandwidth estimates, the STREAM benchmark gives “credit’’ for both memory reads and memory writes. On machines with a write-allocate cache policy each write operation will result in at least one additional 8-byte read per write in order to load the line containing the output into the cache. This data traffic is superfluous to the specified calculation and not counted.

5.2.2 *Dense matrix multiplication*

Dense matrix multiplication is a frequently occurring operation in many scientific and engineering codes. In some applications, large numbers of small to modest size independent matrix multiplications are carried out while in others a few (very) large matrix multiplications may be needed. Matrix multiplication is the dominating operation in well-designed codes for the HPL [Linpack] benchmark. Multiplication of nxn matrices requires O(n^3) operations for O(n^2) data. Good implementations exploit the opportunity for O(n) data reuse. Such implementations for many architectures tend to be limited by the CPU processing capability (as opposed to the memory or communication system).

By often “stressing” the CPU, matrix multiplication dissipates close to maximum dynamic power for many server designs and is a good power stress test. Matrix multiplication is a level 3 Basic Linear Algebra Subroutine (BLAS) function [BLAS] and is included in many mathematical subroutine libraries, such as AMDs ACML [ACML], IBMs ESSL [ESSL], and Intels MKL [MKL] and the EuroBen [EuroBen] benchmark suite as routine mod2am. Dense matrix multiplication is also included in the HPC Challenge [HPCC] benchmark suite both in the form of large numbers of independent matrix multiplications and single large matrix multiplications. The mod2am was chosen as one of the common benchmarks.

5.2.3 *Sparse matrix-vector multiplication*

Sparse matrix-vector multiplication is another important benchmark for scientific and engineering applications. Sparse matrix-vector multiplication appears in many codes using iterative solvers for partial differential equations based on irregular domain discretization, as well as in optimization and clustering problems. Memory references are often highly irregular making sparse matrix-vector multiplication very sensitive to memory and communication system capabilities. Indirect addressing makes sparse matrix-vector multiplication typically dominated by integer operations rather than floating-point operations. Sparse matrix multiplication is included in libraries such as, e.g., Sparsepack [SPARSEPACK], and in the EuroBen benchmark package as the routine mod2as. The latter was chosen as one of the common benchmarks.

5.2.4 *FFT*

The Fast Fourier Transform (FFT) is one of the most widely used computational kernels in science and engineering applications as well as in many other fields. By being very computationally efficient with an arithmetic operations count of O($n$log$n$) for O(n) data, data reuse is modest (O(log$n$)). Furthermore, the data access pattern uses a wide range of power-of-two strides stressing many memory and communication systems. Power-of-two strides result in poor behaviors of common cache associativites and replacement policies. FFT is part of many vendor libraries, such as AMDs ACML, IBMs ESSL and Intel's MKL as well as the basis for specialized packages, such as FFTpack [FFTPACK] and FFTW [FFTW]. FFT is also included in the EuroBen benchmark package as routine mod2f and the HPC Challenge suite. The mod2f code was selected as one of the common benchmarks.

5.2.5 *Linpack*

The HPL [Linpack] Benchmark is commonly used to assess a computer's floating-point rate of execution. It is the basis for the Top500 and Green500 lists of the 500 known most powerful computers in the world and their energy efficiency (for this particular benchmark). It solves a dense linear system of equations, $Ax = b$, in double-precision floating-point arithmetic. HPL is a portable implementation written in C. Regardless of the algorithm used, the operation count used in determining the execution rate is $2/3n^3 + 2n^2$ with n being the number of rows and columns of the matrix A. With O(n^3) operations and O(n^2) data, the data reuse is O(n) on average. Well-designed codes exploit this fact, which implies that such codes are primarily sensitive to CPU clock speed and less sensitive to memory and communication system capabilities.

Since for traditional x86-based servers the CPU is dominating the power consumption the HPL benchmark is often used as a power stress test since for a typical x86 server the computations can keep the CPU busy at about, or in excess of, 90% causing close to maximum dynamic power dissipation. Most of the computations for well-designed codes are carried out as dense matrix multiplications that in many cases have a slightly higher efficiency and hence power consumption than HPL.

5.2.6 *Hydro*

Hydro is a simplified version of the RAMSES [RAMSES] astrophysics code for the study of large-scale structure and galaxy formation. RAMSES is written in FORTRAN with extensive use of the MPI library. Unlike the previously described computational kernel benchmarks Hydro attempts to capture the behaviour of an important class of application codes in a modest number of lines of code (about 1500 lines). The Hydro benchmark has been developed by CEA, the US Department of Energy Los Alamos National Laboratory (LANL) and National Nuclear Security Administration (NNSA) and others. It includes significant use of square-roots and divisions which are relatively time consuming operations compared to addition and multiplication on many architectures. Hydro is well known in the Computational Fluid Dynamics community and has proven to be scalable to tens of thousands of processes. In order to cope with a wide variety of architectures and software environments two main branches of the Hydro code have been developed [Hydro]:

- A FORTRAN branch including OpenMP [OpenMP], MPI [MPI], and hybrid MPI/OpenMP [Hybrid],
- A C branch facilitating porting to GPU platforms, including CUDA, OpenCL, and the HMPP programming frameworks, and novel HPC languages including UPC [UPC-1, UPC-2].

All versions have been developed trying to keep up the following rules:

- No algorithm modification.
- No code structure modification.

The versions have been validated by comparison with the sequential version.

The space domain for Hydro is a rectangular two-dimensional splitting with a regular Cartesian mesh (there is no Adaptive Mesh Refinement). Hydro solves compressible Euler equations of hydrodynamics, using a finite volume numerical method using a second order Godunov scheme [Godunov] for the Euler equations, and a Riemann solver [ROSE] computes numerical flux at the interface of two neighbouring computational cells.

5.2.7 *QCD*

Lattice QCD calculations aim to understand the physical phenomena encompassed by Quantum ChromoDynamics (QCD), the fundamental theory of the strong forces of subatomic physics, and to make precise calculations of the theory’s predictions. These simulations are necessary to solve fundamental problems in nuclear and high-energy physics. The QCD community has long been a large consumer of supercomputer resources and has even been designing and building their own dedicated computer systems, such as for instance the APE, QCDSP, QCDOC and QCD-PAX machines. The PRACE eQPACE Preparatory Phase prototype was in part motivated by the computational needs of QCD and in 2012 a 1216 core 400 GPU cluster was installed in Bielefeld, Germany for QCD studies.

The progress of the field is largely viewed as being limited by available computing power. A recent USQCD report [QCD-2013] estimated the needs for a few sub-problems to be in the hundreds to thousands of TFlops/s years (TFlops/s sustained for a year). Because of this need and the importance of QCD in fundamental physics, QCD is clearly a good candidate application for benchmarking. QCD computations have elements of both capability and capacity computing. Because of the large computational needs the most compute intensive parts of QCD computations tend to be highly optimized for different architectures and it was that no single code would be appropriate for all prototypes and that code adaptation would be required including possible choice of alternate algorithms for different prototypes. With the resource limitations for code adaptation QCD was only evaluated for the GPU prototype for which an optimized code existed, the QUDA code [QUDA].

5.2.8 *IOR*

IOR [IOR] is a synthetic I/O benchmark that allows generating a parallel I/O workload. It is highly configurable and supports various interfaces and access patterns and thus allows mimicking the I/O workload of different applications. The suggested parameter space is listed in Table 19.

Parameter	Purpose	Parametric space
-a <api>	API for I/O	POSIX, MPIIO, HDF5
-B	By-pass I/O buffers (API=POSIX only)	
-b <blk_size>	Continuous bytes to write per task	2 TBytes/$N_{clients}$
-C	Reorder tasks by constant for write/read	
-E	Do not remove test file before write	
-F <task local>	Shared-file (0) or task local (1)	0,1
-i <nrep>	Number of times to run each test	3
-N 0	All tasks should participate	
-o <test_file>	Name of the output file	
-q <offset>	taskPerNodeOffset for read tests	To be tuned to minimize caching effects
-r	Read existing file(s)	
-T 0	Max. Time in minutes to run the tests	
-t <tfr_size>	Number of bytes transferred per call	1 MB, 4 MB
-w	Write files	

Table 19 Parameter space used for the IOR benchmark.

5.2.9 *SIONlib*

SIONlib [FZJ] is a scalable I/O library for parallel I/O which is developed in Jülich. It maps task-local file access to one or a couple of shared files in the parallel file system. The benchmark *partest*, that is part of the library, can generate any kind of application I/O pattern using different numbers of shared files to simulate the (optimal) usage of the SIONlib. It also allows creating task local file I/O for comparison. All tests are repeated 3 times (organised within JuBE) using the following number of $N_{clients}$: 128, 512, 2048, 8192. The suggested other parameters are listed in Table 20.

Parameter	Purpose	Parametric space
-T <tasklocal>	Shared file (0) or task local file (1)	0,1
-n <files>	Number of physical files	To be tuned for performance on given prototype
-L	POSIX call (true) – ANSI calls (false)	Switch on/off
-q -1	Automatic FS block size alignment	
-Z <offset>	Read task offset	To be tuned to minimize caching effects
-b <bufsize>	Buffer size per client per write	To be optimized for given prototype (≥ file system block size)
-s <localsize>	File I/O (size) per client	2 TB/$N_{clients}$

Table 20 Suggested parameter space for the SIONlib benchmark.

5.2.1 *HLRS Metadata Benchmark*

The prototypes were benchmarked with a tool developed by HLRS for performance analysis of the I/O subsystem of a Unix environment. The tool is capable of analysing performance of parallel metadata accesses (create, stat, delete) to the I/O subystems. In a nutshell, the benchmark works as follows: For a specified number of nodes, a number of processes is defined, where each process creates 10 000 files with the size 0 B, calls a *stat* and then removes the files again. The benchmark measures the number of file operations per second (average) for create, stat and delete of 10 000 files in a single directory. The processes are distributed across nodes and hence parallel I/O can be measured.

5.2.2 *Dhrystone*

The Dhrystone [Dhry] benchmark consists of standard code and concentrates on string handling. It uses no floating-point operations. It is heavily influenced by hardware and software design, compiler and linker options, code optimizing, cache memory and integer data types.

5.2.3 *Mprime*

The pre-compiled 64-bit mprime v2.26 benchmark [MPrime] used for temperature test on CooLMUC is freely available. The program is flexible in terms of floating-point computation and main memory access pattern, performing Fast Fourier Transforms in a repeated fashion to search for Mersenne prime numbers. Due to its heavy load on the system it has become one of the favoured system stability tests among private computer users. The “torture test” was used with custom settings to stress the server processors, while main memory access is mild and no communication via network devices is performed.

6 Measurement results

In this section we report the results of the benchmarking of the constructed or acquired prototypes. For each benchmark we first report the benchmark results for the prototypes individually and provide relevant comments to the results, then summarize the results for the benchmark on the prototypes on which it was executed. In the summary section for each benchmark the results for different prototypes are plotted together on the same graph for ease of comparison. Due to the diverse nature of the prototypes and the differences in scales, caution should be employed when comparing these results side by side.

For each benchmark the first plot shows the absolute performance obtained on the respective prototype for varying problem size. In general, the presented sweeps use a weak-scaling approach. These graphs make the relative scales of the prototypes apparent and cannot directly be used to draw conclusions about the merits of the architectures for HPC applications. For such conclusions a "normalization" is required.

The graphs showing efficiency (fraction of peak) and energy efficiency do have a normalization in regards to scale and hence are better suited to make comparison of the relative merits of different architectures for HPC as judged by the benchmark. However, even for these graphs caution must be exercised since there is no uniformity in the level of optimization carried out for the results. Even for architectures with very mature software environments efficiencies can vary by more than an order of magnitude depending on software/algorithm optimization and attention to data layout.

Throughout this section the abbreviations in Table 21 are used to refer to the different prototype installations.

Abbreviation	Prototype Name		Site
BSC-1	An NVIDIA Tegra 2 mobile SoC based HPC cluster		BSC
CaSToRC	GPU-GPU communication over PCIe and IB		CaSToRC
CEA	Exascale I/O		CEA/CINES
FZJ	Exascale integrated I/O subsystem		FZJ
JKU	FPGA matrix computation acceleration		JKU
LRZ	Holistic approach to energy efficiency		LRZ
PSNC-BR	On die integrated CPU and GPU	AMD E-350	PSNC
PSNC-IB		Intel i7-3770	
PSNC-LL		AMD A8-3870	
PSNC-SB		Intel i7-2600	
PSNC-TR		AMD A10-5800K	
PSNC-TR-G		AMD A10-5800K GPU	
SNIC	DSP based node for HPC		SNIC/KTH
UiO	Shared memory through a cache-coherency add-in card (NUMA-CIC)		UiO

Table 21 Abbreviations for the prototypes used in the graphs and the report.

6.1 Node assessments

6.1.1 *STREAM*

As described in Section 5, STREAM is a very simple benchmark that is commonly used to assess memory system performance for strided access. Despite its simplicity it is quite sensitive to compiler optimizations, not all of which might be controllable through compiler optimization flags. For instance, on one of the Preparatory Phase prototypes scale yielded better performance than copy because the compiler did generate vector instructions for scale but not for copy, with the consequence that a copy implemented as a scale with a scaling constant of 1 was faster than copy. STREAM can also be quite sensitive to the data layout in memory since DRAMs do not offer uniform memory access times but performance depends on column, row and bank access orders. The sensitivity to these DRAM properties and the data layout depend on the number of memory channels and memory ranks per channel. The latter may also affect the clock rate by which the memory is operating, i.e., it can cause memory to be “clocked down” compared to its stated frequency. Finally we reiterate that for write-allocate caches the load required for the write is not included in the STREAM reported bandwidth, i.e. for such cache policies STREAM reported bandwidths close to 100% of peak cannot be achieved.

An NVIDIA Tegra 2 mobile SoC based HPC cluster, BSC

The results of the performance measurements of the Tegra 2 node are shown in Figure 56 for varying array sizes and with one and two threads per node. The maximum problem size is N=33 554 432 with an array size of 768 MB. As expected, the single thread performance is best for sizes that fit in the L1 cache, then for the ones that fit in the L2 cache. For large arrays the performance for two threads is better than that of a single thread due to architectural limitations on the ARMv7 cores. GNU/Linux OS and a GCC 4.6 compiler were used for the measurements.

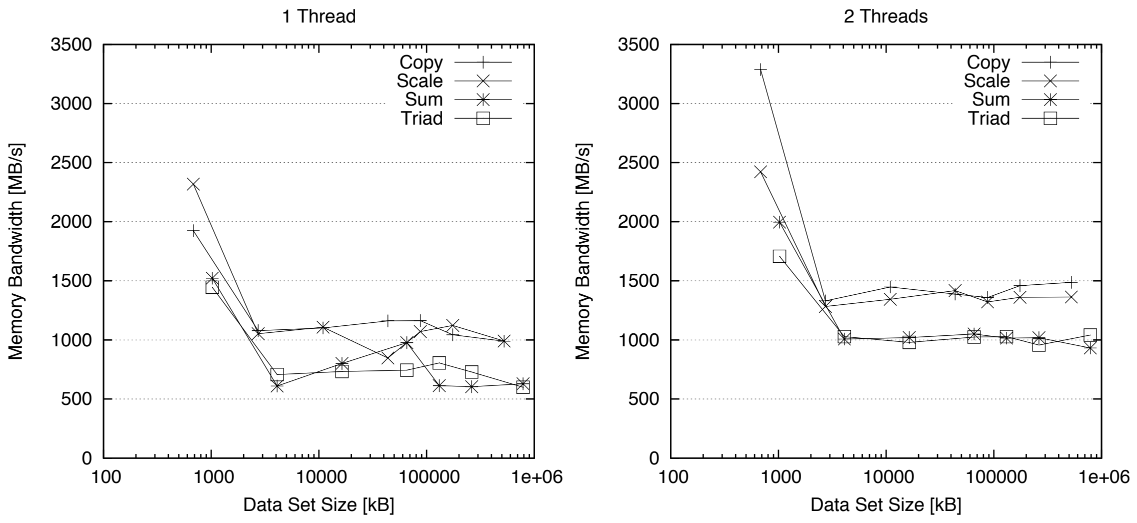


Figure 56 Tegra 2 node STREAM performance for 1 and 2 threads. With the DDR2-667 MHz 32-bit memory the theoretical peak memory bandwidth is 2.66 GB/s.

For the energy efficiency assessment, the node power was measured at the AC socket for all cases. The energy was determined by integrating the power measurements. The energy consumption for all four STREAM benchmarks was the same.

GPU-GPU communication over PCIe and Infiniband, CaSToRC

For the STREAM benchmark a CUDA implementation modified to support benchmarking of multi-GPU nodes and systems was used. Three types of assessments were made:

1. "Pure" intranode peer-to-peer communication within a single host process.
2. Intranode/internode MPI communications using CUDA IPC.
3. Standard MPI communications that served as a reference implementation. (CUDA-IPC is not supported for inter-node peer-to-peer communications for the GPUs used in this assessment.)

Measurements were carried out for both double precision and single precision data types. The first two cases above were carried out using NVIDIA's Unified Virtual Addressing (UVA) technology. An advantage of this technology is that the programmer does not need to keep account of which memory space a given pointer belongs to, and also does not need to explicitly indicate the direction of memory copies (e.g. device-to-host, host-to-device, or device-to-device). The only consideration is that the host memory must be page-locked, which may degrade overall system performance if too much memory is allocated. Due to its simplicity, it is quite straightforward to incorporate UVA in existing MPI-CUDA applications.

Figure 57 shows the STREAM performance results for the three cases studied. For the local GPU memory case the maximum measured bandwidth as about 73% of the 150 GB/s peak, whereas for the case with two GPUs on separate PCIe buses communicating via the PCI switch in the node the maximum measured bandwidth was about 56% of the 16 GB/s PCIe peak bandwidth. The measured peak bandwidth was approximately the same for all four STREAM benchmarks. For the case involving MPI communication over QDR Infiniband the maximum measured bandwidth was about 50% of the 4 GB/s peak QDR bandwidth, with the *sum* and *triad* cases being somewhat lower. Note that the GPUs do not access their GDDR memory though cache so STREAM reports actual memory operations.

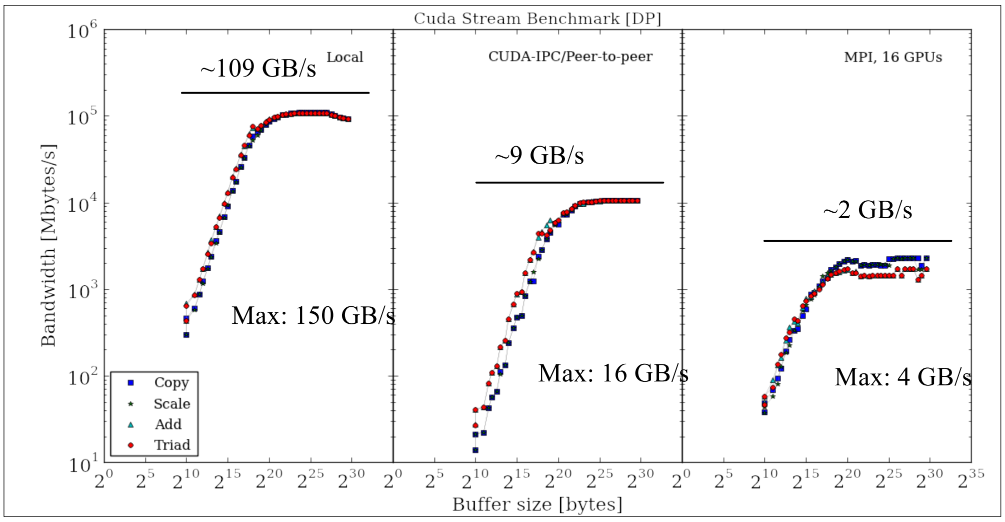


Figure 57 STREAM performance results using NVIDIA's Unified Virtual Addressing technology for local GPU memory access (left), GPU-to-GPU memory access through the PCIe buses and switch on the same node (center), and GPU-to-GPU memory access via PCIe buses and host memories (right).

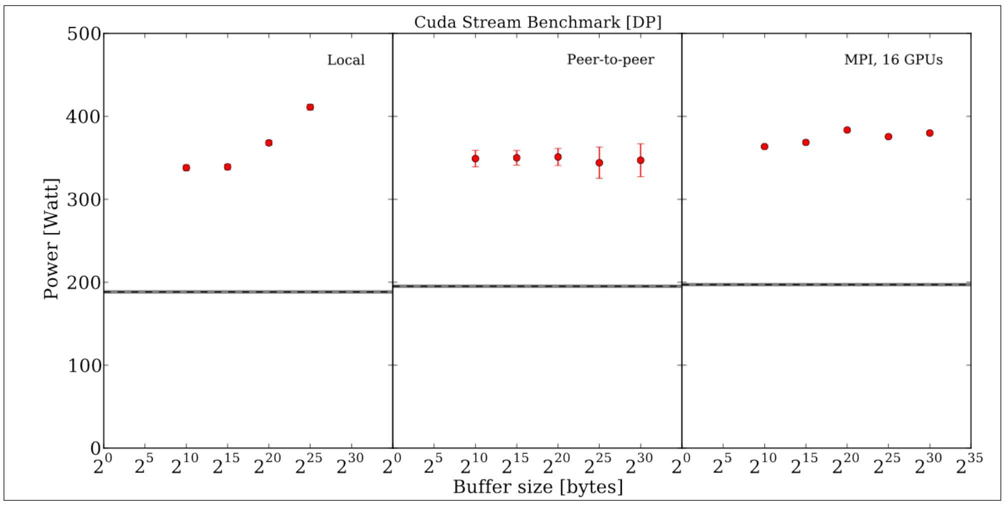


Figure 58 Active power consumption for the local, intra-node and inter-node STREAM benchmarks as a function of array size for local GPU memory access (left), GPU-to-GPU memory access through the PCIe buses and switch on the same node (center), and GPU-to-GPU memory access via PCIe buses and host memories (right).

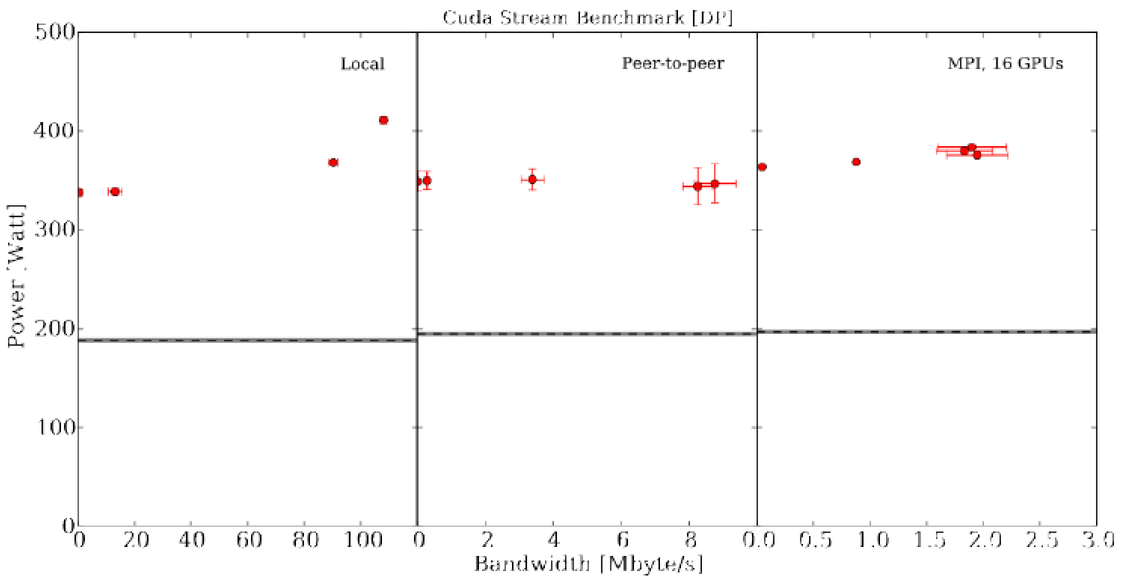


Figure 59 Power consumption for the STREAM benchmark as a function of measured bandwidth for local GPU memory access (left), GPU-to-GPU memory access through the PCIe buses and switch on the same node (center), and GPU-to-GPU memory access via PCIe buses and host memories (right)

Figure 58 shows the measured active power consumption as a function of the array sizes and Figure 59 the relationship between the power consumption and measured bandwidths. Power is measured as the AC power for an entire node.

As can be seen in Figure 59, the power consumption is approximately the same for the local, intra-node and inter-node cases, with a slightly higher active power for the inter-node case. For the local GPU case the active power consumption at the maximum observed data rate is about 30% higher than at a low data rate.

On die integrated CPU and GPU, PSNC

The AMD Brazos x86 core STREAM results are shown in Figure 60. Only close to 25% of the peak DDR3-1066 memory bandwidth of 8.5 GB/s is realized. However, the Brazos last level cache is a write-allocate cache and thus the actual bandwidth to the DDR3 for *copy* and *scale* is 50% higher than what is accounted for by the STREAM benchmark. The significantly lower performance of *scale* compared to *copy* is most likely because of the multiply unit of the Brazos only accepting arguments every other cycle. The impact of this is less significant for the *sum* and *triad* tests since those have more memory operations for each arithmetic operation.

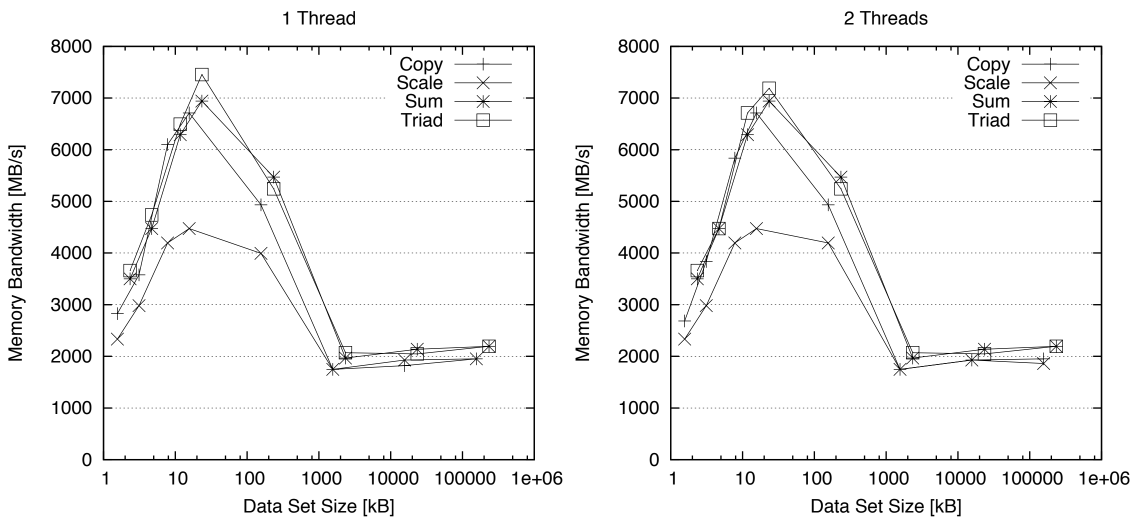


Figure 60 AMD Brazos APU STREAM performance for 1 and 2 threads using OpenMP.

DSP based node for HPC, SNIC

The maximum bandwidths observed for *copy* and *scale* are shown in Table 22 and Table 23. For the L1 and L2 SRAM the peak measured bandwidths are quite close to theoretical peak: 124.2 of 128 GB/s (97%) for L1, and 48 of 64 GB/s (75%) for L2. For the results in Table 22 and Table 23 the DSP was operated at 1 GHz. It is interesting to note that the core power consumption decreases as array size increases to a point where DDR3 gets engaged, most likely because DRR3 access becomes a bottleneck and the core activity decreases somewhat. On the other hand, the DDR3 power consumption increases by a factor of 4.5 resulting in an increase in the total power consumed. In comparing the *copy* and *scale* we see a noticeable increase in power consumption for *scale* compared to *copy* when the data set is sufficiently small to fit in the L1 SRAM in which case the multiplication unit can be fully engaged.

Copy	BW (GB/s)	Core (W)	Memory (W)	Total (W)	Core (GB/J)	DSP (GB/J)
L1	124.2	6.81	0.42	14.20	18.24	17.19
L2	47.6	6.54	0.42	13.85	7.27	6.84
MCSM	38.7	6.65	0.42	13.99	5.81	5.47
DDR3-1333	3.0	5.59	1.95	14.47	0.54	0.40

Table 22 STREAM *copy* performance and energy efficiency on the DSP prototype.

Scale	BW (GB/s)	Core (W)	Memory (W)	Total (W)	Core (GB/J)	DSP (GB/J)
L1	123.7	7.17	0.42	14.69	17.23	16.28
L2	41.8	6.70	0.42	14.05	6.24	5.50
MCSM	21.9	6.44	0.42	13.71	3.40	3.19
DDR3-1333	2.9	5.60	1.91	14.44	0.52	0.87

Table 23 STREAM *scale* performance and energy efficiency for the DSP prototype.

Shared memory through a cache-coherency add-in card (NUMA-CIC), UiO

This prototype is using NumaConnect cards to realize a global shared memory for HyperTransport (HT) based servers. STREAM tests were carried out on a configuration with 23 dual socket 12-core Opteron servers with DDR3-1333 memory, Figure 61, Figure 62, and Figure 63. Thus, each server has a peak memory bandwidth of 85.3 GB/s and the 23-server configuration a peak bandwidth of 1961.9 GB/sec. The HT 3.1 has a bidirectional bandwidth of 51.6 GB/s so the total bandwidth from the NumaConnect cards to the 23 servers is 1168.8 GB/s.

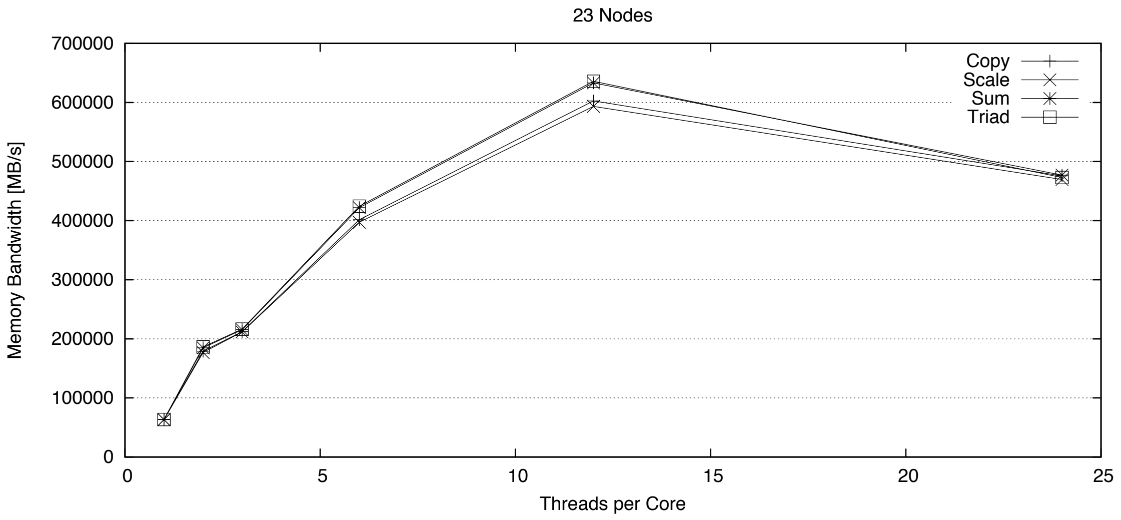


Figure 61 STREAM performance vs. number of threads per node on the Numa-CIC prototype.

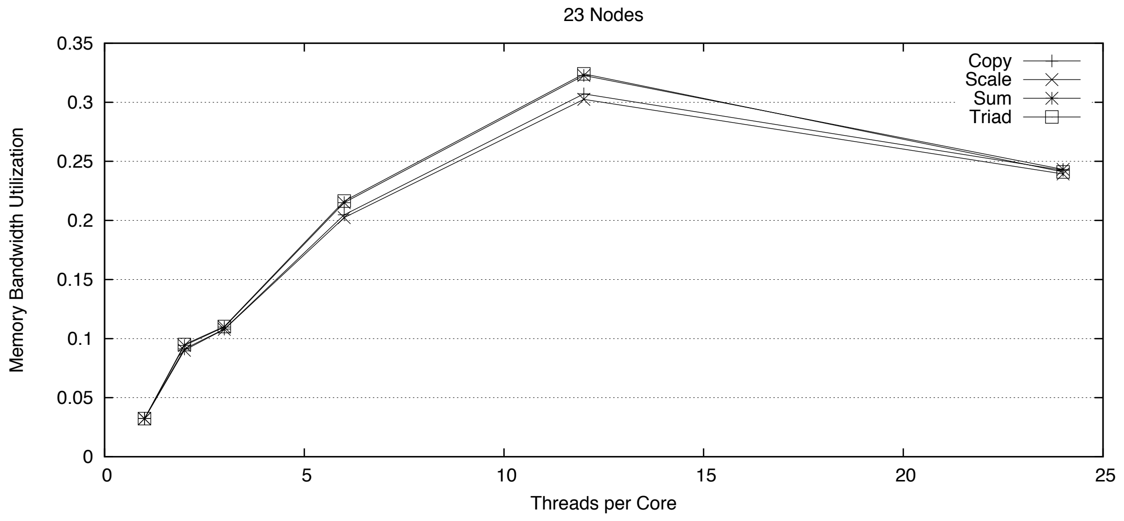


Figure 62 Efficiency of the STREAM benchmark vs. number of threads per node for the NUMA-CIC prototype.

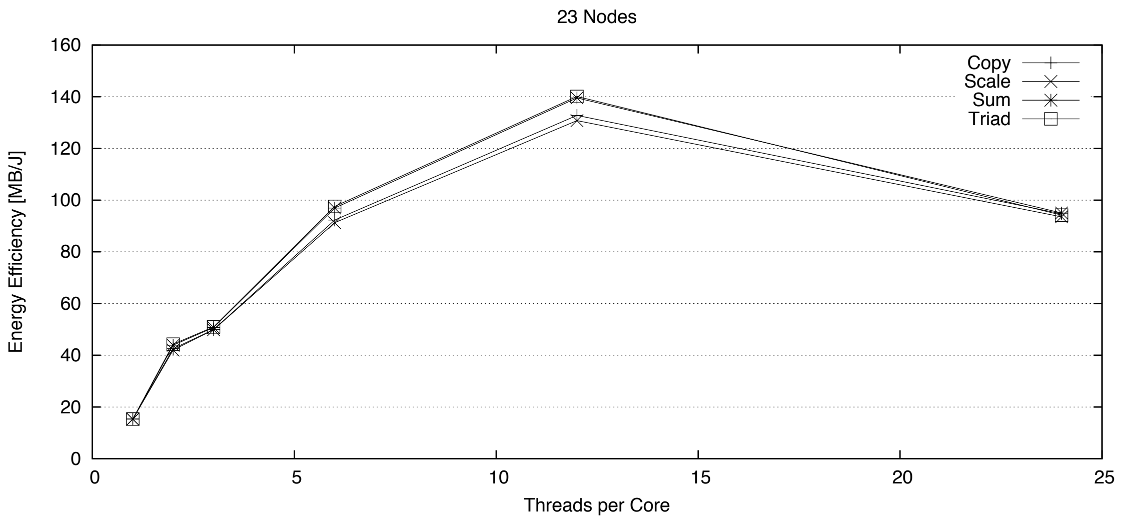


Figure 63 Energy efficiency obtained by the STREAM benchmark for varying number of threads per node for the NUMA-CIC prototype.

STREAM Summary

All of the prototypes, except the Tegra 2 prototype use DDR3 memory operating at either 1066 or 1333 MHz, resulting in a memory bandwidth per channel of 8.53 and 10.67 GB/s, respectively. The Tegra 2 prototype (BSC-1) uses a single 32-bit memory channel for DDR2-667 memory and thus has a peak of 2.66 GB/s. The DSP prototype (SNIC) has a single 64-bit memory channel, whereas the AMD Magny-Cours (LRZ/UiO) has four memory channels and the Intel Sandy Bridge (PSNC-SB) and Ivy Bridge (PSNC-IB) two memory channels per CPU. Table 24 summarizes the peak single node performance results, except for the prototype using NumaConnect (UiO) to implement a distributed shared memory. For this prototype the Table entries represent the results for a 23-node system. Due to problems with the power measurement equipment at PSNC reliable energy measurements could not be obtained in time for this report.

The significantly higher peak memory bandwidth of the GPUs (CaSToRC) than the other prototypes is apparent from the results as clearly seen in Figure 64. However, the realized fraction of peak memory bandwidth does not vary widely across the prototypes, as seen in Figure 65. In fact, in this regard the GPU results are not particularly good since for the GPU case STREAM reports all memory operations, whereas for the other prototypes due to write-allocate cache policies STREAM under-reports memory operations. Given this fact the fraction of peak memory bandwidth realized by the CPU-based prototypes is comparable or in some cases higher than that of the GPUs.

Prototype	**Bandwidth [MB/s]**				**Energy Efficiency [MB/J]**			
	Copy	Scale	Sum	Triad	Copy	Scale	Sum	Triad
BSC-1	1487	1363	932	1041	155.5	138.4	94.3	105.3
CaSToRC	88530	88829	89283	89281	269.5	270.5	271.8	271.8
LRZ	49627	35458	39036	39010	228.5	163.2	179.7	179.6
PSNC-BR	1950	1862	2194	2194				
SNIC	3028	2910	2979	3642	398.9	384.8	403.8	480.5
UiO	*602625*	*593412*	*633213*	*635794*	*132.8*	*130.8*	*139.5*	*140.1*

Table 24 Memory bandwidth and energy efficiency obtained by the STREAM benchmark kernels for the largest reported vector sizes. PSNC power measurement data is excluded due to instrumentation problems, see also section 8.2.5.

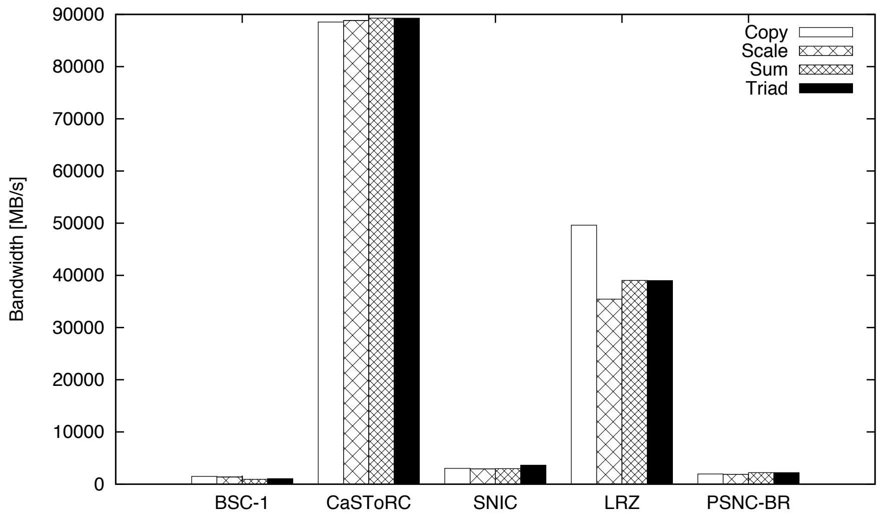


Figure 64 Memory bandwidth obtained using the STREAM benchmark kernels shown for the largest vector sizes reported.

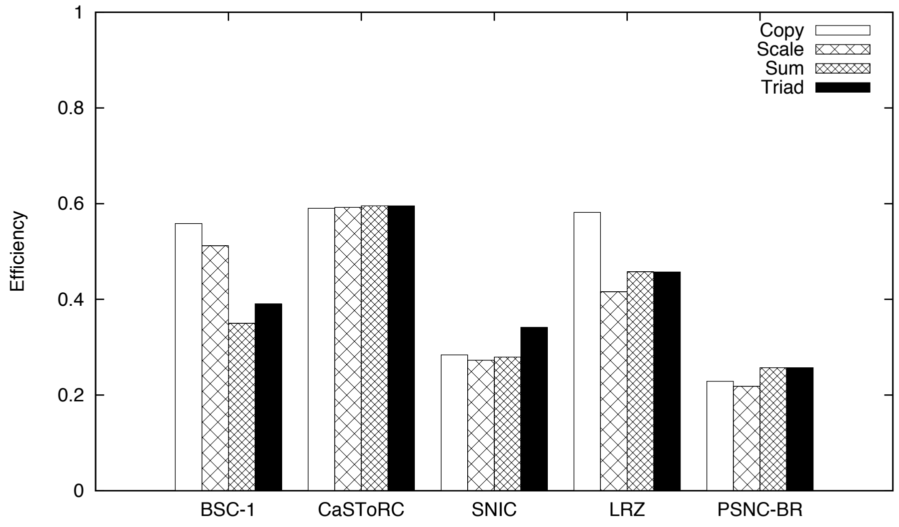


Figure 65 Fraction of peak memory bandwidth achieved by STREAM kernels

The energy efficiency, see Figure 66, of the DSP prototype (SNIC) is about 50% higher than that of the GPU-based prototype (CaSToRC) despite the lower fraction of peak memory bandwidth realized. It is also interesting to note that despite the comparatively high fraction of peak bandwidth achieved for the Tegra 2 prototype (BSC), its energy efficiency for STREAM is poor in comparison with both a one-generation-old standard x86 CPU (LRZ), GPUs and the DSP prototype.

The following graphs summarize the results for performance measured as B/s, efficiency measured as fraction of peak, and energy efficiency measured as B/J for several of the prototypes. The left plot in a pair shows results for total data set sizes that fit in the last level cache and the right plot shows results for larger total data sizes. The total data set size is the sum of the array sizes of the arrays used in the respective benchmarks.

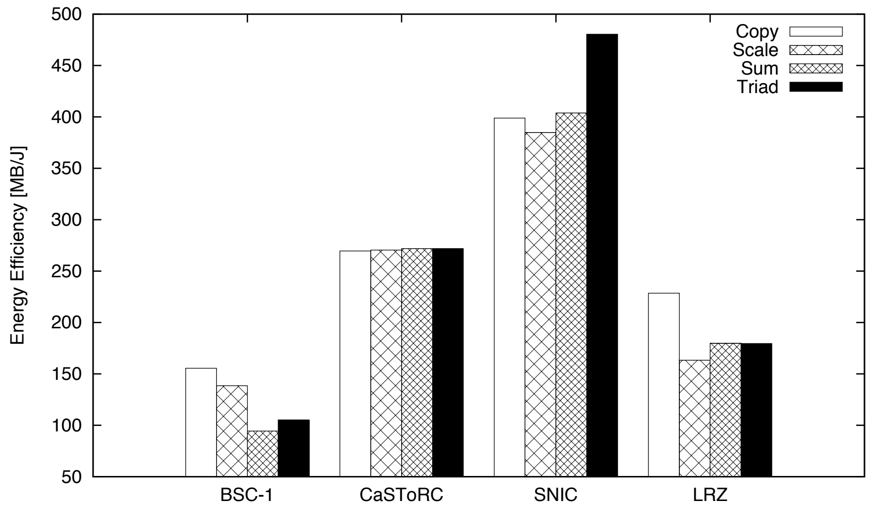


Figure 66 Energy efficiency obtained by the STREAM benchmark kernels for the largest reported vector sizes.

The cache effects for the DSP (SNIC) and Magny-Cours (LRZ) prototypes are clearly visible in Figure 67 and so are the significantly higher memory bandwidths of the x86+GPU (CaSToRC) and Magny-Cours prototypes. The efficiency of the x86+GPU and Mangy-Cours prototypes is fairly comparable, see Figure 68. For the DSP prototype, a higher efficiency should be achievable, but improved benchmark results were not available in time for this report. Despite the DSP prototype's relatively poor efficiency, its energy efficiency is about 30% higher than that of the x86+GPU, twice that of the Magny-Cours prototype and about three times higher than that of the Tegra 2 prototype, Figure 69. Figure 70 – Figure 78 show the corresponding results for the STREAM *scale*, *sum* and *triad* functions.

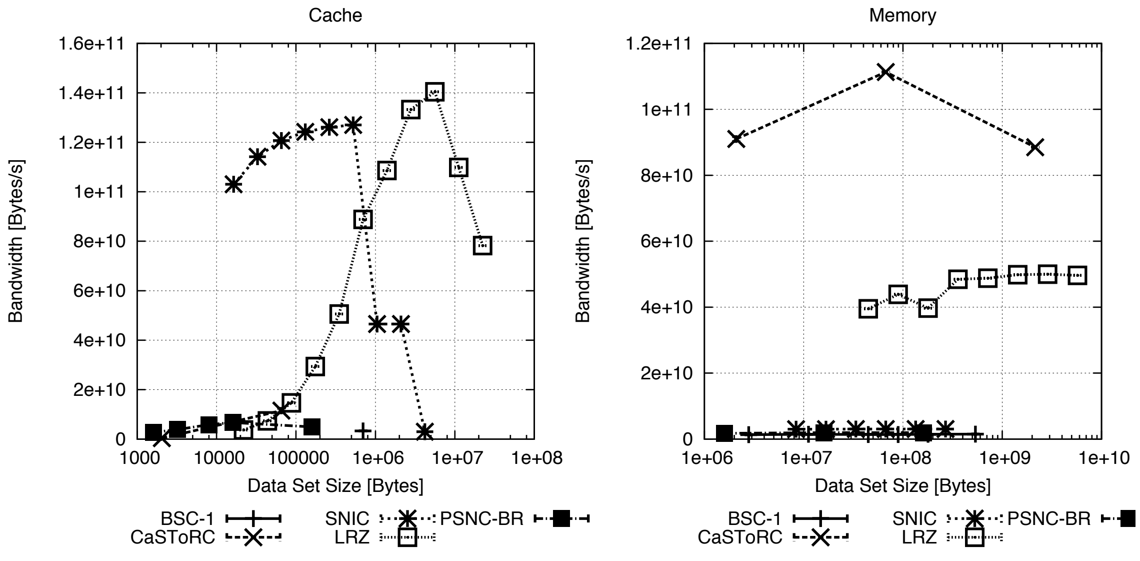


Figure 67 STREAM *copy* performance measured as B/s with the cache effects clearly visible for the DSP (SNIC) and Magny-Cours (LRZ) prototypes and the significantly higher memory bandwidths of the x86+GPU and Magny-Cours prototypes.

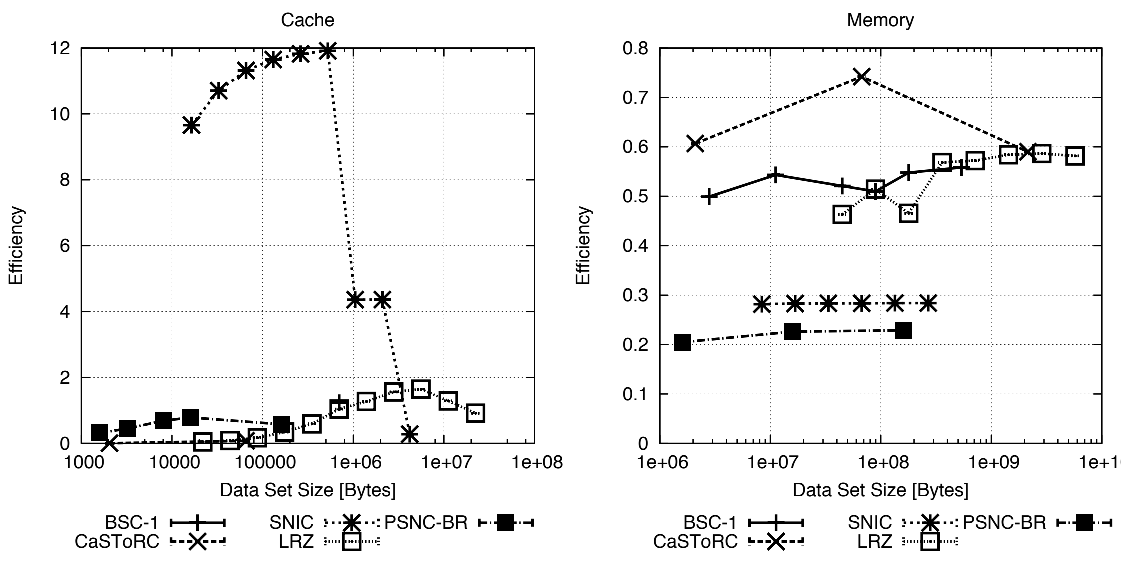


Figure 68 Efficiency of STREAM *copy* expressed as fraction of peak memory bandwidth. The cache memory bandwidth amplification is apparent.

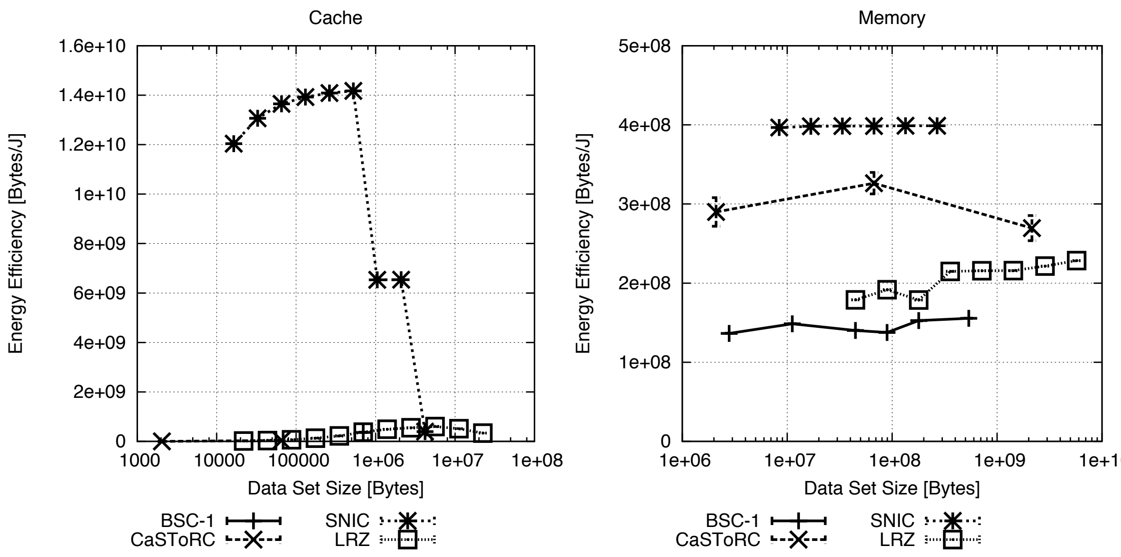


Figure 69 STREAM *copy* energy efficiency measured as B/J.

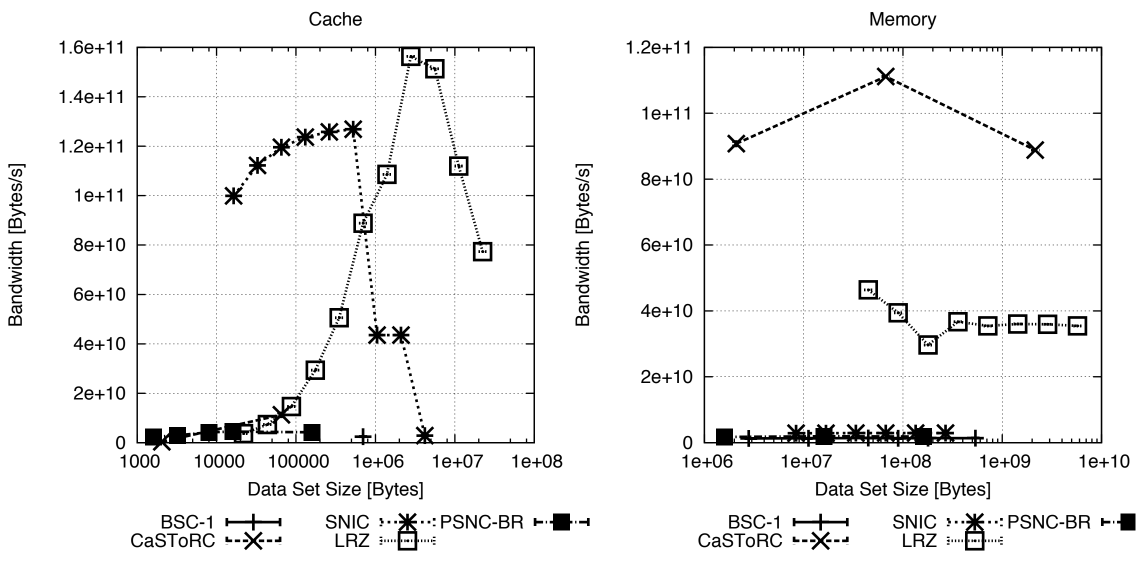


Figure 70 STREAM *scale* performance measured as B/s, with the cache effects for the DSP (SNIC) and Magny-Cours (LRZ) prototypes clearly visible.

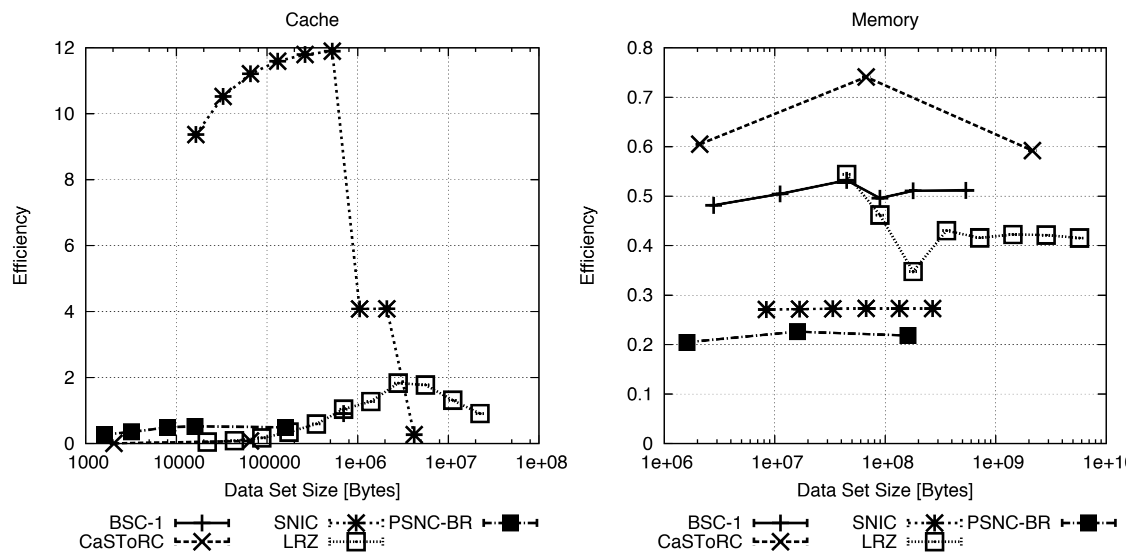


Figure 71 Efficiency of STREAM *scale* expressed as fraction of peak memory bandwidth. The cache memory bandwidth amplification is apparent.

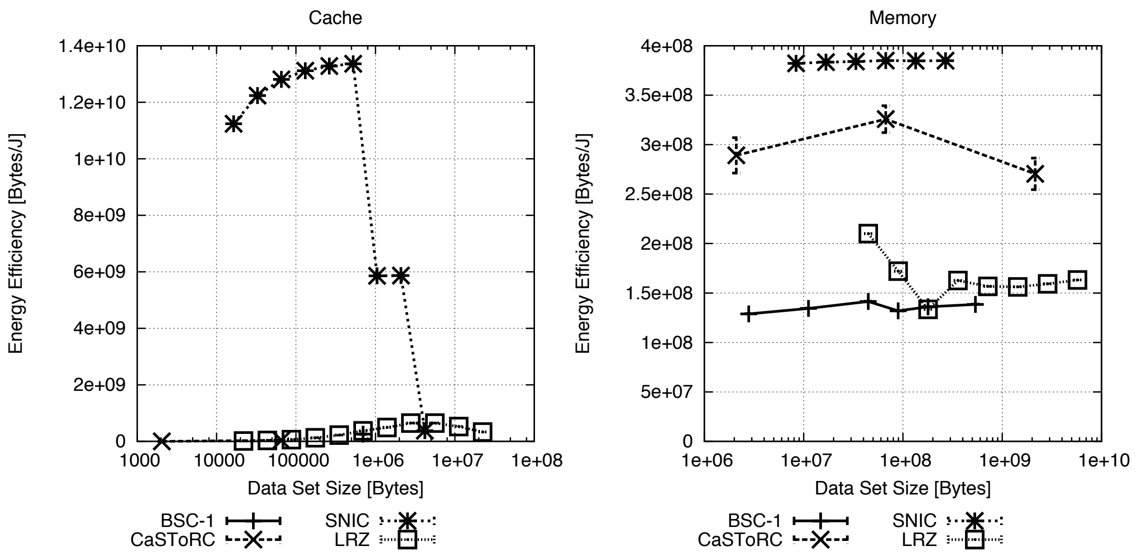


Figure 72 STREAM *scale* energy efficiency measured as B/J.

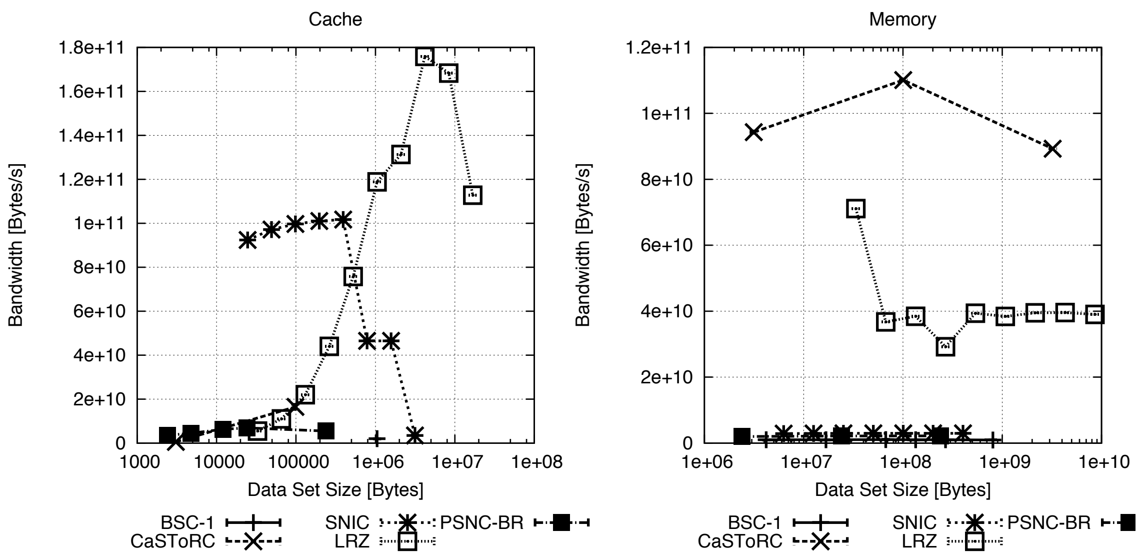


Figure 73 STREAM *sum* performance measured as B/s, with the cache effects for the DSP (SNIC) and Magny-Cours (LRZ) prototypes clearly visible.

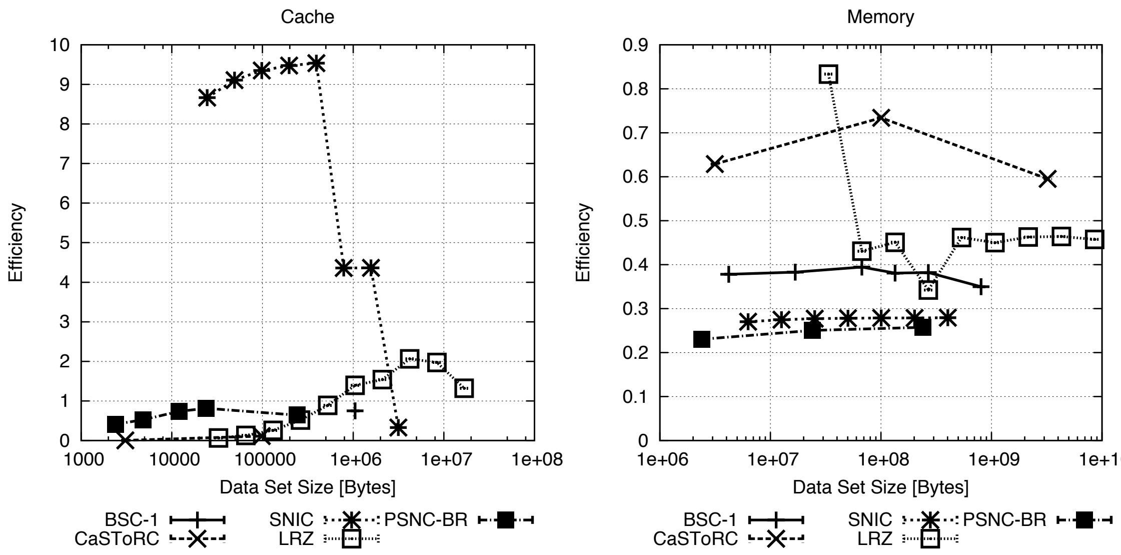


Figure 74 Efficiency of STREAM *sum* expressed as fraction of peak memory bandwidth. The cache memory bandwidth amplification is apparent.

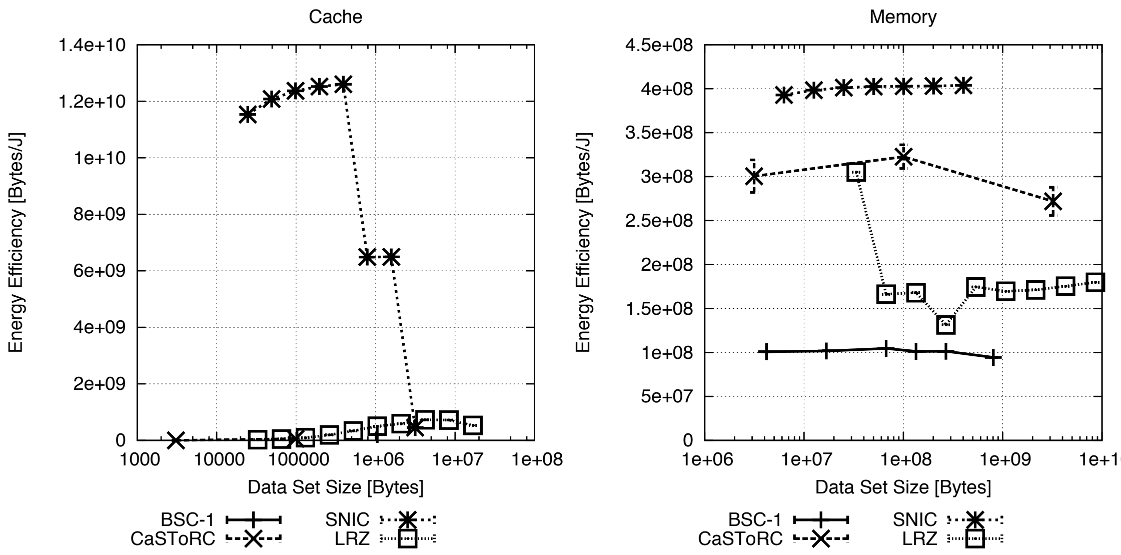


Figure 75 STREAM *sum* energy efficiency measured as B/J.

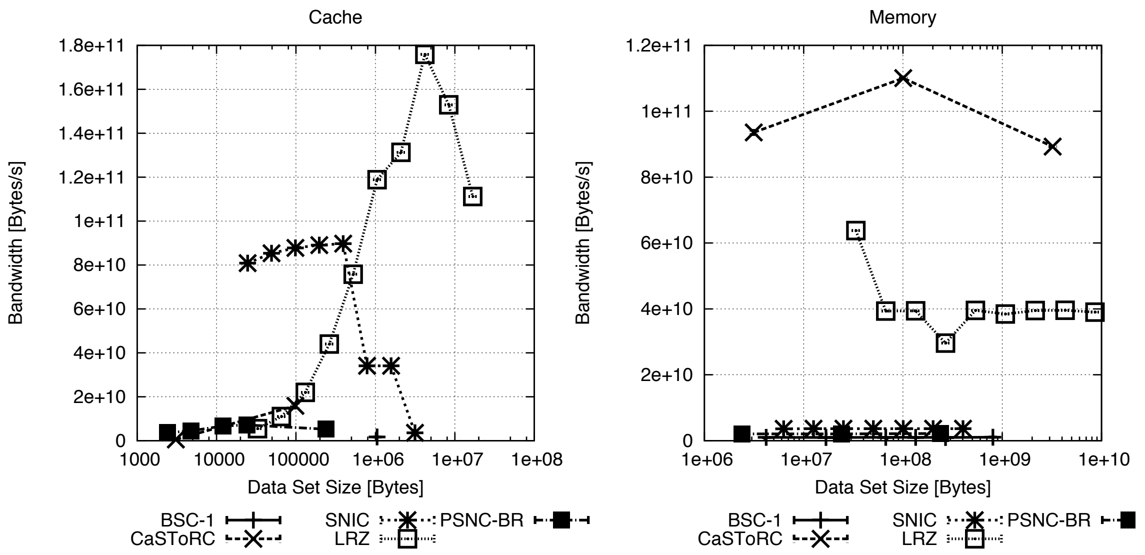


Figure 76 STREAM *triad* performance measured as B/s, with the cache effects for the DSP (SNIC) and Magny-Cours (LRZ) prototypes clearly visible.

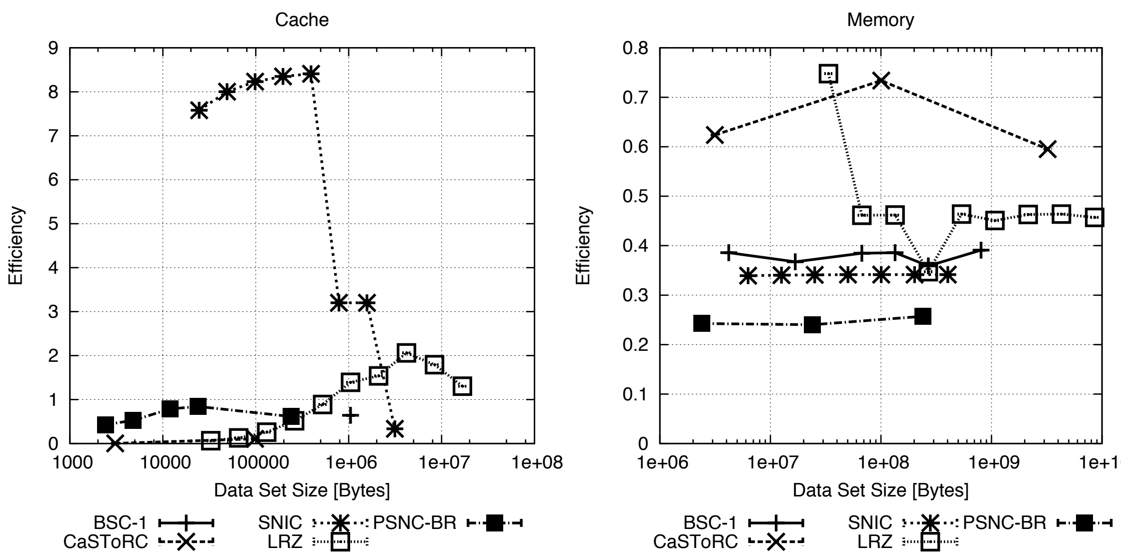


Figure 77 Efficiency of STREAM *triad* expressed as fraction of peak memory bandwidth. The cache memory bandwidth amplification is apparent.

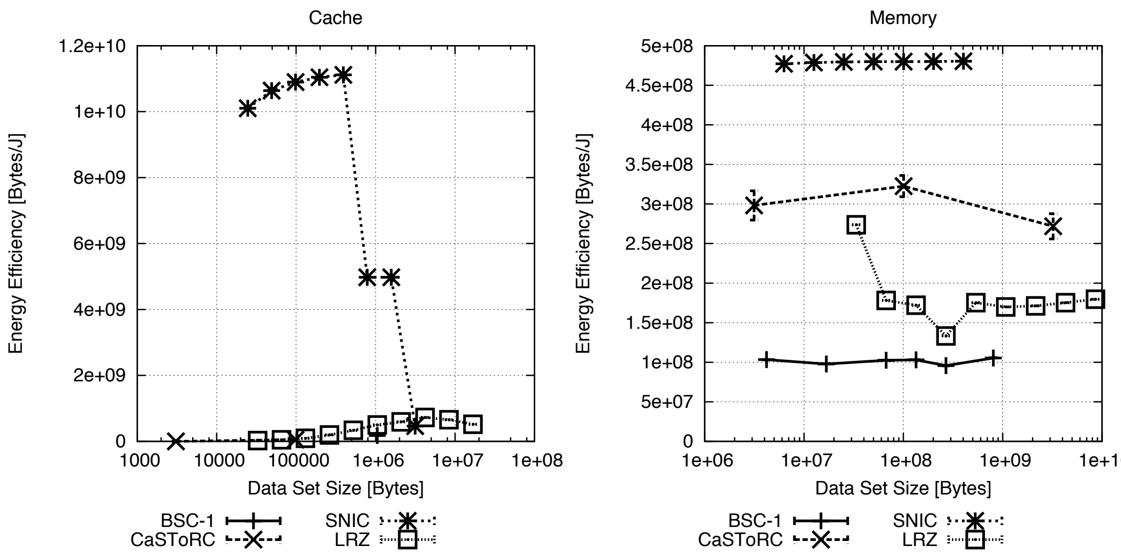


Figure 78 STREAM *triad* energy efficiency measured as B/J.

6.1.2 *Dense matrix multiplication*

Dense matrix multiplication in the form of the EuroBen benchmark mod2am was evaluated on several of the prototypes, with vendor optimized implementations on AMD, Intel and NVIDIA prototypes. The performance difference between an optimized and un-optimized matrix multiplication can be quite large since an un-optimized version may perform quite poorly; this is evident in the results reported below. We describe the FPGA (JKU) matrix-multiplication results in some detail before summarising the results for some of the other prototypes. No explicit prototype specific optimization was made for mod2am. The FPGA implementation did not use the EuroBen mod2am code since the matrix multiplication is directly implemented in hardware.

FPGA matrix computation acceleration, JKU

The FPGA stream architecture for matrix multiplication supports rectangular matrix multiplication with parameterizable problem size. Using a cache oblivious algorithm [Frigo-1999], large matrices are partitioned on the host processor and submatrices multiplied on the

FPGA minimizing the total communication volume between the host and the FPGA accelerator. The stream design targeted 8 x 8 tiles together with PCIe interface and controller logic and a 400 MHz clock for the Xilinx ML605 FPGA. This design has not reached a fully functional state at this time but a 7 x 7 tile design operating at 100 MHz is fully functional and has been evaluated.

The performance of the FPGA stream architecture for matrix multiplication was measured in clock cycles for different problem sizes and for three different numbers of tiles up to 7 x 7 tiles, Figure 79, and compared with a cycle accurate simulation. Based on the measurements the execution time, t, for multiplication of N x N matrices including PCIe Gen-1 x8 communication time is predicted to follow the equation $t=t_{startup}+N^3/(\# \text{ tiles x } f)+t_{drainage}$, where f is the clock frequency, as shown in Figure 80. The contribution of the PCIe communication time is illustrated in Figure 81 that shows the measured communication time as a function of the transferred data set size.

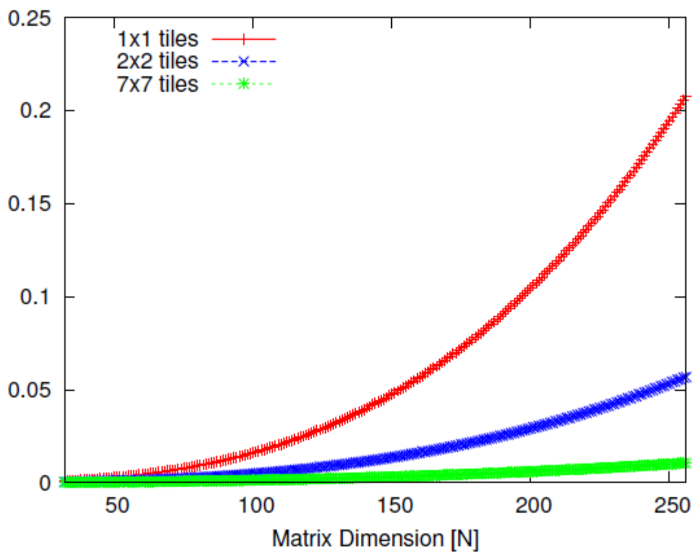


Figure 79 Measured execution time in seconds for matrix multiplication including PCIe communication time on the Xilinx ML605 FPGA for 1, 4 and 49 tiles

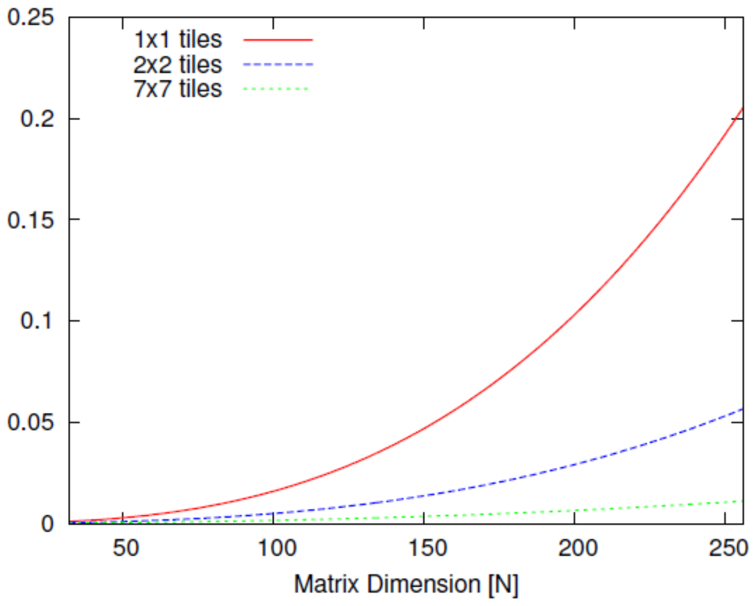


Figure 80 Predicted execution time in seconds for matrix multiplication including PCIe communication time on the Xilinx ML605 FPGA for 1, 4 and 49 tiles.

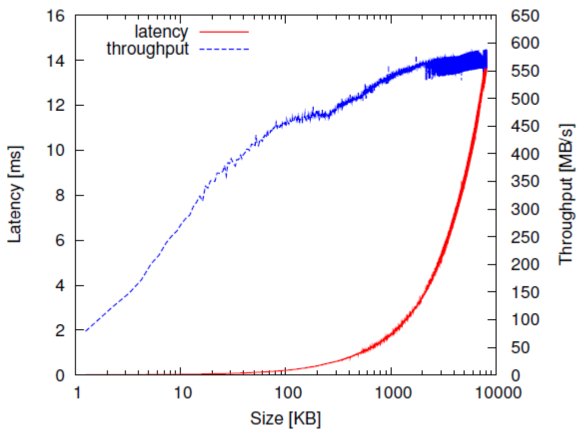


Figure 81 PCIe Gen-1 x8 data transfer time in milliseconds as a function of transferred data set size.

The predicted performance for multiplication of matrices of a size of up to 2000 x 2000 is shown in Figure 82, which shows that for 7 x 7 tiles the stream architecture FPGA design achieves about 8 GF/s for the Xilinx ML605. The measured peak performance for matrix multiplication using 7 x 7 tiles was 4.8 GF/s. (The measured execution time for the AMD Interlagos Opteron core did not use the FMA4 instruction set which limits the efficiency to less than 50%. The Opteron core efficiency in Figure 82 is about a quarter of the efficiency using FMA4 instructions reported in [Mora-2012].)

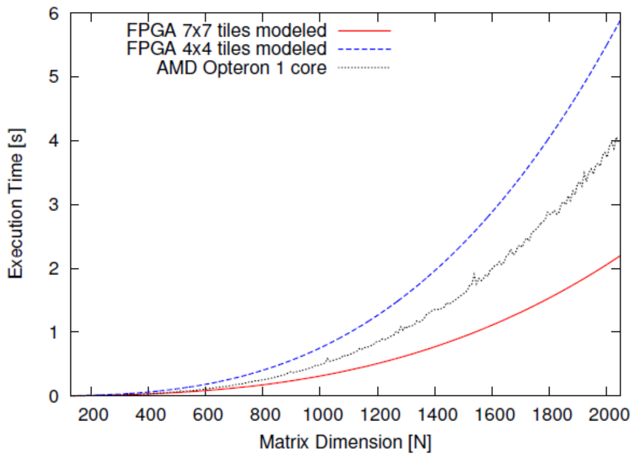


Figure 82 Predicted execution time in seconds using the stream architecture FPGA design for the Xilinx ML605.

The energy efficiency was analyzed based on power measurements via an on-chip hardware system monitor in the FPGA and compared with simulations using the Xilinx XPower analysis tool. The result is shown in Figure 83 for a 32 x 32 matrix multiplication. The energy efficiency was measured at 0.92 GF/J.

This prototype effort offers a limited but constructive perspective on the capabilities of FPGA-based accelerators for floating-point intensive computations. With additional engineering effort to raise the clock frequency to the planned 400 MHz, the performance of the FPGA may be quadrupled. This should be achieved with only a minor increase in energy consumption since for the streaming architecture design for matrix multiplication most of the power consumption is static power as shown in the Table 25 below for up to 8 tiles operated at 250 MHz. Furthermore, with additional engineering, a 64-tile 400 MHz streaming architecture implementation should be able to achieve about 5 GF/J.

FPUs	1	2	4	8
Time, m sec	123	62	31	16
Static Power, mW	2714	2716	2719	2723
Dynamic Power, mW	298	349	448	560

Table 25 FPGA power estimates at 250 MHz clock frequency for 1 to 8 tiles.

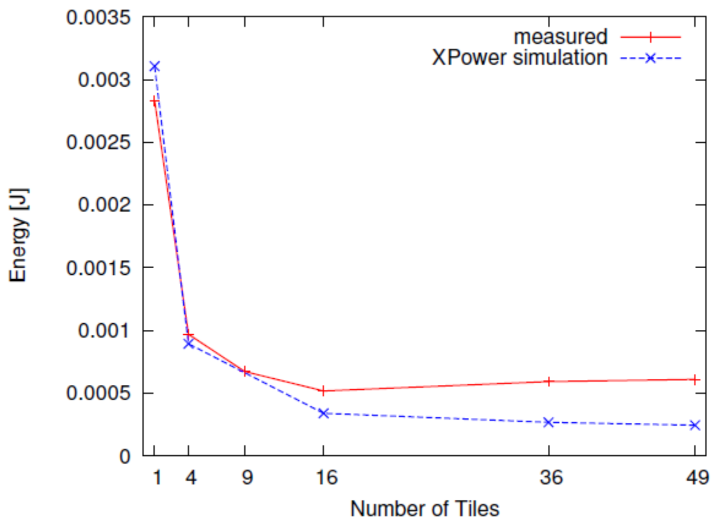


Figure 83 Measured and predicted FPGA power consumption as a function of the number of tiles for a 32 x 32 matrix multiplication.

Summary of Matrix Multiplication Results

Of the prototypes the energy efficiency for dense matrix multiplication for the stream architecture FPGA implementation is the best with an estimated energy efficiency of 4 – 5 GF/J. This result compares very favorably with the 0.9 GF/J reported for the 32 nm AMD 16-core Interlagos at ISC 2012 [Mora-2012].

Performance results for mod2am measured as GF/s are shown in Figure 84. Most of the results are for matrices sufficiently small to fit in last level cache. The corresponding efficiencies are reported in Figure 85, and the energy efficiencies in Figure 86.

The AMD APU results (PSNC-BR, PSNC-LL, PSNC-TR) are based on using ACML enabling the relatively good efficiencies. The Intel Sandy Bridge (PSNC-SB) results are based on MKL; unfortunately the Ivy Bridge (PSNC-IB) data was based on ACML and is therefore showing relatively poor performance. When using ACML on the Sandy Bridge performance drops roughly with a factor of 32, see Table 94 and Table 93. It should be noticed that for the DSP the matrix-multiplication routine used in the HPL benchmark achieved an efficiency of about 95% as opposed to the very poor efficiency of the mod2am benchmark for the DSP.

With the mostly very poor efficiency of the mod2am implementations, the energy efficiency is also poor. For example, for Magny-Cours servers using the same CPU as in our prototype, an energy efficiency of 370 MF/J has been reported [Mora-2010], and for Ivy Bridge measured energy efficiencies exceed 1 GF/J for optimized codes. For the DSP prototype (SNIC) an energy efficiency of 2.6 GF/J was achieved for HPL, which implies that the matrix multiplication routine used in that benchmark achieved even better energy efficiency. It is interesting to note though that the AMD Llano A8 APU, despite having a relatively efficient matrix multiplication implementation, showed a fairly poor energy efficiency compared to the best known results for the AMD Interlagos and Intel Sandy and Ivy Bridge CPUs and optimized DSP energy efficiency.

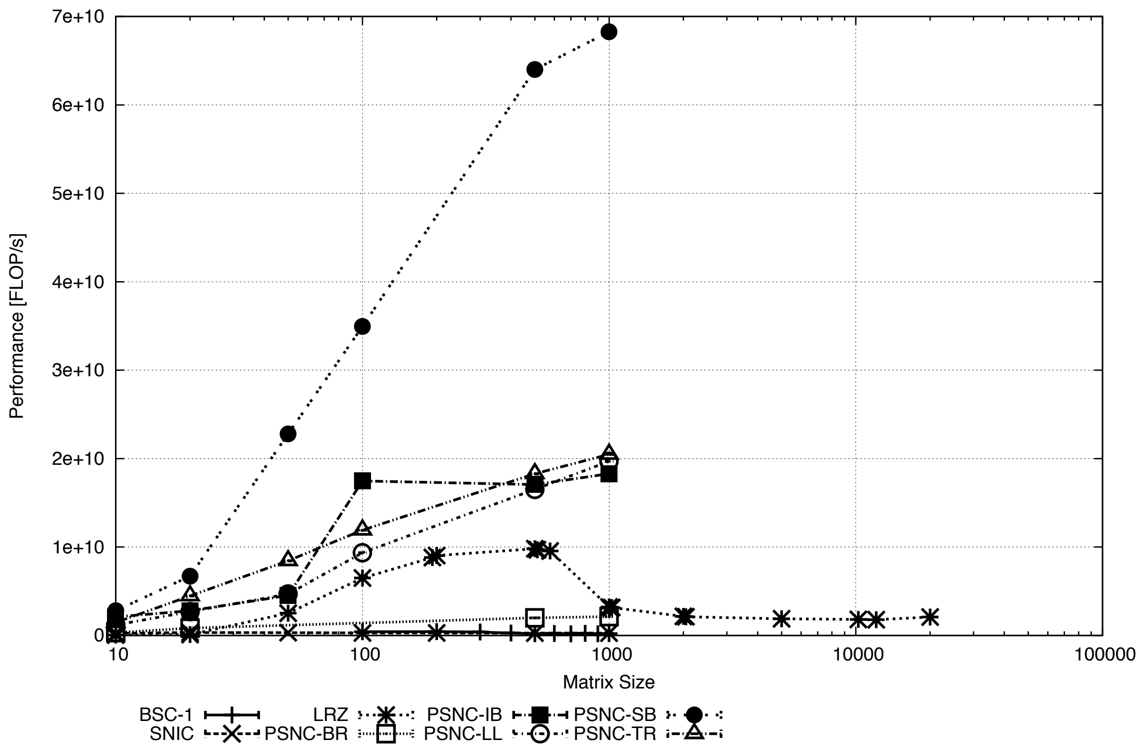


Figure 84 Performance of mostly unoptimized implementations of mod2am on prototypes.

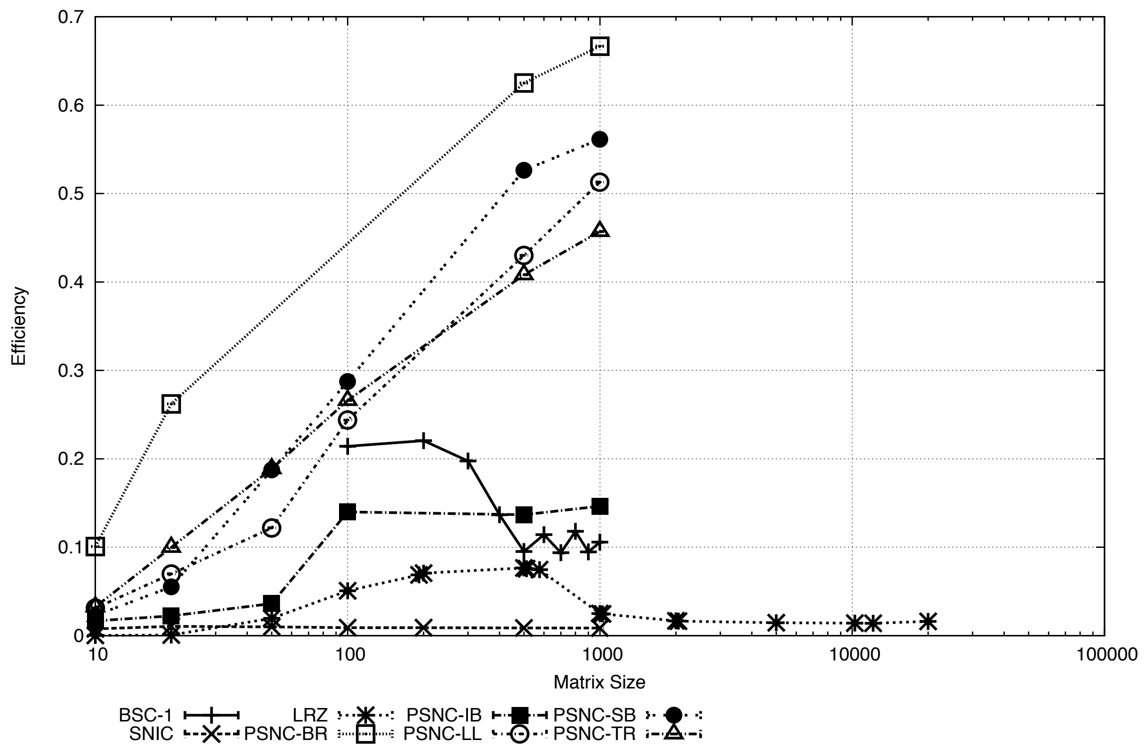


Figure 85 The efficiencies of mostly unoptimized implementations of mod2am.

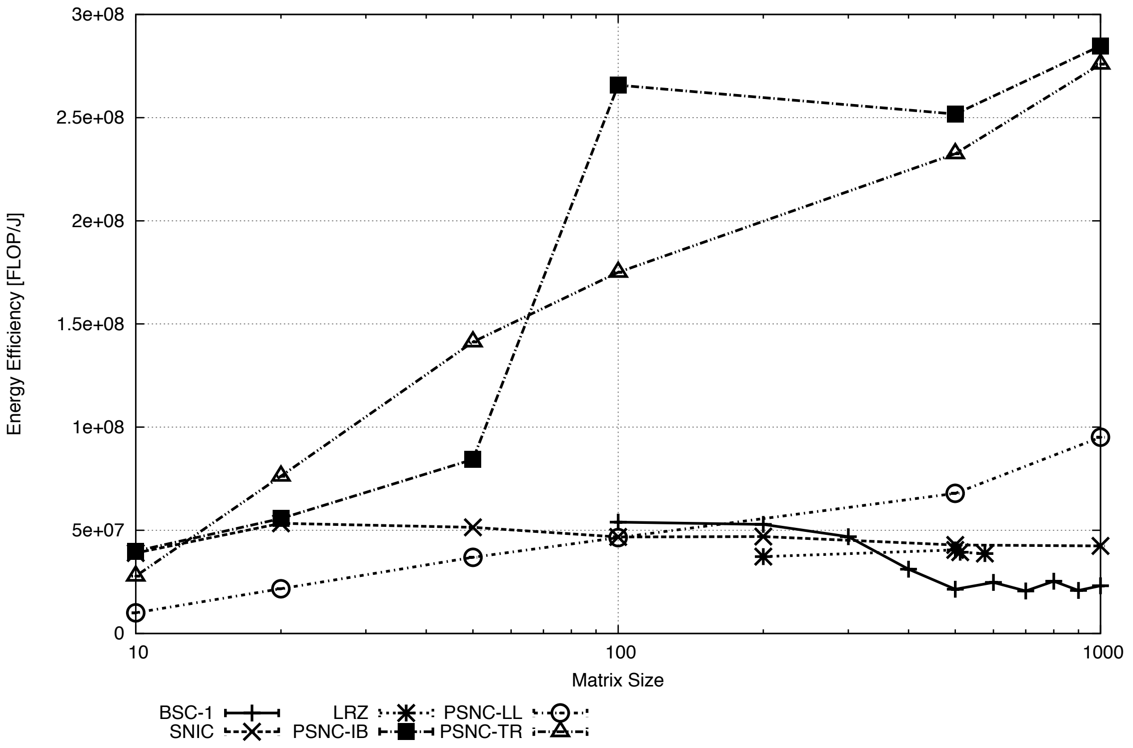


Figure 86 The energy efficiencies of mostly un-optimized implementations of mod2am.

Some scalability results were also achieved for the Terga 2 (BSC-1) cluster, as reported in Figure 87 – Figure 89.

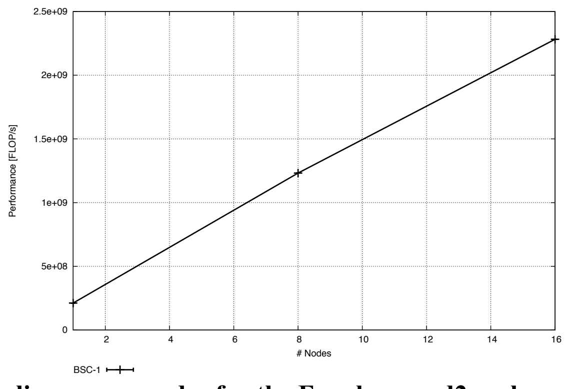


Figure 87 Performance scaling across nodes for the Euroben mod2am kernel on the Tegra 2 prototype.

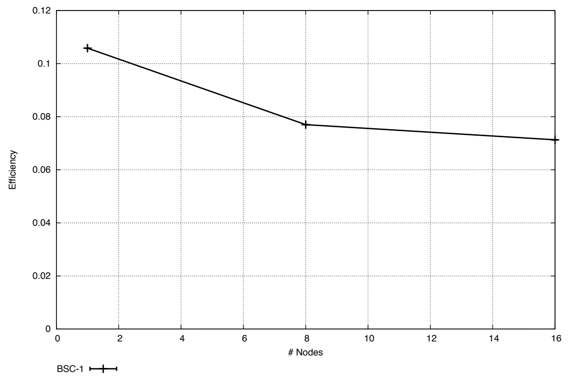


Figure 88 Computational efficiency scaling obtained for the Euroben mod2am kernel on the Tegra 2 prototype.

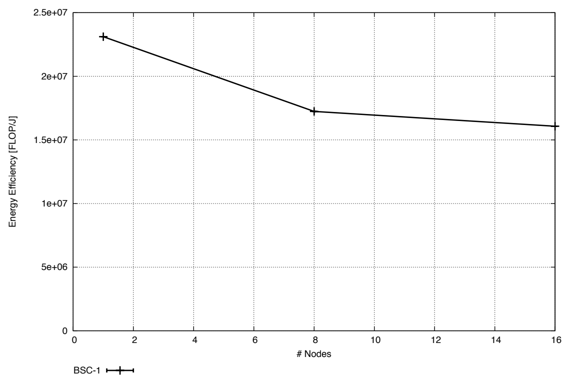


Figure 89 Energy efficiency scaling for the Euroben mod2am kernel obtained by the Tegra 2 prototype.

6.1.3 *Sparse Matrix Vector Multiplication*

Sparse matrix vector multiplication does not perform well on many architectures due to its dependence on indirect addressing leading to both more memory accesses per operation than dense matrix operations, more integer arithmetic for address computations, irregular memory accesses and low chance of benefiting from SIMD instructions. Many architectures are geared towards doing well on functions that can be structured into streaming operations or benefit from data reuse and there can be a stark difference in performance for sparse matrix-vector multiplication compared to dense matrix-vector multiplication.

The performance results of mod2as runs on a few of the prototypes are shown in Figure 90, the efficiencies in Figure 91 and the energy efficiency in Figure 92. The implementations all have poor performance, poor efficiency and correspondingly poor energy efficiency. The Magny-Cours (LRZ) prototype demonstrates the efficiency of its L3 cache (24 MB) with 1 million elements occupying 12 MB of memory. Compared to the other prototypes the AMD Brazos APU (PSNC-BR) and the Tegra 2 (BSC-1) nodes performed well. Interestingly, despite the poor efficiency of mod2as on the DSP (SNIC) its energy efficiency was relatively good.

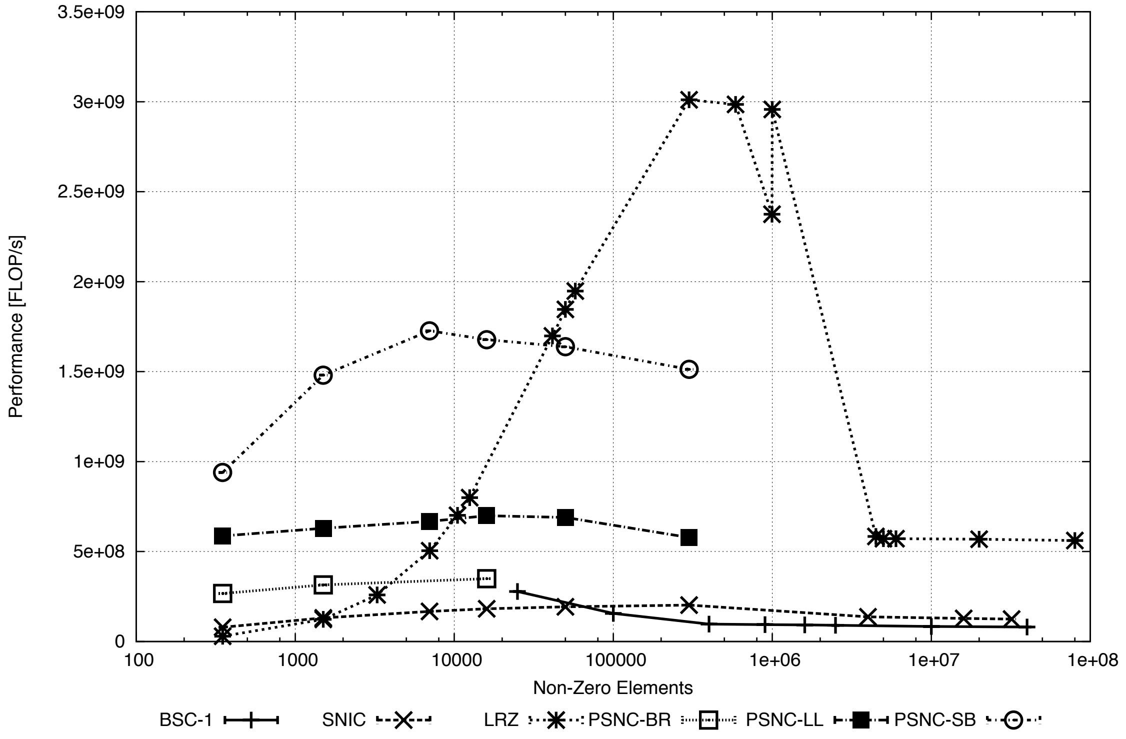


Figure 90 Performance of mod2as on some of the prototypes.

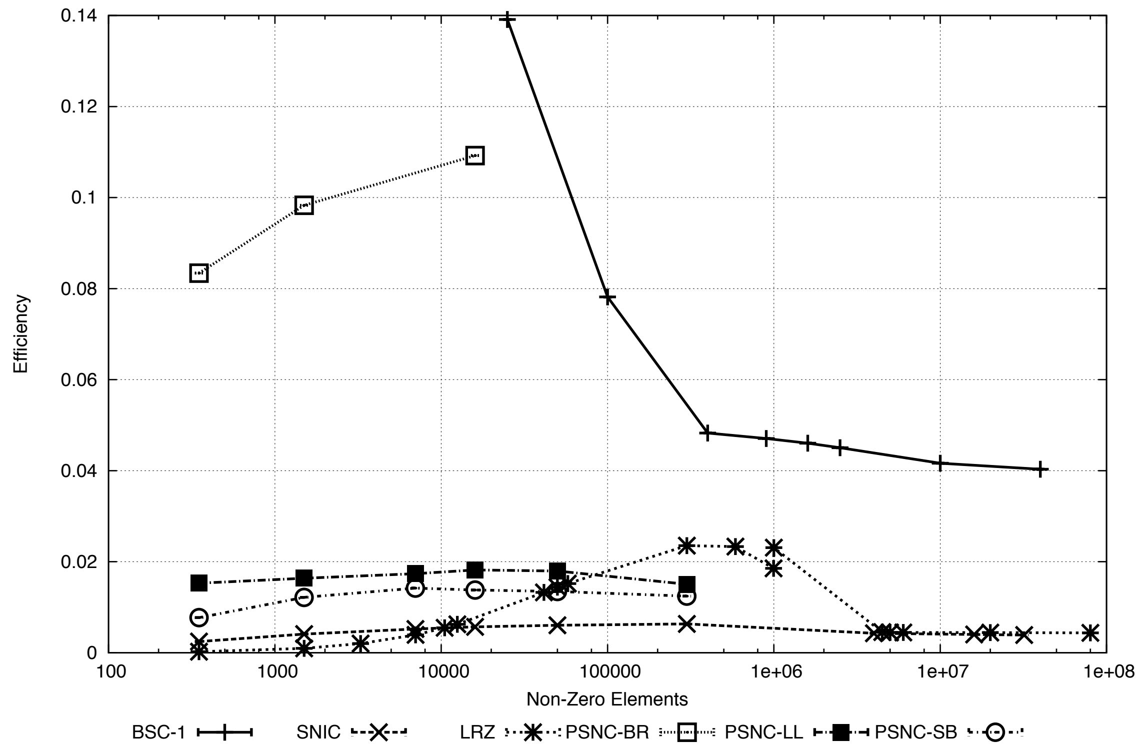


Figure 91 Computational efficiency of mod2as on some of the prototypes.

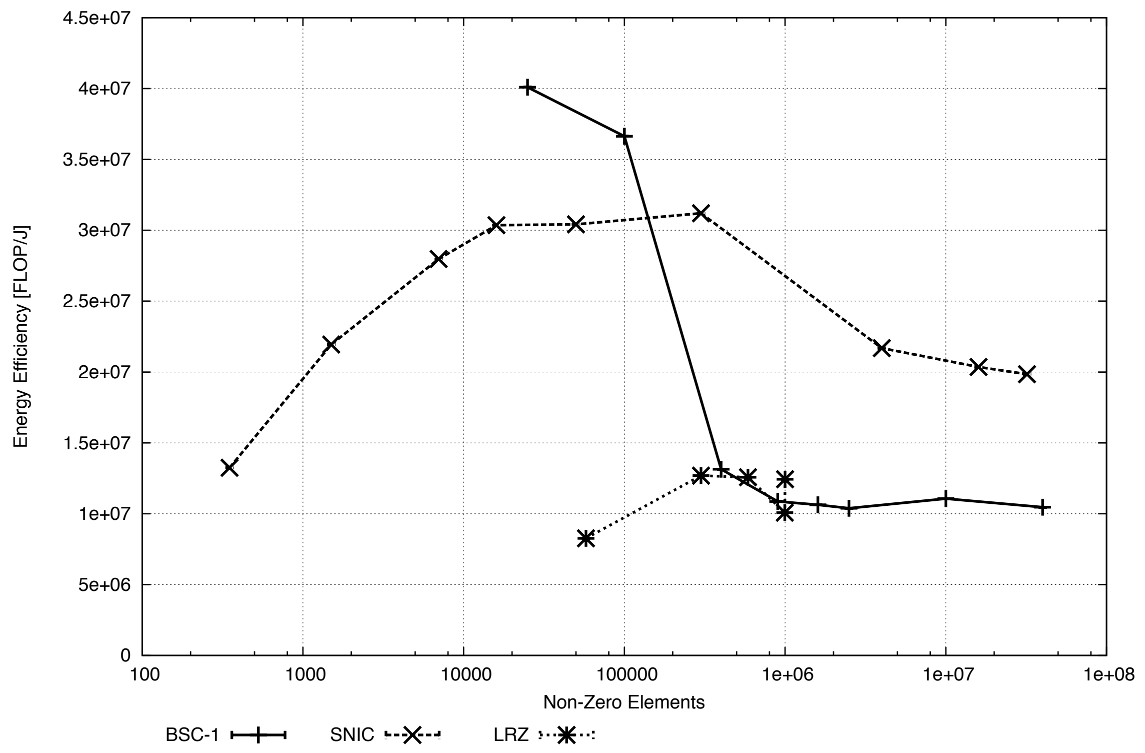


Figure 92 Energy efficiency of the mod2as implementations on some of the prototypes.

6.1.4 *FFT*

Figure 93 summarizes the mod2f performance on the prototypes, Figure 94 the corresponding efficiencies and Figure 95 the energy efficiencies. The efficiencies achieved on most architectures are considerably lower than what may be achieved for matrix multiplication because of the relatively low FFT data reuse (O(logN)), its wide range of power-of-two memory access strides, the unbalanced number of additions and multiplications, and limited ability to use fused multiply-add instructions. Optimizations are more complicated than for dense matrix multiplication.

The efficiencies achieved for the AMD Brazos (PSNC-BR) and Magny-Cours (LRZ) protoypes are however noticeably better than for the other prototypes. The GPU-based (CaSToRC) prototype can showcase the importance of properly optimized software. It uses the vendor-provided CUFFT library achieving over 22% efficiency on a single GPU, yielding about 220 GF/s of raw performance at around 400 MF/J, as detailed in Table 48. Unfortunately due to problems with the power measurement equipment no enery efficiency results are available for mod2f for PSNC.

6.1.5 *Linpack*

HPL (and implicitly dense matrix multiplication) is the most widely used benchmark to assess performance and energy efficiency for engineering and science applications, and the basis for the Top500 and Green500 rankings of computer systems. Vendors of systems for the HPC market often produce highly optimized versions of the official HPL code available from netlib. ARM and Texas Instruments CPUs are not currently used by HPC platform vendors as main computational resources for HPC servers. Hence, optimized HPL implementations were not available for the Tegra 2 (BSC-1) and DSP (SNIC) prototype nodes. For the DSP, an optimized version was developed for assessment of achievable efficiency and energy efficiency. For the Magny-Cours (LRZ) prototype an optimized version from the vendor was used. For the GPU prototype (CaSToRC) vendor-optimized code for concurrent use of CPU hosts and GPUs as well as for direct GPU-GPU communication was used.

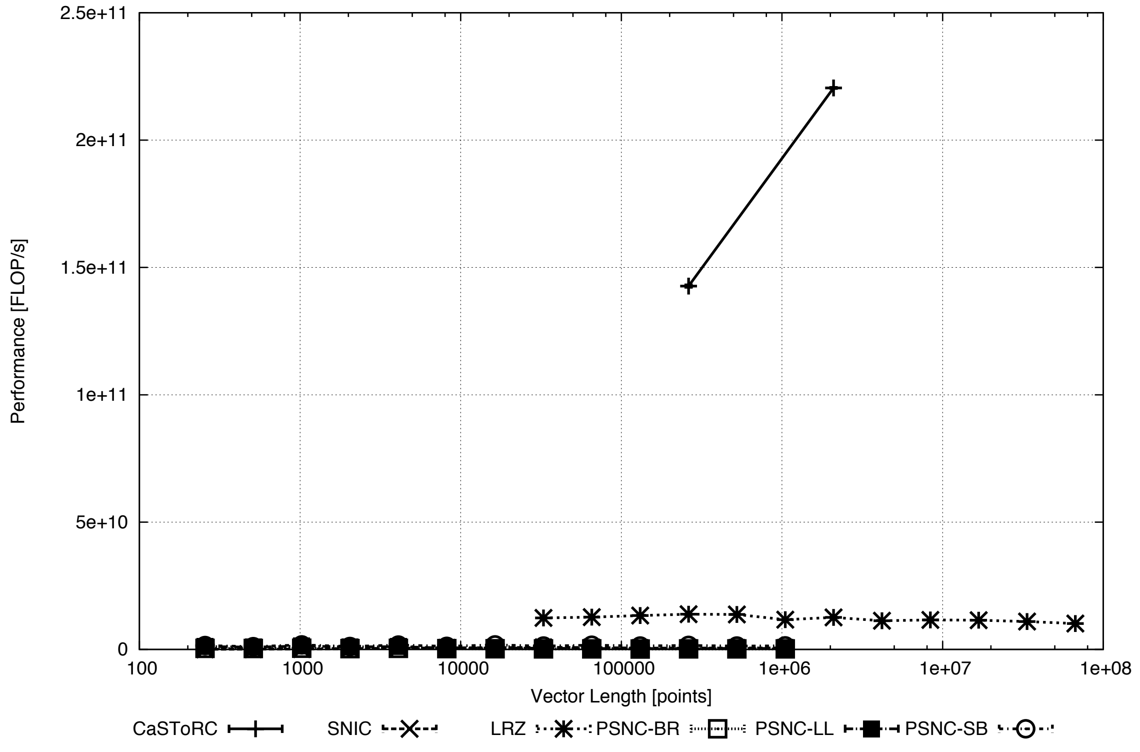


Figure 93 Performance of mod2f on some of the prototypes.

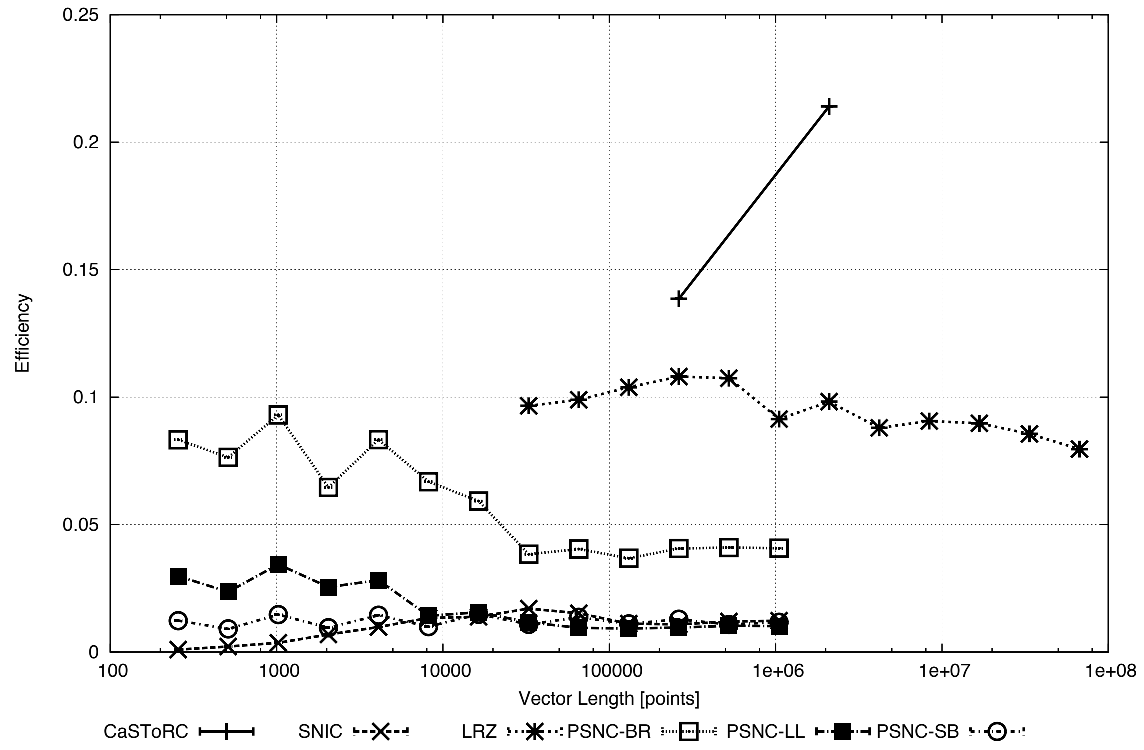


Figure 94 Efficiency of mod2f on the prototypes.

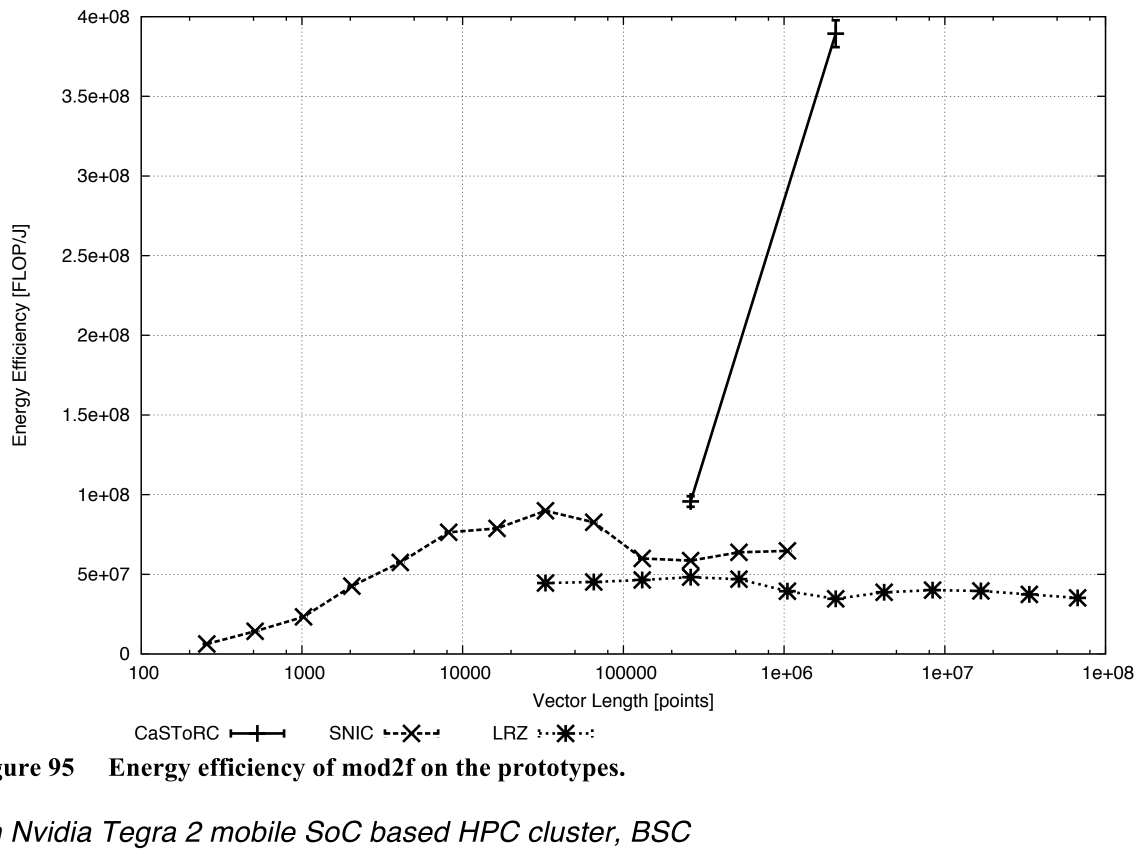


Figure 95 Energy efficiency of mod2f on the prototypes.

An Nvidia Tegra 2 mobile SoC based HPC cluster, BSC

The HPL node results for the Tegra 2 prototype (BSC-1) are shown in Figure 96. The ATLAS linear algebra package was used to search for optimal blocking for the memory hierarchy and resulted in a peak performance of 1.13 GF/s at a power consumption of 8.4 W for the node. With a theoretical peak performance of 2 GF/s at 1 GHz, the maximum achieved efficiency was 57% and the energy efficiency was 0.14 GF/J. The parameters that yield the best performance also yield the maximum power efficiency. Minimizing the execution time does not maximize the energy efficiency for all architectures, as for example has been demonstrated to be the case for the Intel Polaris CPU [Intel-2007] and x86 CPUs [Supermicro-2008].

The distribution of the power consumption in a single node when HPL is executed is shown in Figure 97. As seen from the Figure 97 the total power is significantly higher than the sum of the power of all components that are actually used in computation (cores, memory, network interfaces…). The large part of the power not accounted for (labelled OTHER) accounts for more than 67% of the total power. That part of the power includes voltage regulators, on-board multimedia devices with related circuitry, per blade PSU losses and on-chip power sinks. If boards were to be optimized for HPC systems, a large fraction of the OTHER power (HDMI, keyboard controller, …) could be eliminated.

In addition to single-node performance, HPL shows good weak scaling for the Tegra 2 prototype cluster with 256 nodes. As shown in Figure 106 and Table 106 the cluster achieves 118 MF/J using 96 nodes at an aggregated performance of 99 GF/s which translates to 51% computational efficiency before slowly dropping off to 101 MF/J using 224 nodes. The single node benchmark achieves 140 MF/J at about 57% computational efficiency. This is competitive with AMD Opteron 6174 and Intel Xeon E5660-based clusters, but 10x lower than the most efficient GPU-accelerated systems, and 20x lower than the current leader on the Green500 list [Green500-2012].

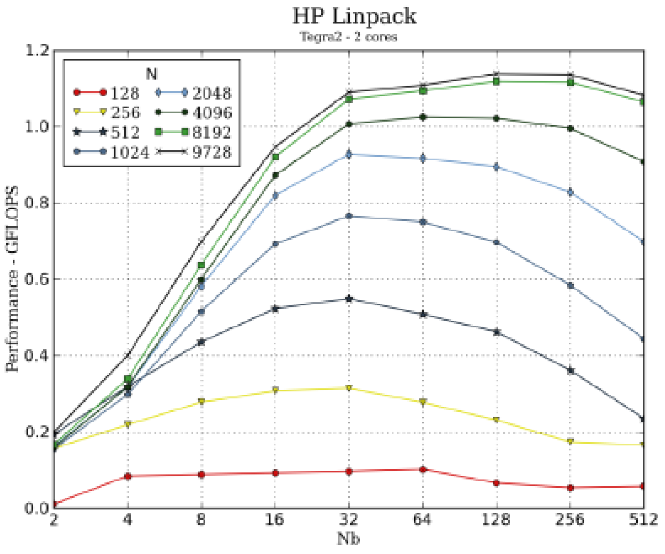


Figure 96 Tegra 2 HPL node performance for a range of matrix sizes and block sizes. Node theoretical peak performance 2 GF/s at 1 GHz.

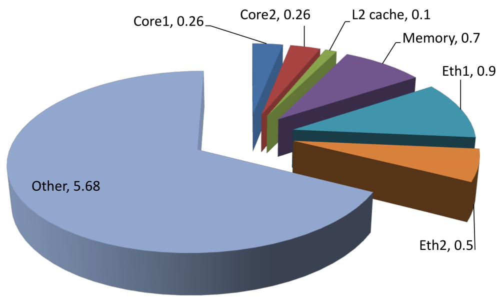


Figure 97 Power consumption of various components of the Tegra 2 prototype node for HPL. Total power consumption is 8.4 W.

GPU-GPU communication over PCIe and Infiniband, CaSToRC

A CUDA implementation of HPL allowing concurrent execution on the CPUs and GPUs was used. Apart from GPU implementations of the necessary compute kernels, this version employs pinned (page-locked) memory for asynchronous PCIe 'Host-to-Device' transfers in conjunction with the CUDA Stream API to overlap communications with computations. The key point in the hybrid CPU/GPU version is to find the optimal distribution between the CPUs and GPUs of the input matrices, knowing the relative performances of the CPU and GPU cores on DGEMM. The code was compiled with Intel MKL and OpenMPI. The input/output parameters used for the benchmarking are shown in Table 26.

The maximum achieved performance was 4.1 TF/s of a theoretical peak of 9.26 TF/s (8.24 TF/s for 16 GPUs and 1.02 TF/s for the 16 CPUs). Thus, the achieved efficiency was 44% and the corresponding energy efficiency was about 750 MF/J, which is consistent with Top500 entries of similar architectures [Green500-2012]. As a comparison, NVIDIA has reported a peak performance of 5.1 TF/s and 55% efficiency for an eight node, dual socket server with the same CPUs with 48 GB of memory, twice that of the prototype, the same GPUs and QDR Infiniband.[NVIDIA-2010]. NVIDIA did not report power consumption for the HPL result.

Parameter	Purpose	Parametric space
N	The order of the coefficient matrix	98 304
NB	The partitioning order factor	512, 768
PMAP	The process mapping	0 = Row-major
P	The number of process rows	2
Q	The number of process coloumns	8

Table 26 HPL parameters used for benchmarking of the GPU prototype.

DSP based node for HPC, SNIC

Hand-crafted assembler code for the inner matrix-matrix multiplication operations in combination with utilizing the L1 and L2 SRAM as scratch pad memories and the IDMA and EDMA3 engines to overlap the main memory data transfers with calculations yielded a matrix-matrix multiplication implementation with 95% efficiency (30.6 of 32 GF/s at 1 GHz). At 1.25 GHz the complete HPL benchmark implementation reached up to 30.9 GF/s or about 77% of theoretical peak performance. The difference in efficiency between the matrix-multiplication kernel and the HPL benchmark is in part due to the fact that the triangular solve and back-substitution is executed on a single core by un-optimized C code. Table 27 shows some of the HPL benchmark data for various matrix sizes on the 8-core DSP operating at 1.25 GHz.

The HPL efficiency is about 5% lower than the Top500 results reported for AMD Interlagos and 10 - 15% lower than what is reported for Sandy Bridge server CPUs. The energy efficiency is about 2.5 times higher than that of Interlagos and Sandy Bridge CPUs, both of which use 32 nm technology whereas the DSP is made in 40 nm technology. Figure 98 illustrates in some detail the power consumption for different matrix sizes reported in Table 27. Figure 99 shows the energy efficiency for the DSP at three different clock frequencies. For the DSP the way it is implemented, the power reduction at lower frequencies is not large enough to outweigh the increased execution time, so the maximum frequency that minimizes the execution time also maximizes the energy efficiency.

Size	Performance (GF/s) Peak 40 GF/s	Efficiency (%)	Power (W) Cores+Memory	Energy Efficiency (F/J)
127	0.7	1.7	4.8	143
255	3.0	7.6	5.6	541
511	6.8	16.9	7.0	961
1023	13.0	32.4	8.4	1548
2047	19.2	48.1	9.5	2035
4095	25.9	64.8	10.7	2421
8063	30.9	77.2	11.7	2644

Table 27 Performance, efficiency and energy efficiency of the 8-core DSP prototype node.

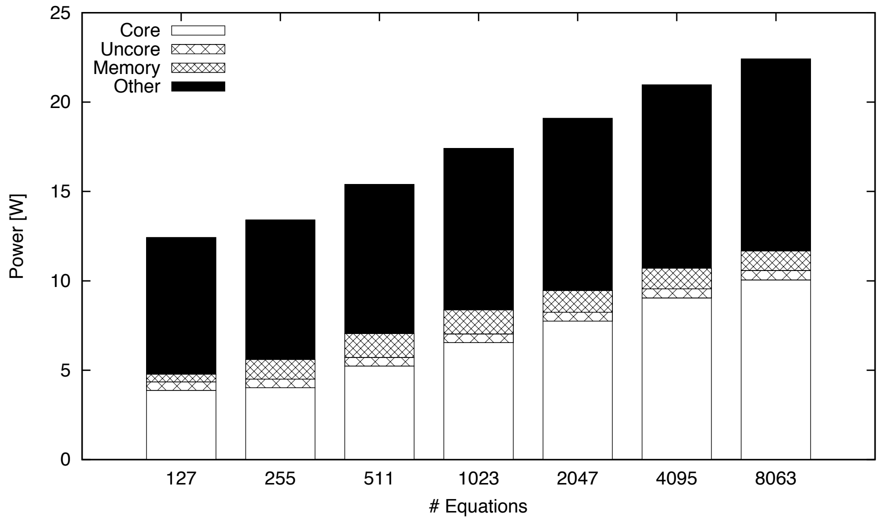


Figure 98 Power consumption of the measured subsystems of the DSP node for different HPL runs at 1250 MHz clock frequency.

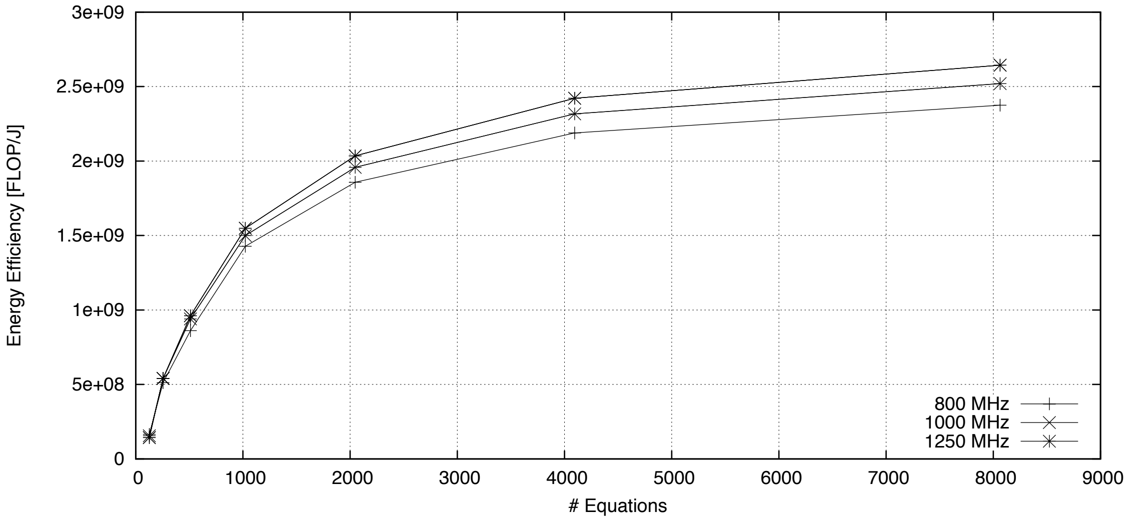


Figure 99 Energy efficiency for HPL on the DSP prototype for various problem sizes at different clock frequencies.

Linpack Summary

The performance of HPL on several prototypes is summarized in Figure 100 and the efficiency is summarized in Figure 101. The peak performance of the prototype nodes covers a range of about a factor of 500 from the 2 GF/s of the Tegra 2 (BSC-1) node to about 1.15 TF/s for the x86+GPU (CaSToRC) prototype, which to a large degree is the reason for the measured performance range shown in Figure 100. The achieved HPL efficiencies show a modest range with the optimized DSP code having the best efficiency for matrix sizes that are sufficiently small to fit in the 512 MB DDR3 memory of the DSP (SNIC) prototype, followed by the optimized vendor HPL code for the Magny-Cours (LRZ) prototype. The maximum DSP efficiency was 77% at which point the Magny-Cours efficiency was close to 65%, though the maximum efficiency for the Magny-Cours prototype was close to 85% for very large matrices. The Tegra 2 implementation yielded an efficiency of 57% and that of the optimized GPU implementation an efficiency of 44%. For the Magny-Cours prototype, the measured total node power consumption was 333 W. The resulting node efficiency was 77% and the peak node energy efficiency was 0.326 GF/J which is in line with vendor values [Mora-2010]. A HPL implementation optimized by the vendor was used for the benchmark. Scalability tests were made using Intel MPI version 4.0.

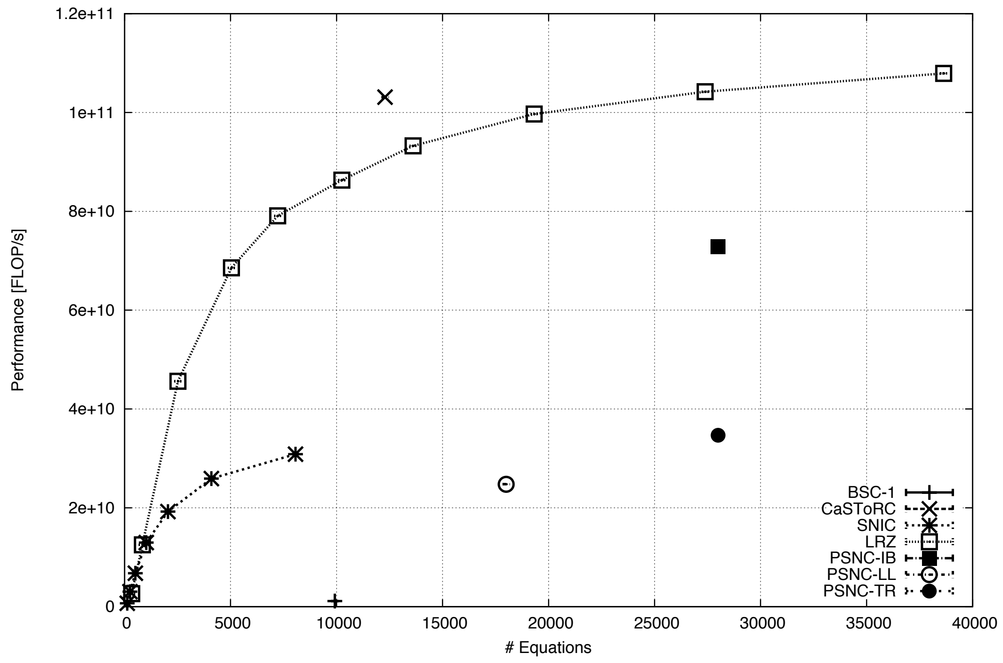


Figure 100 HPL node performance on several prototypes.

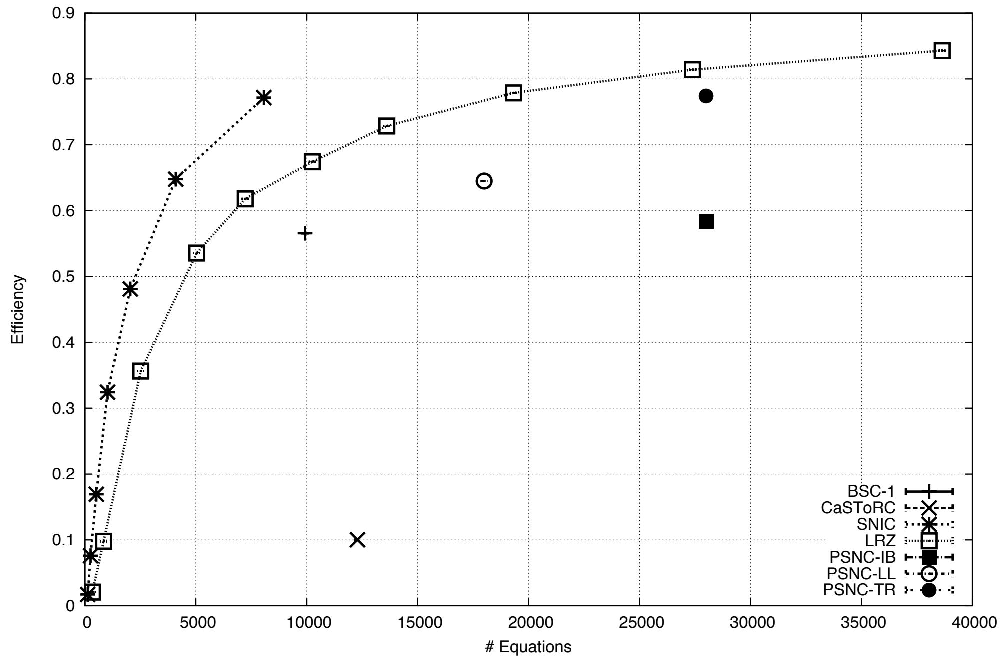


Figure 101 HPL node efficiencies achieved on several prototypes.

In regards to energy efficiency the DSP node at 2.64 Gflop/J far exceeds the energy efficiency of any of the other prototypes, Figure 102.

HPL scalability experiments were carried out on the Tegra 2, x86+GPU and Magny-Cours clusters. The results on performance, efficiency and energy efficiency are shown in Figure 104 through Figure 106. The surprising increase in efficiency as the number of nodes increases for the x86+GPU prototype is due to the fact that, by mistake, the matrix dimension N was scaled in proportion to the number of GPUs used for the benchmarking. Thus, the size of the data set in each node was increasing with the number of nodes.

The relationship between the performance and the power consumption while scaling across nodes for three of the prototypes is shown in Figure 103. The power-performance relationship is indeed very "clean" with the Tegra 2 showing the slowest growth rate and the x86+GPU prototype showing the highest growth rate.

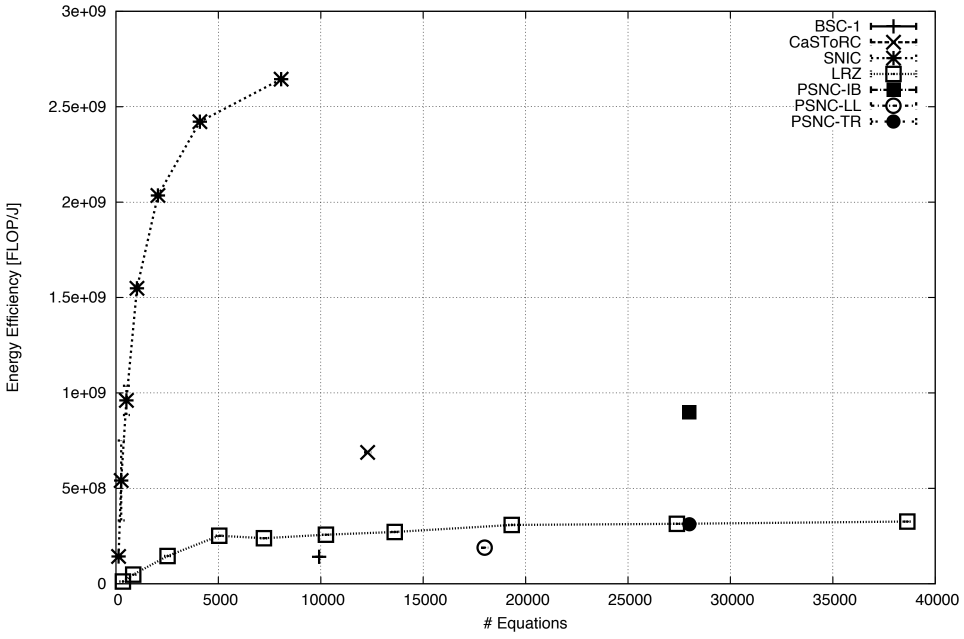


Figure 102 HPL node energy efficiencies achieved on several prototypes.

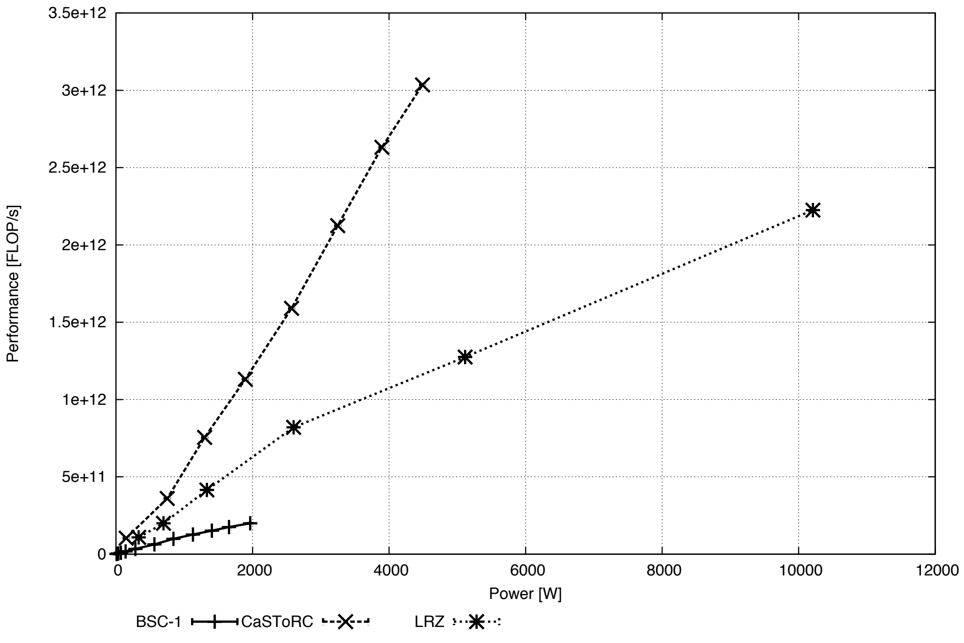


Figure 103 Performance plotted against power for the HPL benchmark scaling across nodes.

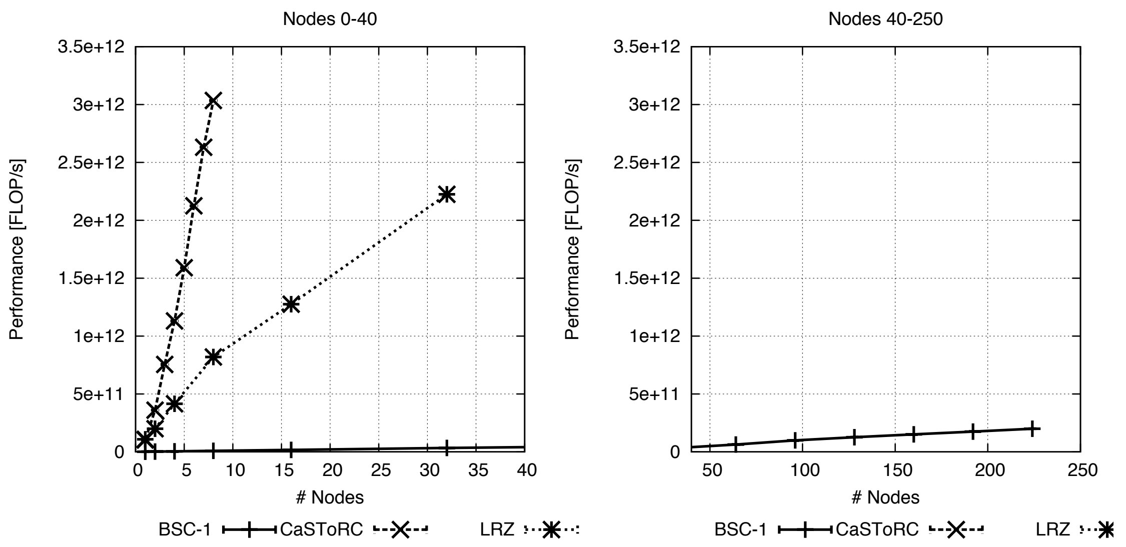


Figure 104 Scalability of HPL performance on the Tegra 2, x86+GPU and Magny-cours clusters.

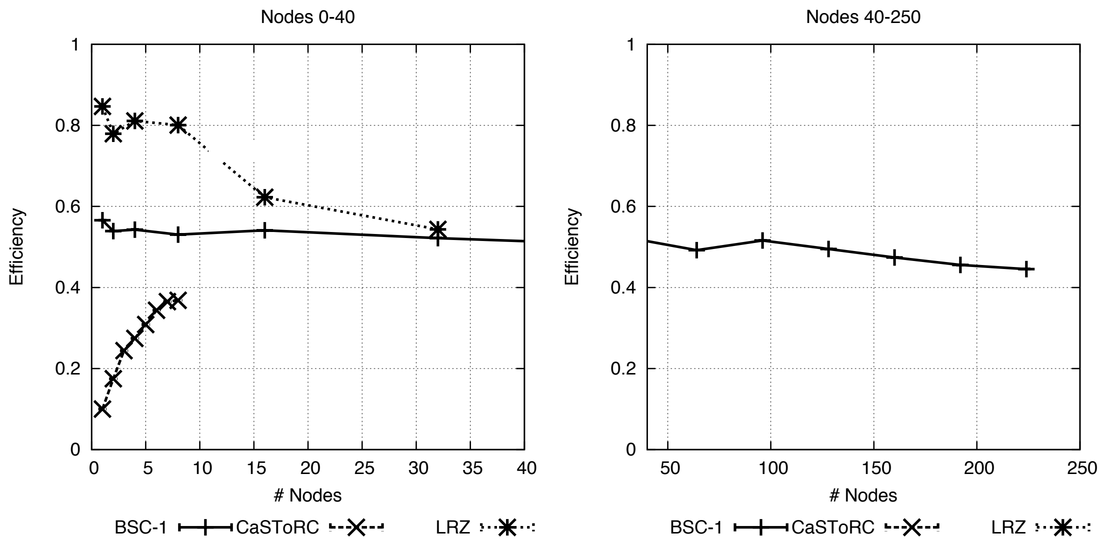


Figure 105 Scalability of HPL efficiency on the Tegra 2, x86+GPU, and Magny-cours clusters.

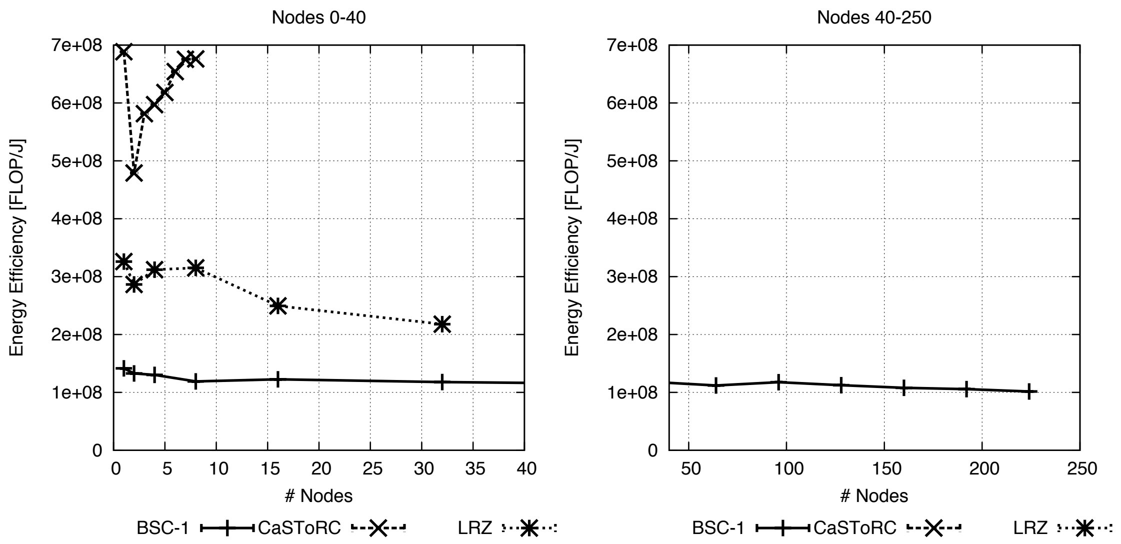


Figure 106 Scalability of the HPL energy efficiency on the Tegra 2, x86+GPU, and Magny-cours clusters.

6.1.6 *Hydro*

The Hydro code captures the essence of an application in fluid dynamics, whereas the previous benchmarks represent computational kernels in different types of applications with the exception of STREAM that is a memory system test relevant for applications with regular/strided memory accesses. Hydro includes many types of operations and the simple measures of floating-point operations per second or bytes per second in achieved memory accesses are not the most meaningful for the code, nor easy to use to infer the performance of other applications. Therefore, a more Hydro specific measure is used in assessing the prototypes capabilities for this benchmark: grid-point updates/sec and grid-point updates/J. The assessment was carried out for a range of problem sizes, with the range being dependent on the prototype assessed, but with some overlap between prototypes to allow for a direct comparison of some cases.

Hydro Summary

Since the floating-point and memory bandwidths of a node differs by more than two orders of magnitude between different prototypes it is not surprising that the performance varies widely, Figure 107. The best update rate was achieved by the GPU prototype and was 24.4 million updates/s for large grids on a node using two GPUs. For the DSP prototype the results are for a single core on a node, not the complete node.

The more interesting comparison is the energy efficiency achieved on the various platforms, Figure 108. Here the GPU prototype (CaSToRC) was again most efficient for larger grids with 63100 updates/J. The energy efficiency of the Magny-Cours node (LRZ) varied from 54571 to 64619 updates/J for the smallest grid sizes. The Magny-Cours prototype was about twice as energy efficient as the Ivy Bridge prototype (PSNC-IB) that was about three times as energy efficient as several of the non-traditional HPC node prototypes for small grid sizes. For large grid sizes the advantage of the Magny-Course node decreases to about a factor of two.

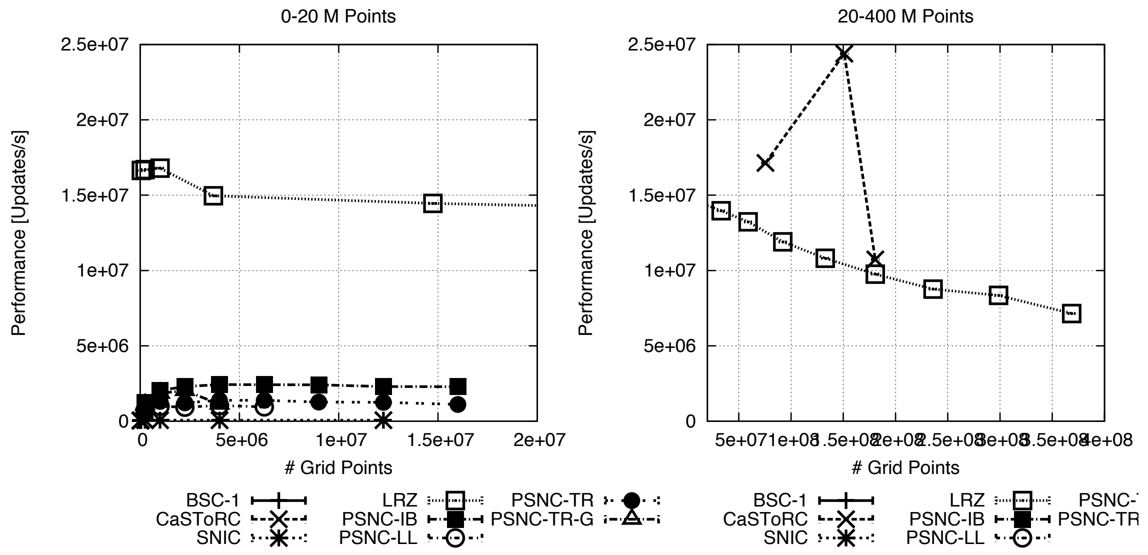


Figure 107 **Performance of the Hydro benchmark measured in terms of updates per grid-point per second for a range of grid points.**

The scalability of the Hydro benchmark across cluster nodes is shown in Figure 109. The parallel efficiency for the x86+GPU cluster is close to 100% for the 8 nodes in the cluster, whereas it is about 75% for the Magny-Cours cluster. The impact of scaling on the energy efficiency is shown in Figure 110. For the x86+GPU cluster with almost perfect scaling the energy efficiency remains fairly constant, while for the Magny-Cours cluster the less than perfect speed-up impacts the energy efficiency negatively.

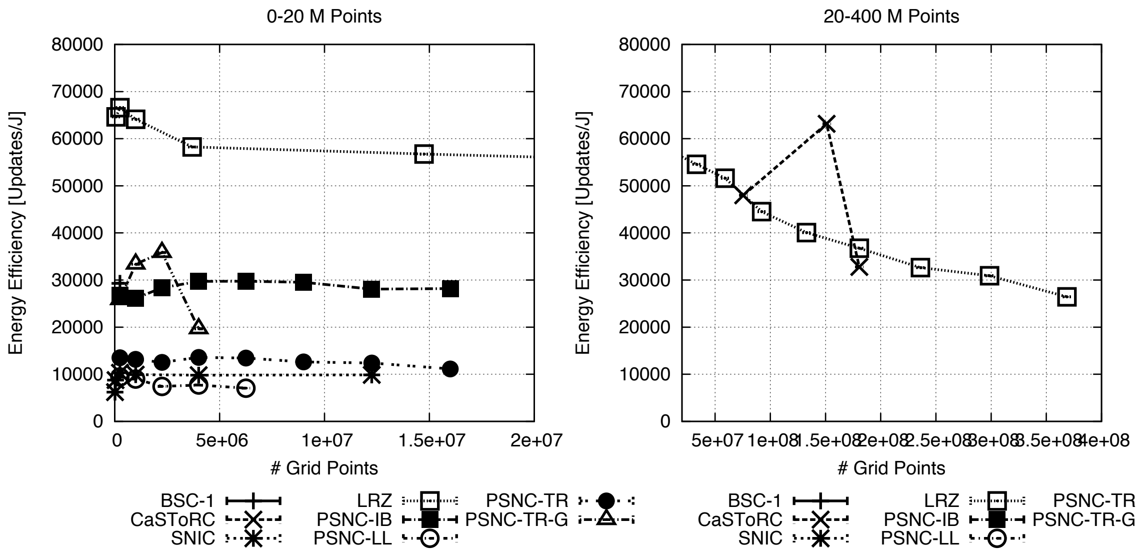


Figure 108 Energy efficiency of the Hydro benchmark measured in terms of updates per grid-point per second for a range of grid points.

The relationship between performance and power for the Hydro benchmark is shown in Figure 112. As in the case of Linpack, the performance power relationship is linear except for the x86+GPU, for which the somewhat "erratic" performance as a function of grid size does make the performance-power relationship not smooth. However, a linear relationship may perhaps also be the case for this prototype which could be proved or disproved with tests of additional grid sizes that would generate additional statistics for the performance grid size relationship.

Some detailed results for the x86+GPU prototype are shown in Figure 111 for grid sizes of up to 24 576 × 24 576 grid points, the maximum that fit in the GPU memory. For the benchmarking the iterations were adjusted to achieve ~3 min run-time.

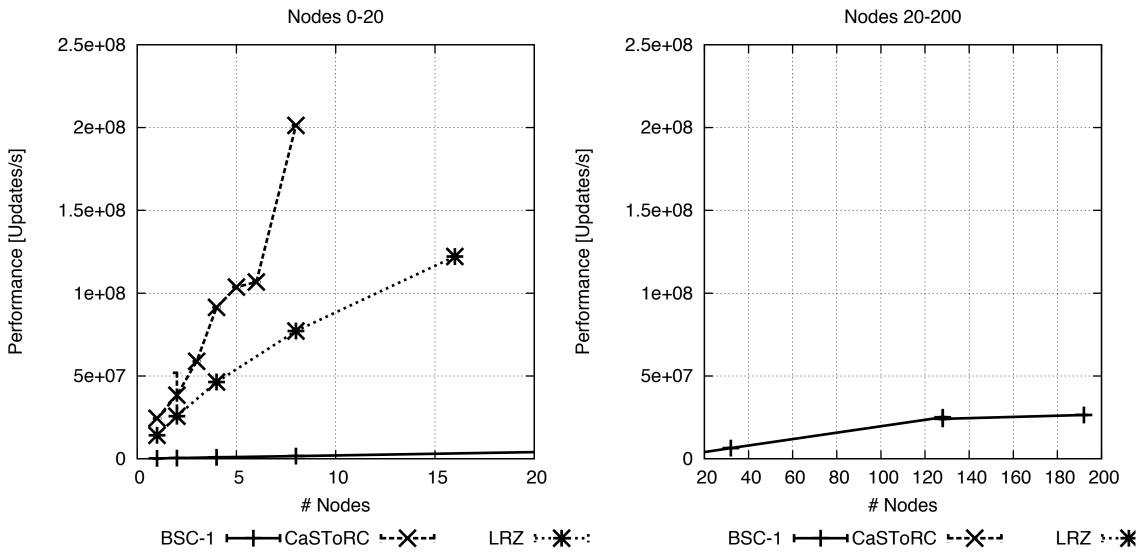


Figure 109 Scalability of the Hydro benchmark across the Tegra 2, x86+GPU, and Magny-Cours cluster measured as grid point updates/sec.

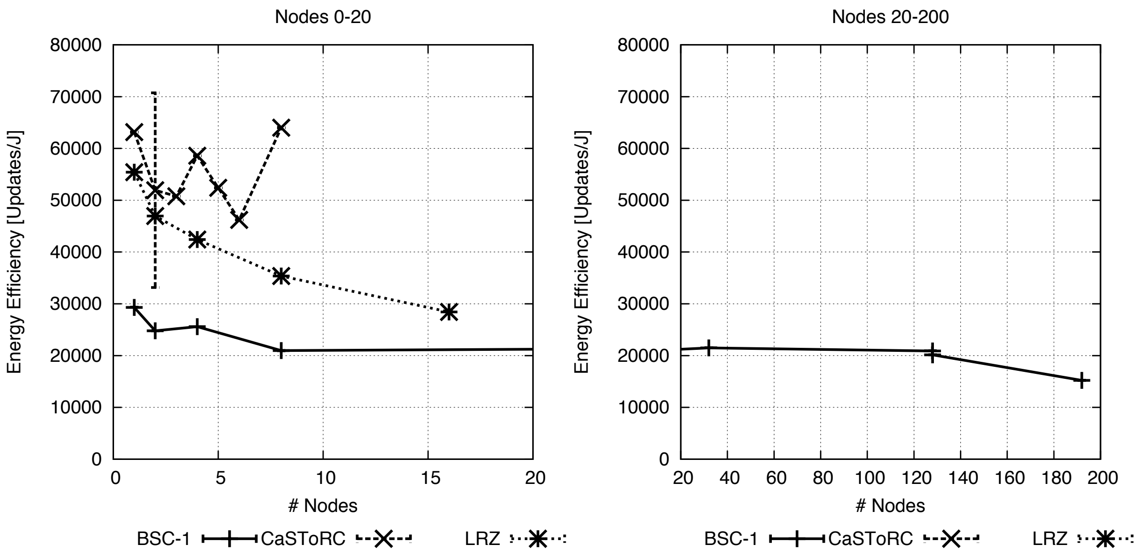


Figure 110 Scalability of the energy efficiency of the Hydro benchmark across the Tegra 2, x86+GPU, and Magny-Cours nodes measured as grid-point updates/J.

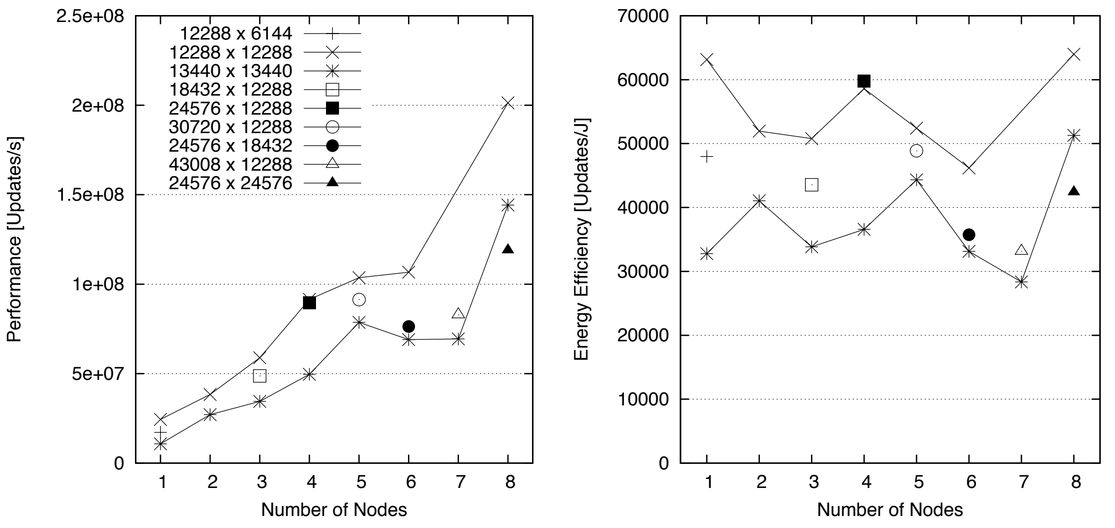


Figure 111 Hydro Performance and energy scalability for different problem sizes on the x86+GPU (CaSToRC) prototype.

6.1.7 *QCD*

GPU-GPU communication over PCIe and IB, CaSToRC

For our assessments we used the lattice QCD package QUDA, a CUDA implementation of a QCD code. QUDA is the most well established community CUDA code. It is a GPU implementation of the time-consuming kernels of typical lattice QCD calculations. It includes optimized implementations of a number of different discretizations of the continuum QCD fermion operator and a range of iterative solvers for these fermion actions. For the system benchmarking we used the MPI-version of the Domain Wall fermion operator (developed within PRACE-1IP, WP7.5). Figure 113 shows the strong scaling results of the mixed-precision solver with a global lattice volume of $16^3 \times 192 \times 8$, and the corresponding total power consumption. The implementation achieved ~14% of peak performance at a total power consumption of 4.5 kW. The energy efficiency of the QUDA implementation was about 300 MFlop/J as shown in Figure 114.

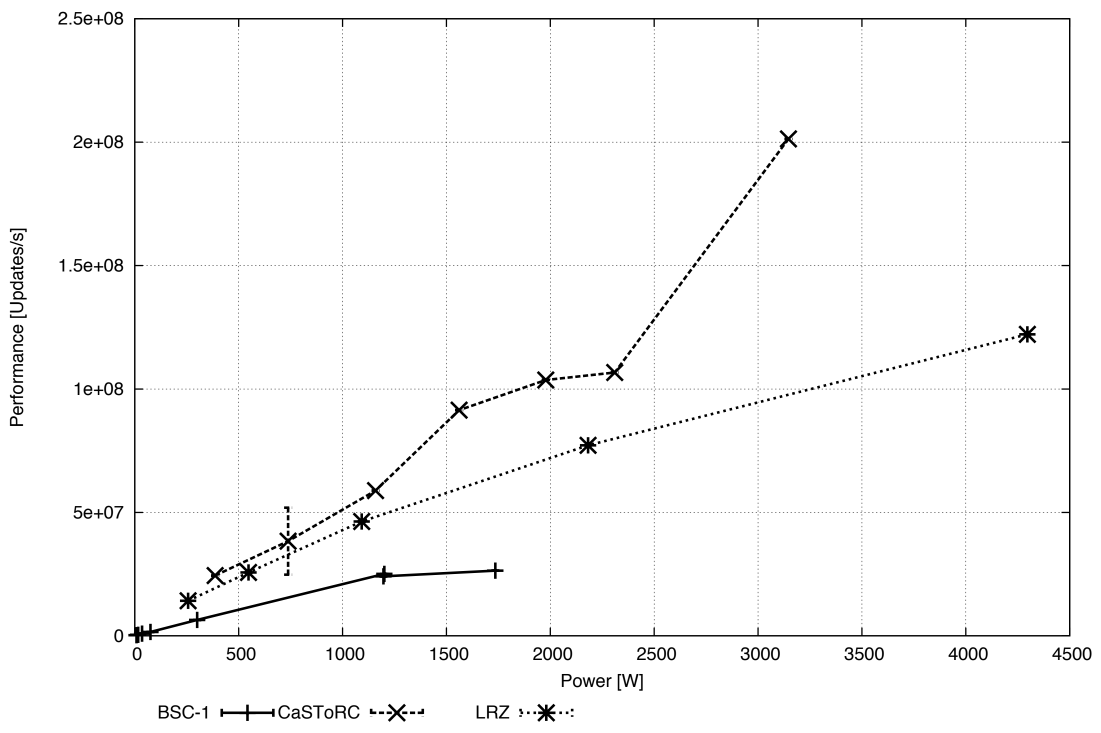


Figure 112 Performance of Hydro versus measures power for multiple node scaling runs.

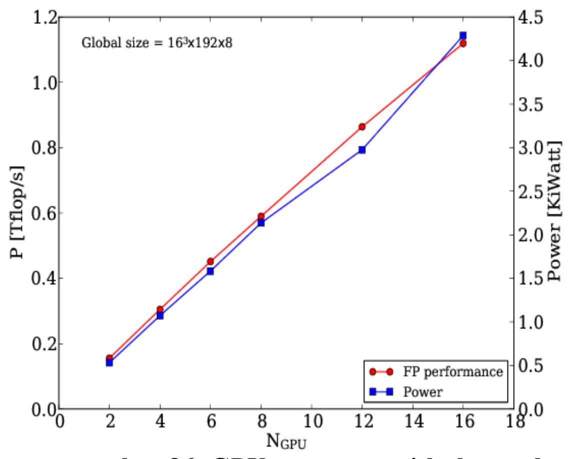


Figure 113 QUDA performance on the x86+GPU prototype with the total power consumption.

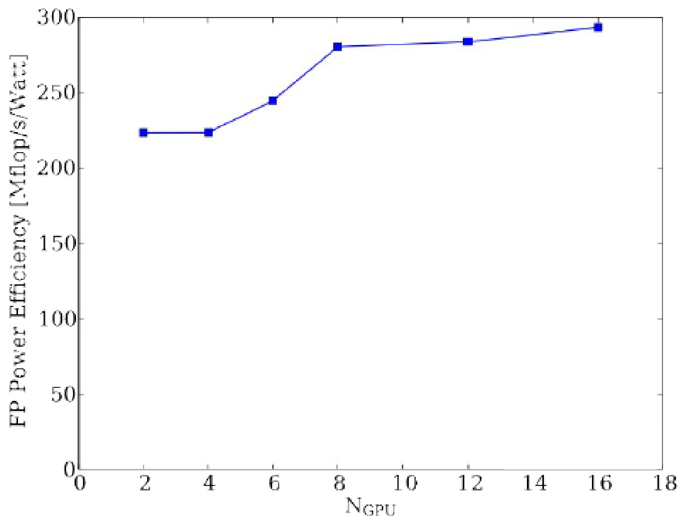


Figure 114 The energy efficiency of the QUDA code on the x86+GPU prototype based on total power consumption.

6.1.8 *Dhrystone*

An NVIDIA Tegra 2 mobile SoC based HPC cluster, BSC

The Dhrystone benchmark was only used for the Tegra 2 prototype. It is not an HPC benchmark, but is commonly used to gauge the performance of applications for other domains, and ARM Cortex-A9 Dhrystone results can easily be found. The objective of this benchmark was to validate that the nodes function correctly through comparison with published Dhrystone benchmark data. For comparison, Dhrystone was also run on a laptop Intel Core i7 processor. The same working set size were used on both platforms which have GNU/Linux OS and the GCC 4.6 compiler installed. Power consumption is measured at the AC socket connection point for both platforms, and energy consumption calculated by integrating power samples. The results are summarized in Table 28.

Platform	Dhrystone		
	DMIPS	energy	
		J	norm
Core i7	19246	116.8	1.056
Cortex A9	2466	110.8	1.0

Table 28 Dhrystone performance, measured in Dhrystone MIPS (DMIPS), for the Tegra2 prototype node in comparison with an i7 node.

The results show that the Tegra 2 Cortex-A9 (BSC-1) achieves the expected peak DMIPS, and that it is 7.8x slower than the Core i7. Normalizing the results with clock frequency results in the Tegra 2 having 2.79x lower performance/MHz. The Tegra 2 uses 5% less energy, so it is more energy efficient for the Dhrystone benchmark than the Intel i7 mobile CPU used.

6.2 I/O

6.2.1 *IOR Benchmark Results*

Exascale I/O, CEA/CINES

The two generations of tested ClusterStor hardware have very similar power consumption as shown in Table 29, with a slight power increase for the new CS6000 controllers, probably due to the increased TDP of the Sandy Bridge processor (70W vs. 65W for the Jasper Forest).

Model	Active Power (Idle)	Active Power (Max, Lustre Read)	Active Power (Max, Lustre Write)
CS3000	926 W	1052 W	1133 W
CS6000	942 W	1097 W	1134 W

Table 29 Active Power consumption of one Xyratex SSU measured at CEA.

Figure 115 shows the active power consumption of one SSU (84 drives and 2 servers) when the system is idle and during an IOR benchmark (*write*, then *read*). *Write* operations cause a power increase of about 150 W (up to 192 W), and *read* operations an increase of about 130 W (up to 155 W).

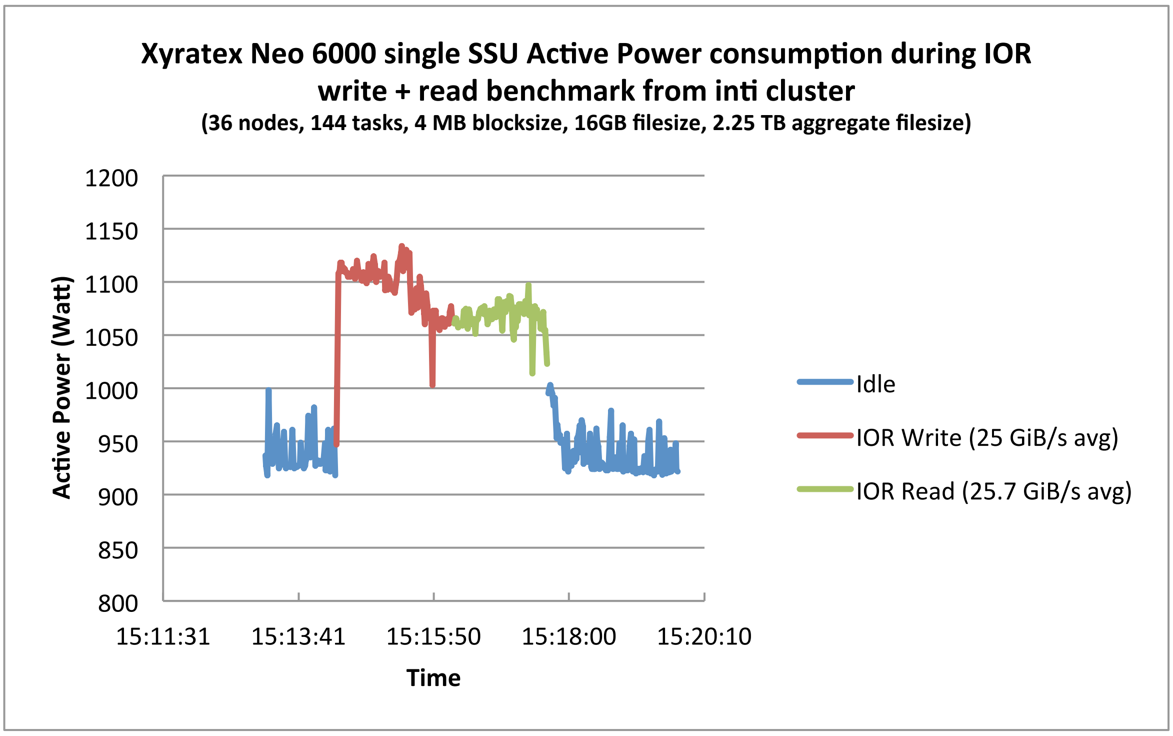


Figure 115 Xyratex CS6000 SSU Activer Power in idle state and during an IOR benchmark.

Table 30 summarizes the performance per watt (MB/J) measured on Xyratex CS3000 and CS6000 Scalable Storage Unit for *read* and *write*.

After a file system installation at CEA it is validated through a few benchmarks such as, **lnet_selftest**, which is a kernel module to test performance of the Lustre network, and **ost-survey**, an I/O benchmark against individual OSTs to allow for performance comparisons. Figure 116 shows quite stable **ost-survey** results for the CEA ClusterStor 3000 system (first full configuration of 9 SSUs).

Model	Lustre READ	Lustre WRITE
CS3000 SSU	2.39 MB/J	2.76 MB/J
CS6000 SSU	2.58 MB/J	3.69 MB/J

Table 30 Performance per watt of Xyratex ClusterStor SSUs.

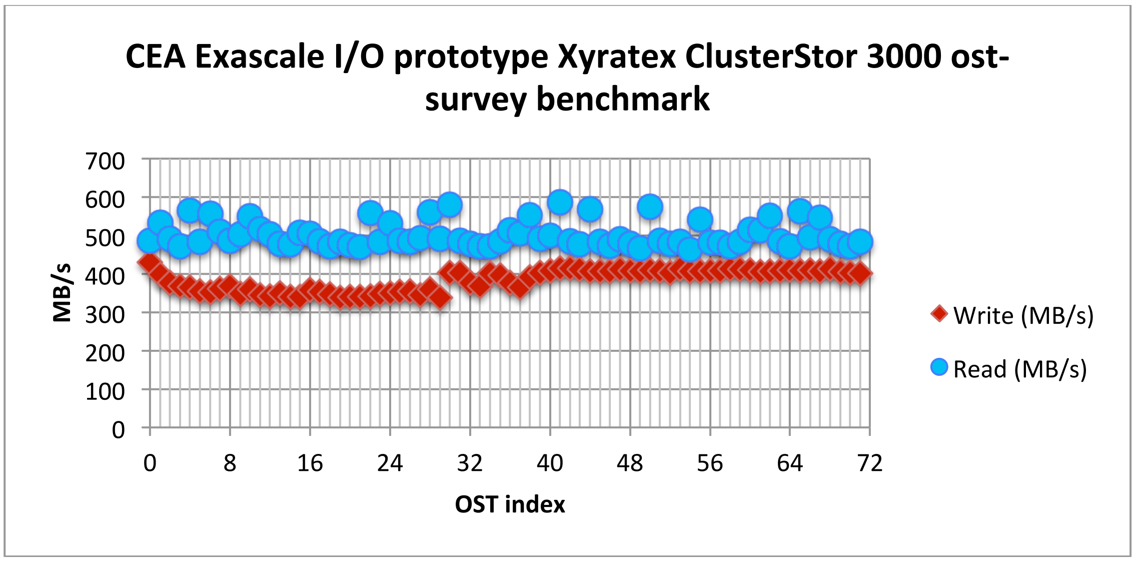


Figure 116 ost-survey Lustre benchmark results with the CS3000 system at CEA.

To get good results using the **ost-survey**, but also for better overall performance with Lustre, the value of /proc/fs/lustre/osc/*/max_rpcs_in_flight (default is 8) was increased on the CEA Inti cluster to which the Xyratex system is connected. The increased setting allows more concurrent RPCs in flight from a Lustre client to the server. Figure 117 shows the effect of changing max_rpcs_in_flight on the CS3000.

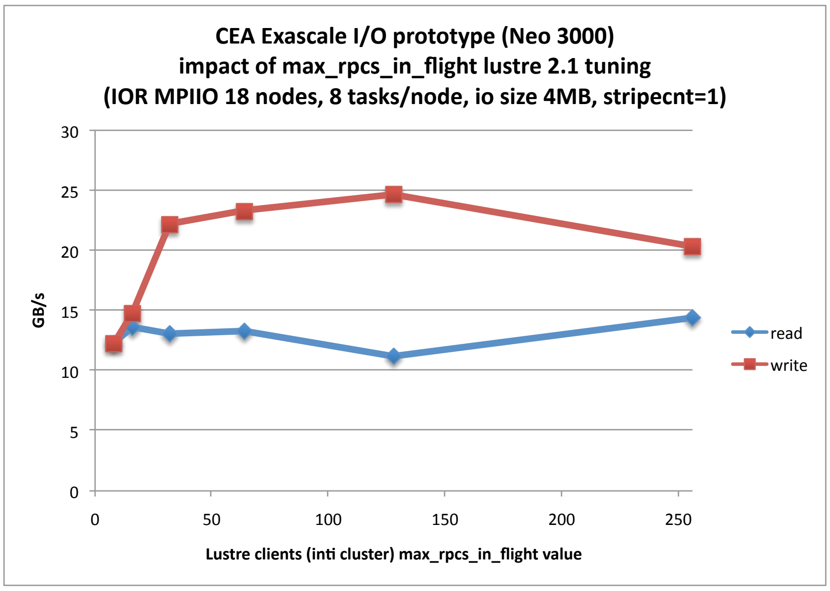


Figure 117 Impact of max_rpcs_in_flight value on I/O performance.

Figure 118 shows IOR benchmark results for the CEA ClusterStor 3000 system (9 SSUs), using the Inti cluster. The *read* performance was not good compared to *write* performance. This issue was also seen at the CINES site and was not resolved.

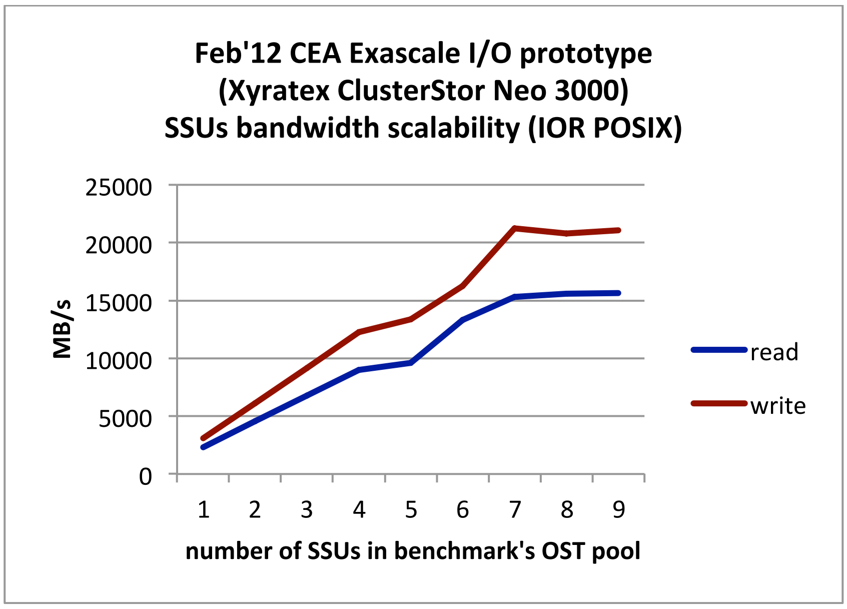


Figure 118 CS3000 IOR POSIX benchmark.

Figure 119 shows IOR benchmark results for the CEA ClusterStor 6000 system using the Inti cluster. Overall I/O performance is better than for the CS3000 system. Also, *read* and *write* benchmark results are now similar.

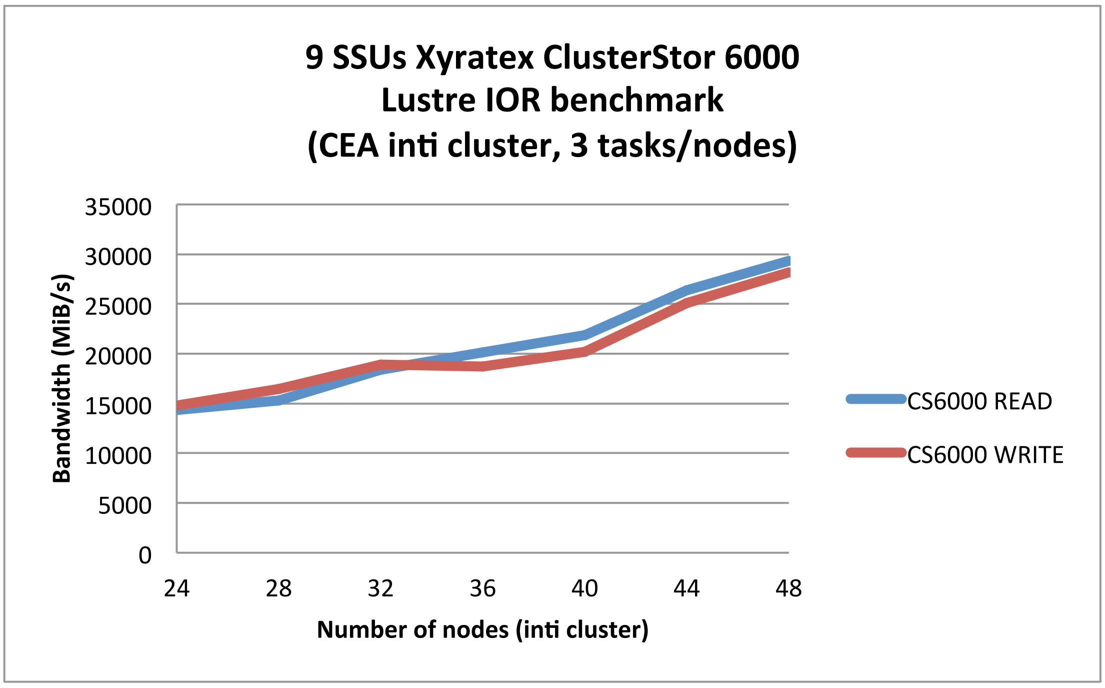


Figure 119 CS6000 IOR POSIX benchmark.

Exascale integrated I/O subsytem, FZJ

For the FZJ prototyoe eight Fusion-io cards on four nodes were used to create a GPFS file system only using the flash cards. This file system was mounted by the JUQUEEN BG/Q system. The configuration was designed to achieve a maximum I/O bandwidth of 8×1.5 GB/s ~ 12 GB/s. IOR was running on 512 compute nodes (one thread per node) using a transfer size of 4 MB. The results are shown in Table 31.

	Bandwidth
GPFS WRITE	1929 MB/s
GPFS READ	13597 MB/s

Table 31 GPFS *write* and *read* performance using 8 Fusion-io cards on 4 nodes.

The *read* performance is good but the *write* performance is not as expected. Two possible reasons for the observed performance are: 1) at the time when the bandwidth was measured, the BG/Q software stack had problems in writing to GPFS file systems, 2) the Fusion-io driver itself. The first reason is responsible for about 50% of the performance impact. Figure 120 shows the observed bandwidth during IOR *write* and *read* opertions for various subsystems on the IO-node. It is apparent that about 1 GB/s of data is read from the Fusion-io cards (fio read, red) during most of the *write* operations. At the same time about 1.7 GB/s of data are written to the card (fio write, green). The data received via the network (netstat rcv, magenta) matches the difference between the read and write bandwidth to the Fusion-io cards, leading to the hypothesis that the driver started to reorganize data on the Fusion-io cards during the *write* operations. During *read* operations the Fusion-io bandwidth matches the outgoing network traffic bandwidth (netstat-write, blue) as expected. More details are discussed in the SIONlib result section 6.2.2.

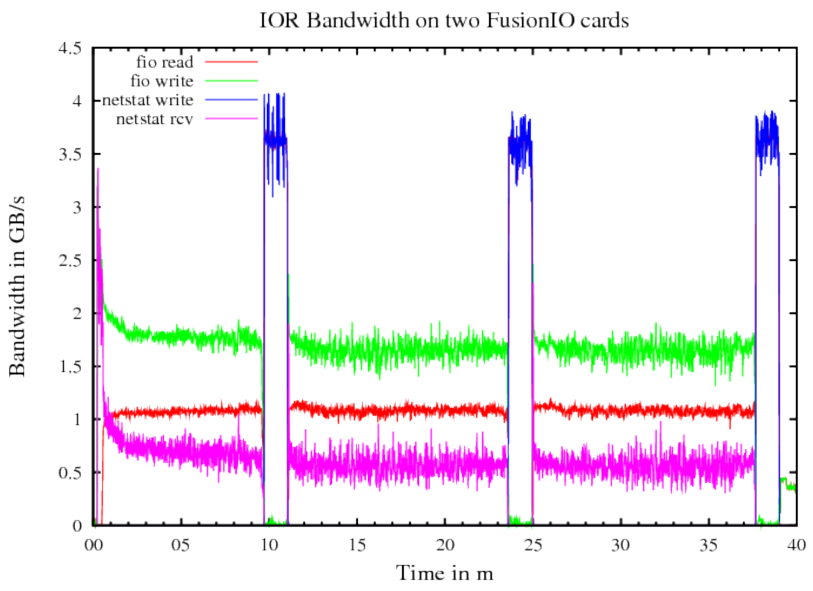


Figure 120 GPFS bandwidth measurements for the IOR benchmark on two Fusion-io cards.

The power measurements were done on two levels. At the node level PDU readings were used (Each node with two Fusion-io flash cards installed is connected to one power outlet on the PDU.) Figure 121 shows the power consumption during several IOR benchmark runs. The power gap between the idle and non-idle states is about 30 W.

For a more detailed understanding of the power consumption power readings were taken from the flash devices themselves. Figure 122 shows that the two cards together consume 24 W in idle state, but when used for I/O 60 W may be consumed. Different behaviours can be observed between *read* and *write* access. In the *read* section of the IOR benchmark, the power consumption of both cards together is more or less constant at 47 W,but in the during *write* there is a huge variance.

The conclusion of this power measurement is that for a 3 GB/s I/O bandwidth by using a standard x86 platform with two Fusion-io devices, a node requires a power consumption of up to about 130 W out of which the two Fusion-io cards need up to 60 W.

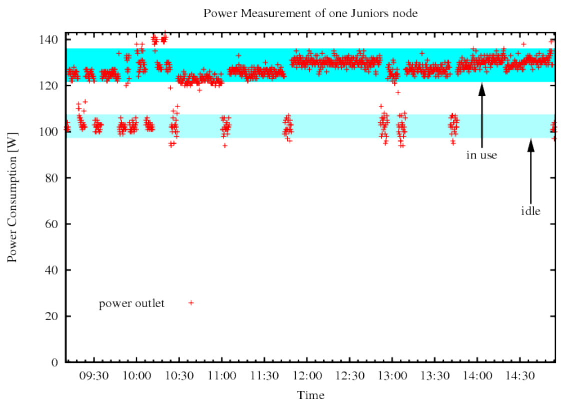


Figure 121 Power consumption of one I/O node with two Fusion-io cards as read from PDU.

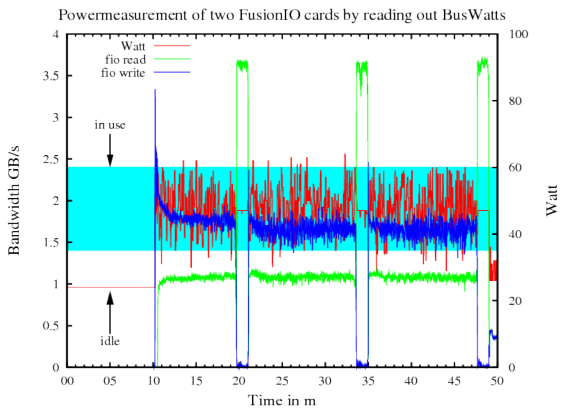


Figure 122 Power consumption of two Fusion-io cards as read from the cards.

6.2.2 *SIONlib Bencmark Results*

Exascale I/O, CEA/CINES

The SIONlib benchmark was used to investigate the scaling of the Exascale I/O prototypes. The focus was on shared and task local I/O using the ANSI and POSIX interface. For optimization in the shared file mode the number of files was increased using one file per node and the *lustre stripe count* was also increased from 1 to 36 at 112 nodes (896 tasks).

The measurement results show that 16 GB/s can be achieved for *read*. There is no significant differences between ANSI or POSIX mode and using shared or task local files.

For *write* the SIONlib reaches 24 GB/s with task local IO, Figure 123, but for shared files using the ANSI interface the *read* performance is up to 28 GB/s, Figure 124.

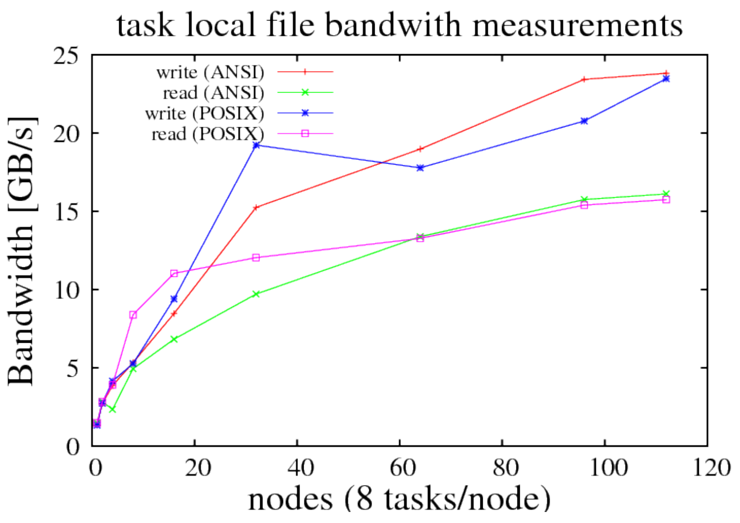


Figure 123 SIONlib benchmark results for task local files.

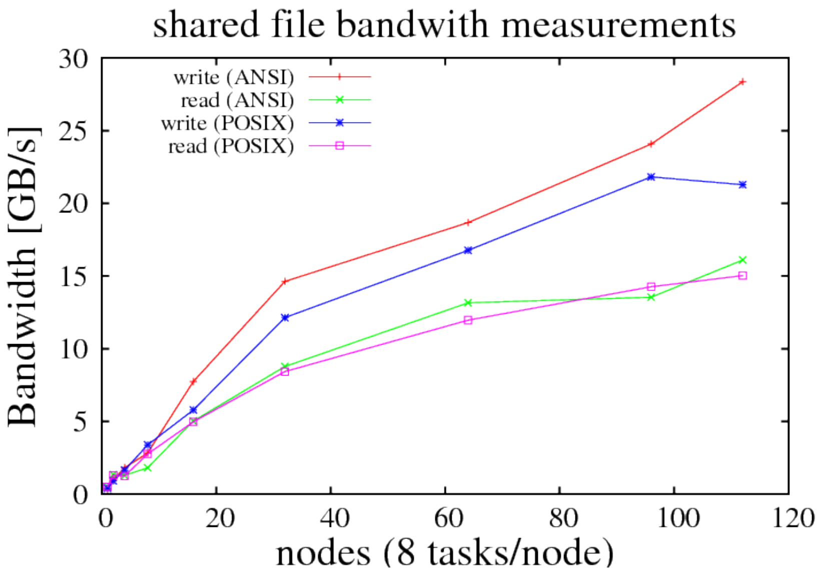


Figure 124 SIONlib benchmark results for shared files.

Exascale integrated I/O subsytem, FZJ

The same configuration was used as for the FZJ IOR test (eight Fusion-io cards on four nodes, 12 GB/s peak). For task local and shared files we used the same parameters in the SIONlib setup: 512 compute nodes each using 64 threads and 32 MB buffer size (blocksize), giving the results shown in Table 32.

	Write MB/s	Read MB/s
Shared files	3620	12379
Task local files	3288	13056

Table 32 SIONlib *write* and *read* results for Fusion-io nodes at FZJ.

The results for *read* show that the maximum expected bandwidth is reachable. But for *write* the values are very low. We found two issues for this poor behaviour: The first is that the BG/Q software stack performance problems (mentioned for the IOR benchmark) when writing to GPFS file systems is responsible for about 50% of the performance impact. The second issue was the Fusion-io driver. After several seconds (in this case at ~64 seconds) the driver starts to reorganize the data blocks on the flash device when the writing to the flash disk starts, Figure 125. This leads to a performance drop for writing the incoming data to disk.

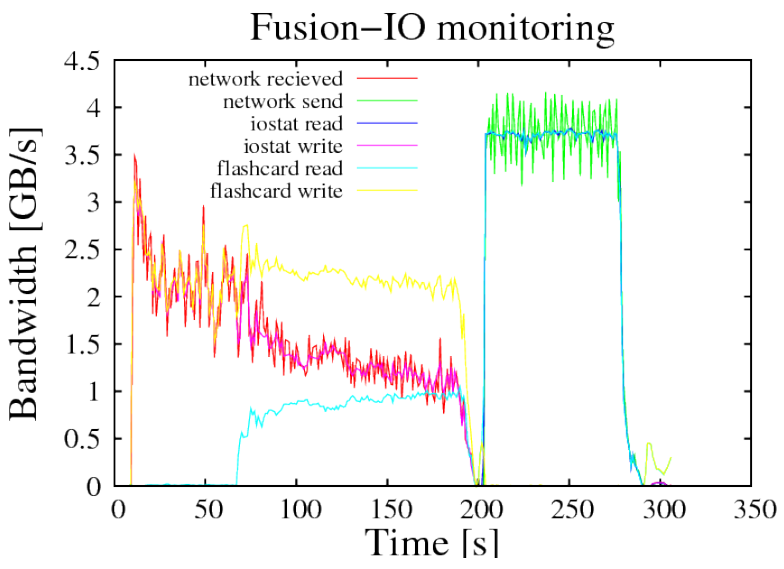


Figure 125 Fusion-io card monitoring results for detailed performance analysis.

6.2.3 *Hydro*

Exascale integrated I/O subsystem, FZJ

The Hydro benchmark was used also for assessing I/O performance of a cluster with four server nodes (8 cores per node)and four I/O nodes each with two Fusion-io cards and 10 GbE interconnection links. Table 33 shows the results obtained for grid sizes ranging from 500 x 500 to 10 000 x 10 000. The noutput parameter defines the sampling rate in time steps by which output is written. An initial output is written, except for noutput=0 for which no output is written. The maximum number of time steps for the results was set to 1000. Thus, for noutput=2000 only a single output is written while for noutput=1, an initial output followed by an output for each step for a total of 1001 outputs. For each output about 5 MB is written for the 500 x 500 grid and about 2 GB for the 10 000 x 10 000 grid. The grid points were uniformly distributed across the four servers. For the 10 000 x 10 000 case the I/O rate per server amounts to about 7.2 Gbps.

Test case nx=ny	Hydro ver. C/CU/OCL #MPI tasks	I/O (noutput=2000) Run time (s)	I/O (noutput=1) Run time (s)	No I/O (noutput=0) Run time (s)
500	32	7.762	31.442	7.870
1000	32	27.557	60.891	27.860
2000	32	109.222	166.986	106.663
5000	32	713.219	905.420	797.331
10000	32	2985.870	3558.290	3157.874

Table 33 Timing results for the Hydro benchmark on a cluster of four servers with four I/O nodes each with two Fusion-io cards and 10 GbE interconnection.

Figure 126 shows the timing results for the different grid dimensions and different I/O settings. Figure 127 shows the relative I/O volume and time increase as grid dimensions are increased from 500 x 500 to 10 000 x 10 000, and Figure 128 shows the data rate per server.

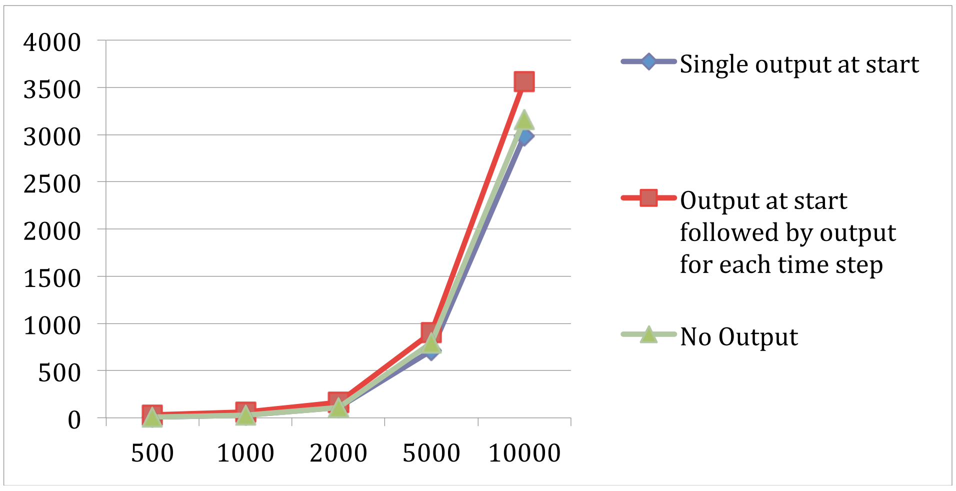


Figure 126 Timing results for the Hydro benchmark on a cluster of four servers with four I/O nodes each with two Fusion-io cards and 10 GbE interconnection.

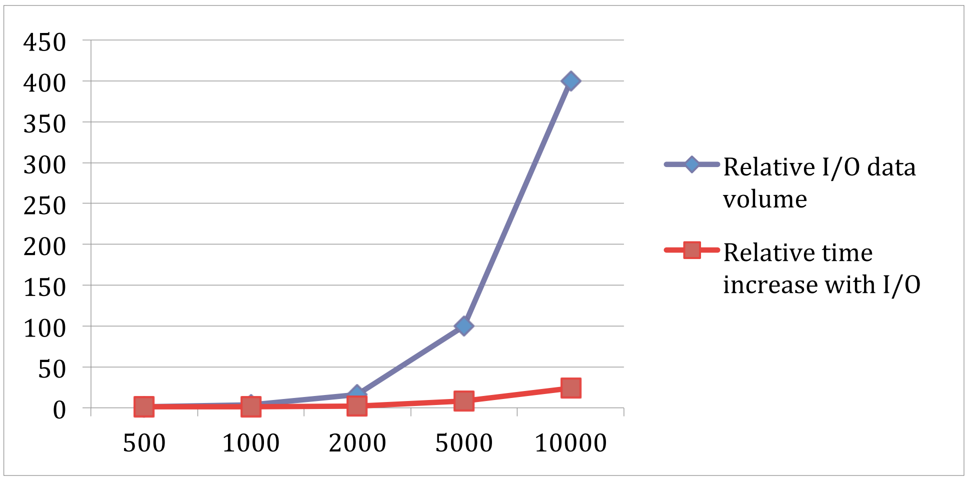


Figure 127 Scalability of data volume and incremental time for I/O for the Hydro benchmark on a cluster of four servers with four I/O nodes each with two Fusion-io cards and 10 GbE interconnection.

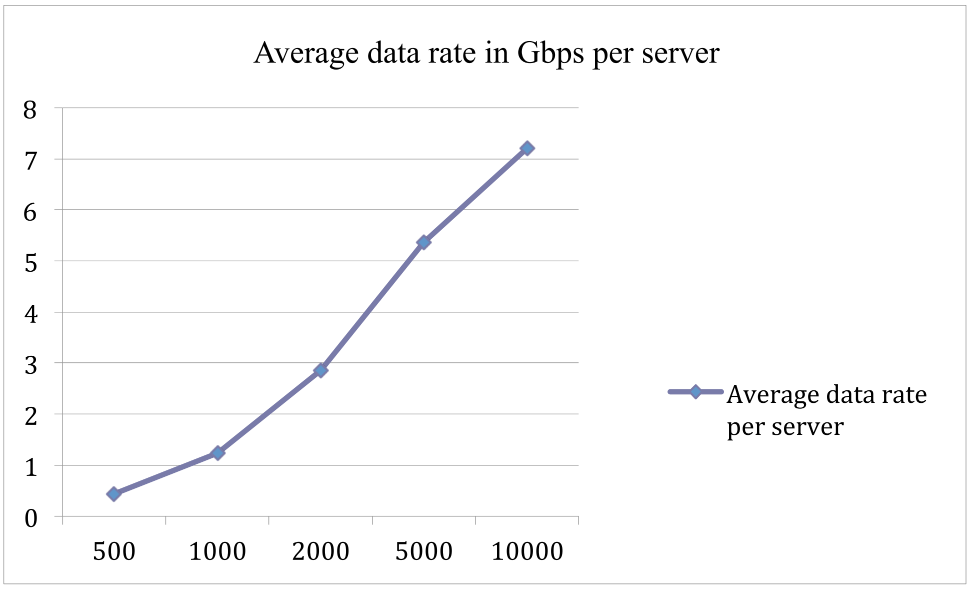


Figure 128 Average data rate per server for I/O for the Hydro benchmark on a cluster of four servers with four I/O nodes each with two Fusion-io cards and 10 GbE interconnection.

6.2.4 *Metadata benchmarks*

Exascale I/O, CEA/CINES

HLRS contributed metadata benchmark results for the CEA/CINES prototype. Test cases were created for 2, 4 and 8 nodes and 2, 4, 8, 16 processes. The processes were distributed round-robin to the nodes and run on both the Xyratex ClusterStor 3000 and 6000 systems. From the results for two nodes shown in Figure 129, Figure 130 and Figure 131, it can be seen that in general ClusterStor 6000 performs better than ClusterStor 3000, both per node and in total. This is especially the case for a small number of processes, where the ClusterStor 6000 can handle approximately 1.5 times more I/O operations than ClusterStor 3000. This behaviour holds for *create*, *stat* as well as for *remove* as shown in Figure 129 – Figure 131. For *create* with 2 processes, the file operations ratio is 900/700 = 1.28, Figure 129, for *stat* the ratio is 2000/1500 = 1.30, Figure 130, and for *remove* the ratio is 1500/800=1,88, Figure 131. For higher number of processes, the I/O-operations per second of ClusterStor 3000 and ClusterStor 6000 converge - the benefit of the upgraded prototype diminishes with an increased number of processes, except for *stat.*

Figure 132 depicts the results for *create* for 2, 4 and 8 nodes and 2, 4, 8, and 16 processes.

The behaviour is qualitatively the same for 4 and 8 nodes as for 2 nodes. The total number of I/O operations for all processes follows the same trends as the result for individual processes, Figure 133. From these results it is clear that the Xyratex ClusterStor 6000 performs better than ClusterStor 3000, in particular for few processes. Figure 134 and Figure 135 show the total I/O operations for *stat* and *remove*, respectively. A detailed review of Figure 133 – Figure 135 reveals that the I/O rate remains fairly constant if the number of processes equals the number of nodes, i.e., for one process per node. With multiple processes per node, the total number of I/O operations per second decreases significantly, with some exceptions.

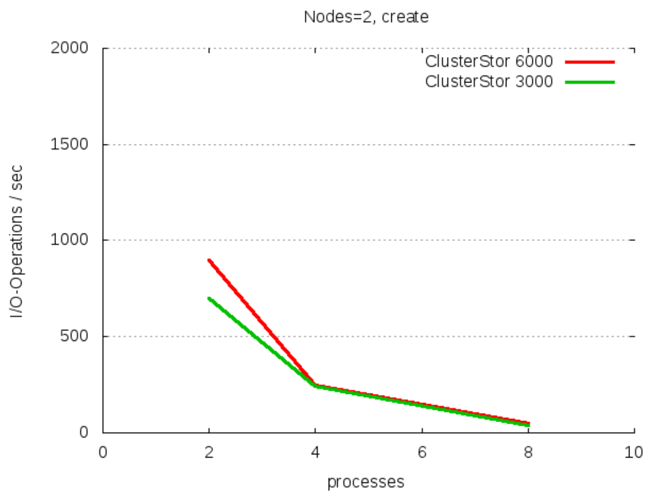


Figure 129 I/O-Operations/sec per process, for *create* on two nodes.

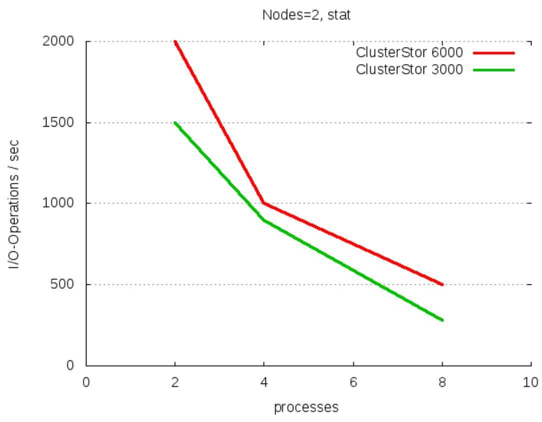


Figure 130 I/O-Operations/sec per process, for *stat* on two nodes.

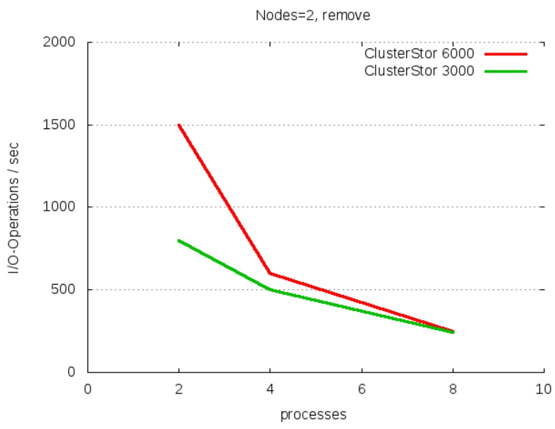


Figure 131 I/O-Operations/sec per process, for *remove* on two nodes.

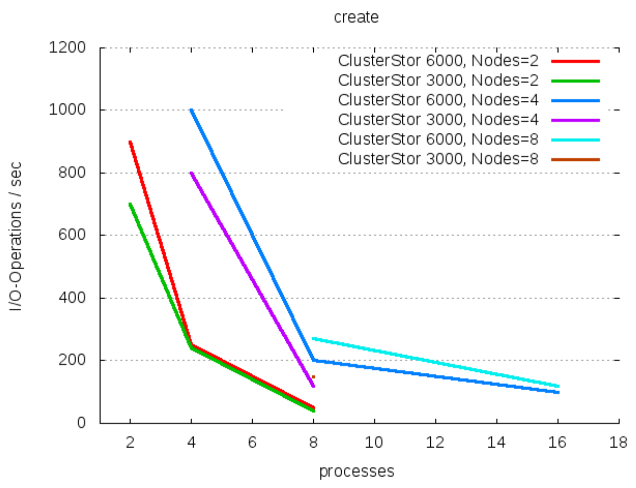


Figure 132 I/O-Operations/sec per process, for *create* on two, four and eight nodes.

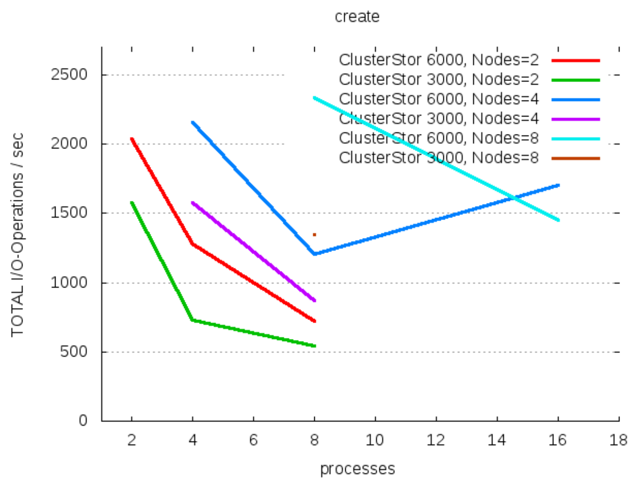


Figure 133 Total I/O-Operations/sec, for *create* on two, four and eight nodes.

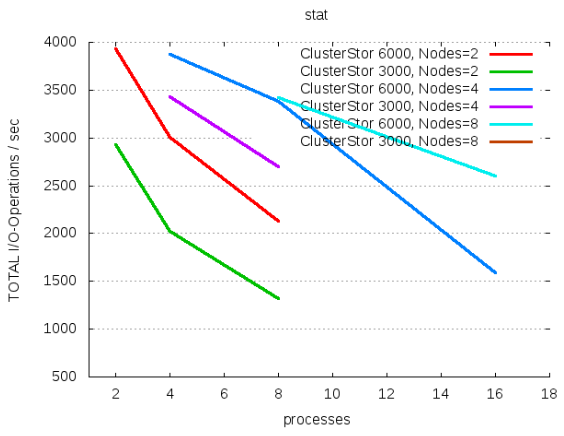


Figure 134 Total I/O-Operations/sec, for *stat* on two, four and eight nodes.

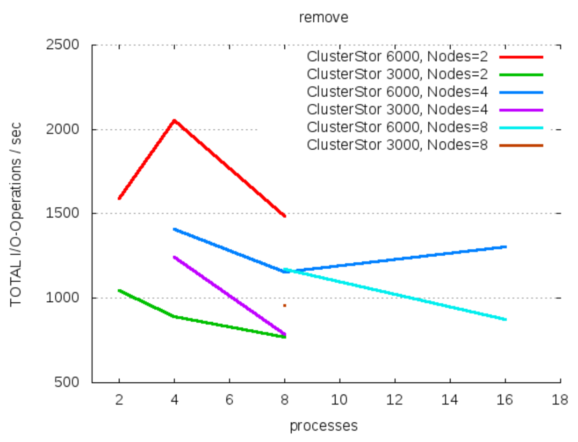


Figure 135 Total I/O-Operations/sec, for *remove* on two, four and eight nodes.

For *create* the maximum number of total I/O operations/second is about 2350 for 8 processes and 8 nodes, Figure 133. For fewer nodes with one process each, the maximum I/O operations per second decreases somewhat but not significantly. From the measurements it is clear that for the configurations tested the maximum *create* rate for the ClusterStor 6000 is well below 2500 operations/second. Comparing measurements with one and two processes per node, a dramatic decrease in total I/O performance is apparent for two processes per node regardless of the number of nodes. Comparing the results for different number of nodes but a constant number of processes, it is plausible that either the node or the local network is the limiting factor.

For *stat*, Figure 134, the same tendency holds as for *create*: The maximum of about 4000 I/O operations/s is measured for two nodes with one processes each. For 4 nodes with one process each only a minor decrease occurs. For 8 nodes with 8 processes the performance drops below 3500 operations/s. This is approximately the same performance as for 8 processes on 4 nodes. The ClusterStor 6000 performance for *stat* is decreasing with increasing number of concurrently accessing processes.

The *remove* operation, Figure 135, shows a very interesting behaviour. The performance for 2 nodes is better than for 4 nodes, which is equal or better than the performance for 8 nodes. For 4 processes on 2 nodes, we measure the best total performance at slightly more than 2000 *remove* operations/s.

6.2.5 *Scanning large file system issues*

Exascale I/O, CEA/CINES

In order to make good backup policy decisions, it is necessary to examine each object in the file system. Figure 136 shows the architecture of the CINES data management and back-up system. File system *scan* operations are critical for good back-up polocies and has been evaluated for global mode operation and for differential update.

The CINES storage architecture is built around the Data Infiniband Fabric. Data is generated on the local Ambre cluster, and written to the Lustre file system (pictured in the center of Figure 136). The Lustre file system relies on a Xyratex Neo3000 appliance able to handle 115 TB of data with a bandwidth of 3 GB/s.

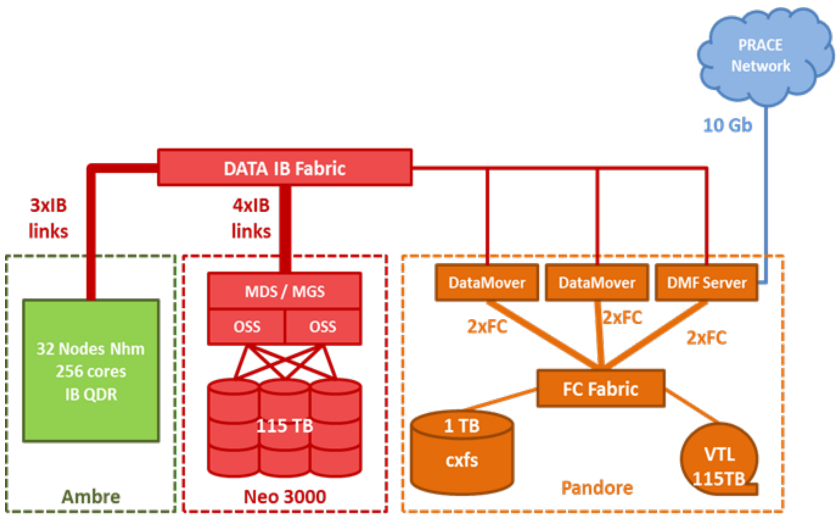


Figure 136 Data management and back-up system for the CINES prototype.

Data on this Lustre file system is automatically secured on another file system: Pandore. It relies on two storage pools: a fast one (CXFS disks), and a slower higher capacity Virtual Tape Library (VTL). The VTL, based on a Copan 400 MAID, emulates the behaviour of a tape library. These two spools are managed by a DMF server that handles data movements between the Lustre file system and the storage pools. Scans performed on the Lustre file system aim to provide DMF with lists of files that should be secured.

The Lustre file system test bed has 2.5 million objects, with a size repartition close to "real-life conditions" (described in the section "impact of object distribution" below).

	First Scan	Regular scan
rsync	~50 minutes	Idem
Robinhood	~5h	DB request, < 1s
lfs-find	~15 minutes	Idem

Table 34 Performance of various file system *scan* approaches for Lustre.

For the existing Lustre file system a first scan is required prior to regular scans performed each time securing data is required. The first scan, using the Robinhood Policy Engine (RPE), is much slower than *rsync*. The Lustre adapted command (*lfs-find*) taking advantage of the Lustre API is as expected faster than *rsync*. But, each time data need to be secured the RPE shows its superiority compared to the other methods. The RPE keeps a database that is updated based on analysis of Lustre change logs. Therefore, the RPE is able to provide a list of changed files by simply making a database request. *rsync* and *lfs-find* require a *scan* of the whole file system leading to unacceptable performance (for 100 million files a *scan* would take more than one day and a half by *rsync*, and more than 10 hours using *lfs-find*). Results from benchmarking these methods on the testbed are listed in Table 34.

One key aspect of avoiding heavy and long *scans* is the availability of the Lustre filesystem for user processing, especially for metadata traffic. The RPE also provides a partial file system *scan*, for example, for making its database consistent after a communication loss with the Lustre MDS. This avoids a complete *scan* that would require a long time. As a bonus, the RPE is also able to trigger data movement on-demand when new data is created. Thus, launching periodic data movement using a cron task is not required. This feature can be specified by setting policies according to the needs. The benefits are numerous; the highest benefit is that data synchronization does not lead to periodic overloads of the data movers. The benefits of using changelogs for file management has also been recognized by the Lustre developers and has been introduced in Lustre v2.0 in *lustre_rsync*.

Highlight of Robinhood Policy Engine processing

The Robinhood Policy Engine performs administration procedures, such as deleting temporary files, raising alerts on unsuitable behaviour, or backing up data. The design supports parallel processing. Figure 137 gives an architecture overview [RPE].

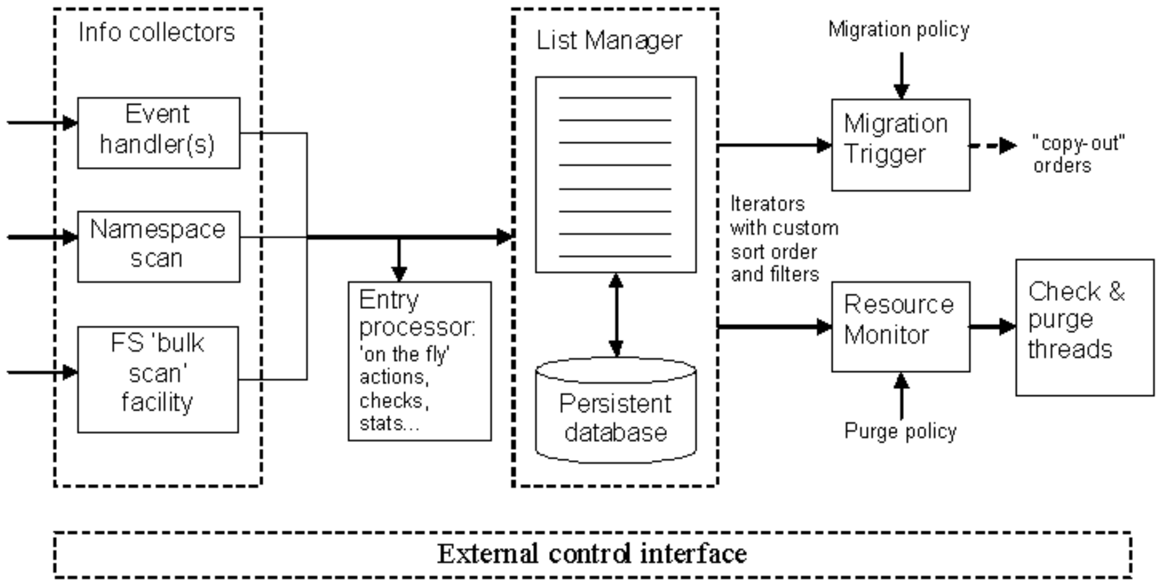


Figure 137 The Robinhood Policy Engine architecture.

The Robinhood Policy Engine triggers jobs depending on policies based on objects' size, ownership and timestamps. For the sake of performance and to reduce the load on the managed file-system, after an initial scan, it collects the necessary information from Lustre changelogs (since Lustre V2). The data is saved into a persistent database (MySQL), which offers an up-to-date view of the file-system. With RPE each request for statistical purposes or quota check doesn't cost much more than a database query. With RPE migrations are quicker than with classical tools; there is no need to perform heavy *scans* of source and target directories because of RPE data base.. It should also be noticed that a customizable shell-script is triggered for every migration copy. It gives the ability to change the copy tools used to perform the data movement. Any copy command can be selected, including any specific data management software command, such as dmarchive from DMF.

Next step: HSM integration

An early HSM integration has been performed on the Exascale I/O CINES configuration. The development led by a CEA team is integrated as follow:

PANDORE (DMF Server & DMF Data Movers)

- SuSe SLES 11 SP 1
- Lustre 2.1 + *Module HSM CEA*

Ambre (Client):

- SuSe SLES 11 SP 1

Neo3000 (Lustre Server):

- Scientific Linux 6.1 Carbon
- Lustre 2.1 + *Module HSM CEA*
- Enhanced Kernel

The HSM integration add-on significantly extends the functionality in the following *lfs* control commands: hsm_state, hsm_set, hsm_clear, hsm_archive, hsm_restore, hsm_release, hsm_remove, hsm_cancel.

Sample screens:

```
pandore:/lustre_fs/ExaScale-IO-WP9 # mount
…
192.168.22.40@o2ib0:192.168.22.41@o2ib0:/fs1 on /lustre_fs type lustre

pandore:/lustre_fs/ExaScale-IO-WP9 # ls -l
total 44000640
-rw-r--r-- 1 root root 10737418240 Apr  6 18:13 file1-11GB
-rw-r--r-- 1 root root 23581450240 Apr  6 18:23 file1-22GB
-rw-r--r-- 1 root root 10737418240 Apr  6 18:16 file2-11GB

pandore:/lustre_fs/ExaScale-IO-WP9 # lfs hsm_state *
file1-11GB
        states: (0x00000000)
file1-22GB
        states: (0x00000000)
file2-11GB
        states: (0x00000000)
pandore:/lustre_fs/ExaScale-IO-WP9 # lfs hsm_archive file2-11GB
pandore:/lustre_fs/ExaScale-IO-WP9 # lfs hsm_state file2-11GB
file2-11GB
        states: (0x00000001) exists
        action: ARCHIVE is running, 0x23d500000 Bytes moved → ONGOING MIGRATION
pandore:/lustre_fs/ExaScale-IO-WP9 # lfs hsm_state file2-11GB
file2-11GB
    states: (0x00000009) exists archived                  → MIGRATION ENDED
pandore:/lustre_fs/ExaScale-IO-WP9 # du -sh *
11G     file1-11GB
22G     file1-22GB
11G     file2-11GB                                        → DATA STILL ON DISK
pandore:/lustre_fs/ExaScale-IO-WP9 # lfs hsm_release file2-11GB
pandore:/lustre_fs/ExaScale-IO-WP9 # du -sh *
11G     file1-11GB
22G     file1-22GB
0       file2-11GB                                     → SPACE RELEASED ON DISK
pandore:/lustre_fs/ExaScale-IO-WP9 # cat file2-11GB > /dev/null
pandore:/lustre_fs/ExaScale-IO-WP9 # du -sh *
11G     file1-11GB
22G     file1-22GB
11G     file2-11GB                                     → AUTOMATIC RECALL
pandore:/lustre_fs/ExaScale-IO-WP9 # lfs hsm_state *
file1-11GB
        states: (0x00000000)
file1-22GB
        states: (0x00000000)
file2-11GB
        states: (0x00000009) exists archived        → DATA RESIDES ON BOTH  LEVELS
```

This feature is expected in Lustre 2.4 or 2.5 by 2014. Any HSM software compliant with the Lustre HSM specification will be able to manage any Lustre file system such as DMF/SGI.

CINES object distribution

Object statistics from the GENCI Tier-1 centre CINES' JADE facility is a good example of a real production HPC centre situation. The centre supports over a thousand scientists across a wide range of disciplines that has created a large number of files of varying sizes. The file system for permanent objects contains more than 34 million files and the scratch file system at times contains more than 100 million files.

Figure 138 shows the distribution of file sizes for the 34 million files and Figure 140 shows the cumulative distribution. Of the 34 million files more than 10 million are smaller than 1 kB, more than half are less than 8 kB, and more than 95% are less than 256 MB. Large numbers of small files is a serious challenge for many file systems. The total amount of data contained in files as a function of size is shown in Figure 139 and the cumulative distribution is shown in Figure 141 adding up to more than 1.4 PB.

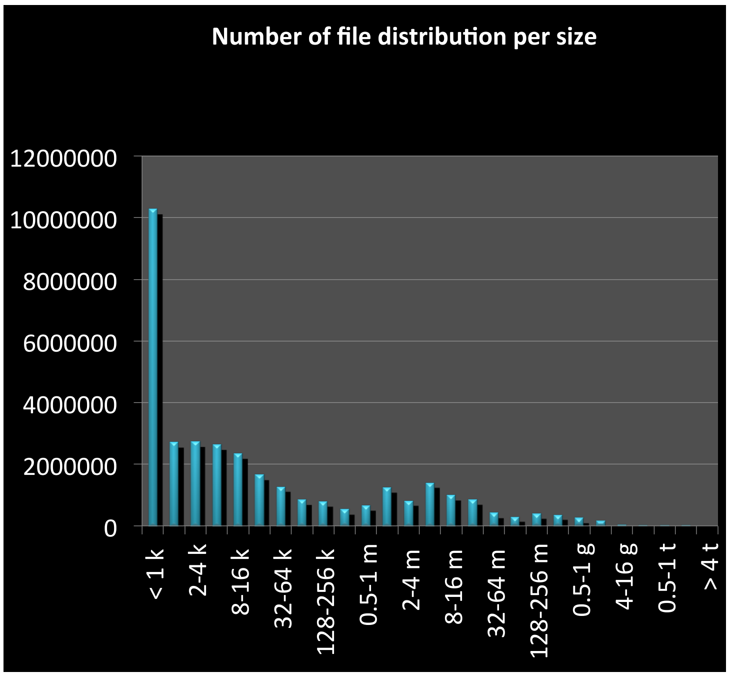


Figure 138 Distribution of file sizes for the 34 million files in the CINES file system.

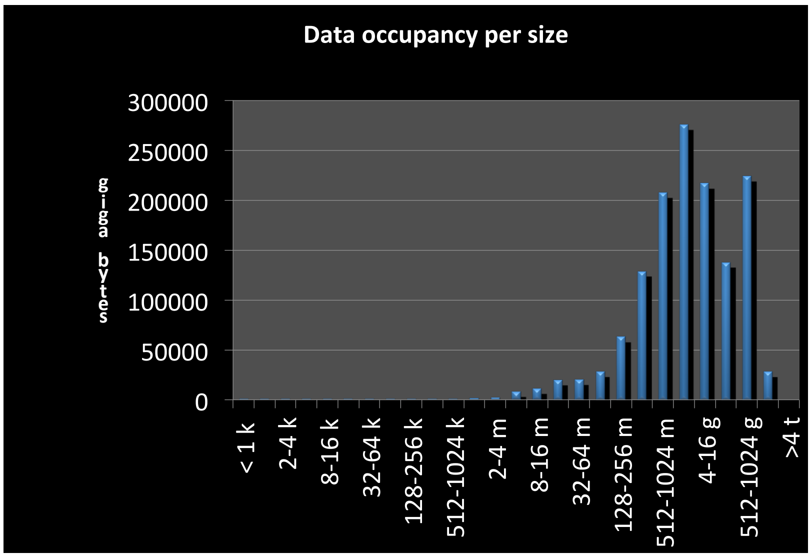


Figure 139 Total data volume as a function of file size.

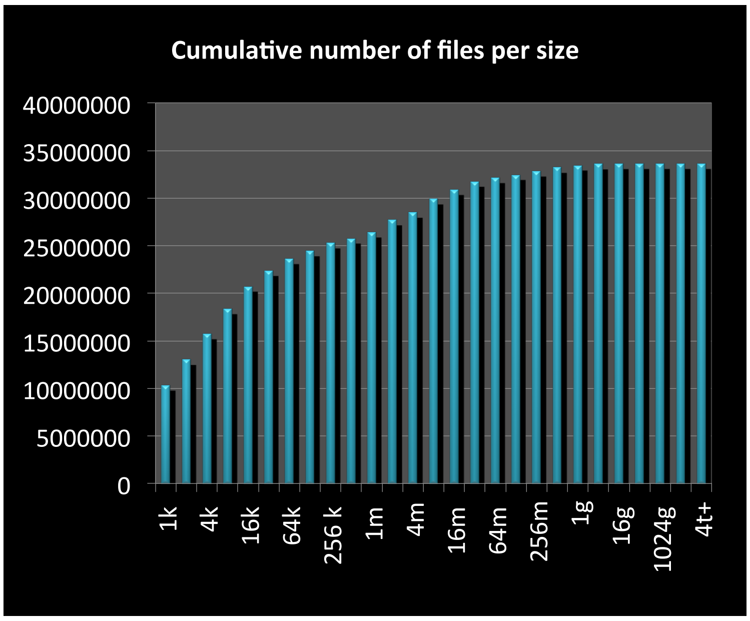


Figure 140 Cumulative distribution of files in the CINES file system.

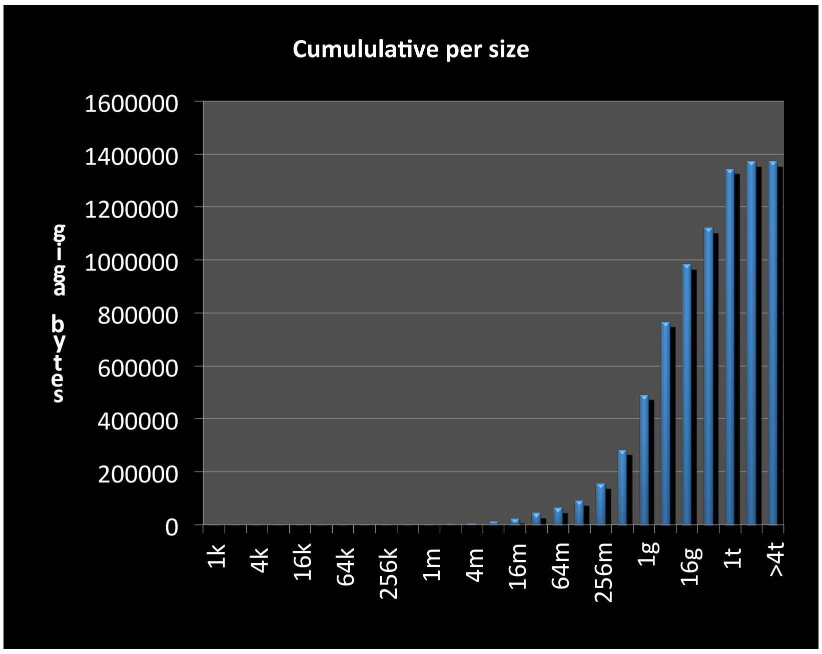


Figure 141 Cumulative distribution of data volume in files as a function of size.

Taking into account that sequential medias (tapes) have orders of magnitude higher latency than spinning disks, designing the storage system to effectively handle small files is important. MAID technology is of interest in this regard since MAID latency is significantly less than tape latency, and latency has a big impact on the effectivess of handling small files. In the CINES case 95% of the files of a size up to 256 MB account for about 10% of the 1.4 PB of the long term storage needs: it make sense to design 10% of the storage to take care of the 95% of small size objects.

6.3 Cooling

6.3.1 *Holistic approach to energy efficiency, LRZ*

Table 35 lists the observed quantities for the CooLMUC system.

Sensor	CooLMUC
Water inlet temperature (T_{in})	Yes
Water outlet temperature (T_{out})	Yes
CPU Temperature (T_{cpu})	Yes (Package)
Power Consumption per Node (P_{node})	Yes (from PDU)
Power Consumption of the entire System (P_{sys})	Yes (sum of nodes + cooling + network)

Table 35 Available measurement quantities on CooLMUC.

The evaluation strategy for the prototype is to vary the water inlet temperature while monitoring all other quantities described above. While it is desirable for an evaluation to include full load of all system components, such a setup is difficult to realize in practice. Typically, the load on the server processors is reduced during main memory operations or communication via network devices (the DSP STREAM measurements showed a decrease in core power consumption with increased DRAM power consumption) . Since the CPUs are the dominating contributors to heat generation, a system test should stress mainly the processors' pipeline systems as well as the cache systems.

The pre-compiled 64-bit mprime v2.26 benchmark was chosen to run the temperature tests. This benchmark is freely available from the mprime author's website (see [MPrime]). The program is flexible in terms of floating-point computation and main memory access pattern, performing FFTs in a repeated fashion to search for Mersenne prime numbers. Due to its heavy load on the system it has become one of the favoured system stability tests in the community of private computer users. The "torture test" is used with custom settings to stress the system. The custom setup mainly stresses the server processors, while main memory access is mild and no communication via network devices is performed.

Measurements are carried out of the quantities stated above for a range of water inlet temperatures T_{in} from 27 °C up to 50 °C using mprime. Figure 142 shows the temperature T_{cpu} as a function of the water inlet temperature T_{in}. Shown here is the average individual node CPU temperature[4]. A roughly linear increase of the core temperatures with the inlet temperature T_{in} was observed. The spread of the CPU temperatures is caused by multiple factors. A single pipe system provides the heat transfer from four (CooLMUC) CPU sockets. Consequently, CPUs that are cooled later in the chain are provided with slightly warmer water causing a higher CPU temperature. Another effect on the spread is the quality of the heat transfer from the CPUs to the water pipes: a few nodes show undesirable behaviour, i.e., the core temperatures are very high even at low water temperatures. At high water temperatures automatic CPU throttling inhibits severe damage of these chips[5]. Clearly, those anomalous nodes have to be investigated and the water pipe system has to be re-assembled at the board level.

[4] The average node core temperature is the mean of the temperature values delivered by the temperature sensors embedded into each of the processor's cores.

[5] CPU throttling sets in at a core temperature of 66 °C on the AMD Magny Cours CPUs. The average core temperature may be lower than that threshold because of the temperature spread amongst the cores.

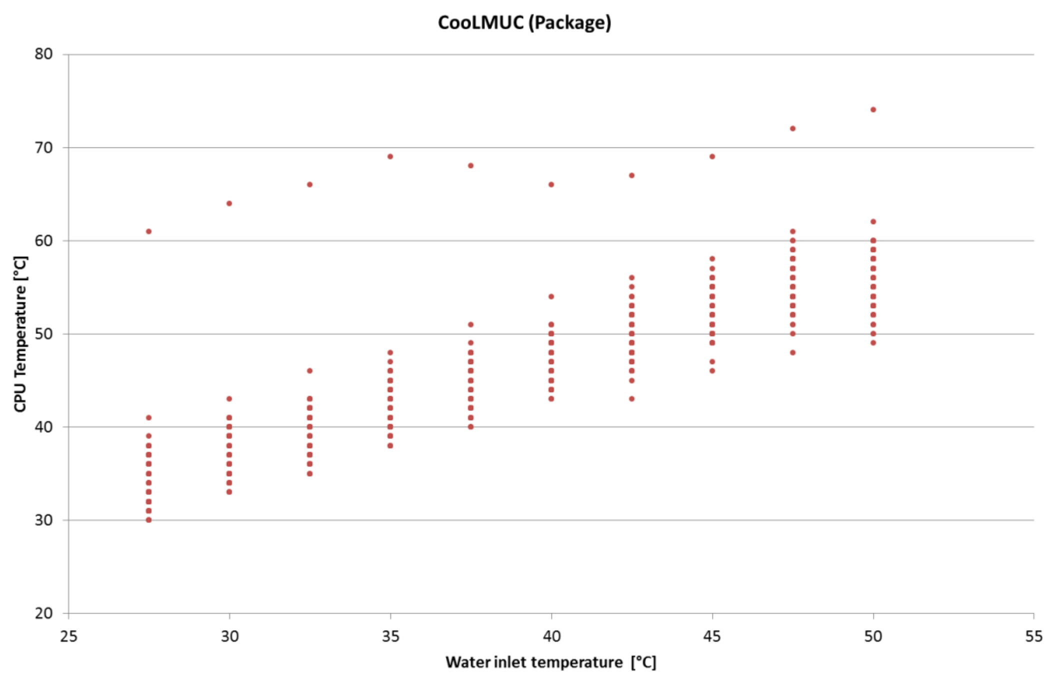


Figure 142 CPU temperatures in relation to water inlet temperature.

Figure 143 displays the CooLMUC water cooling system and building infrastructure response to different building warm water loop water temperature setpoints. The setpoint was changed from 40 °C to 50 °C, to 30 °C, to 20 °C and back to 40 °C, which is the current operating temperature. The measurements show very clearly that the building warm water loop regulation reacts relatively sluggishly, mainly related to the motorized pipe vales.

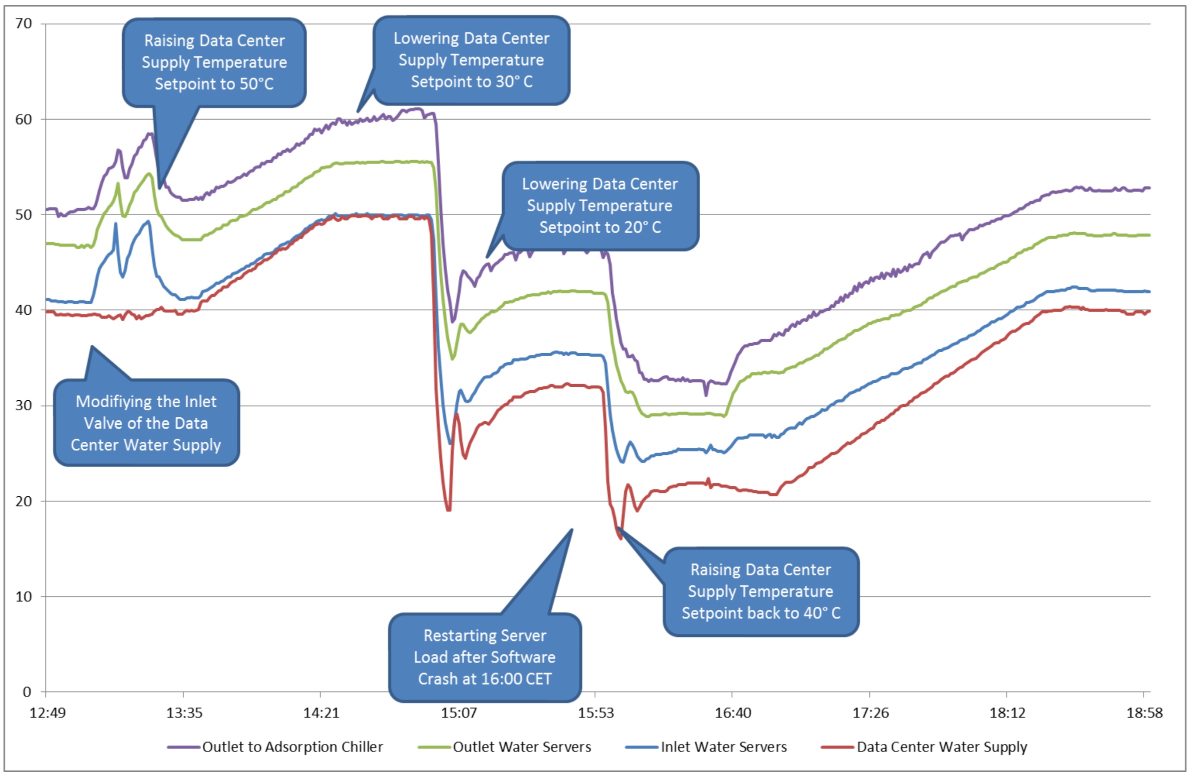


Figure 143 Water inlet setpoint and water outlet temperature response curves.

Assuming proper assembly of the cooling loops, the water inlet temperature could safely be increased even further up to about 70 °C without affecting the system performance due to down-clocking of individual processors, but the upper temperature limit is determined by the in-rack air-cooling equipment.

The air-cooling loop uses the same cooling water as the CPUs to remove the heat generated by the compressors. It further increases the temperature of the water returning from the nodes and therefore lower inlet temperatures of water are still sufficient to drive the adsorption chiller.

For energy reuse, in this particular case the operation of an adsorption chiller, the water outlet temperature should be as high as possible. When measuring the water outlet temperature as a function of the water inlet temperature an approximately linear scaling can be observed with a temperature difference between outlet and inlet of $T_{delta} = T_{out} - T_{in}$ of up to 7 °C. However, as shown in Figure 144, if the water inlet temperature is increased, the temperature difference drops below 6 °C. It is well known that at high water temperatures the amount of heat transferred to the water circuitry drops since the power dissipation of the cooling circuitry by convection depends linearly on the temperature difference between water and surrounding air. The amount of heat that is not transferred to the coolant has to be removed by the data centre's air-cooling mechanism and thus is not available for energy reuse.

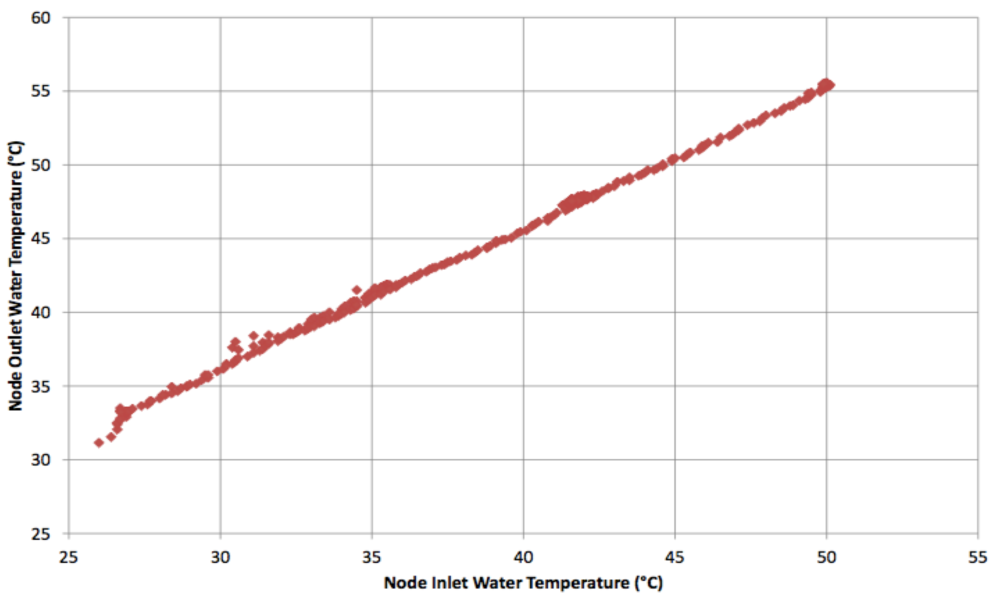


Figure 144 Water inlet vs water outlet temperature under max CPU load.

For comparison of air and direct water cooling of nodes, air-cooled compute nodes containing hardware similar to the water- cooled nodes were tested under the same load. Figure 145 shows the comparison of the air-cooled node which was cooled at 23 °C to the water-cooled node with respect to T_{CPU} and P_{Node} at different water inlet temperatures. In this experiment, the advantage of direct liquid-cooling becomes clearly visible: not only is the power consumption of the air-cooled nodes higher due to the necessity of additional chassis fans, but also the CPU temperature of the liquid cooled nodes stays below the CPU temperature of the air cooled node. The CPU temperature of the liquid cooled node only exceeds the temperature in the air-cooled node if the water temperature approaches 45 °C.

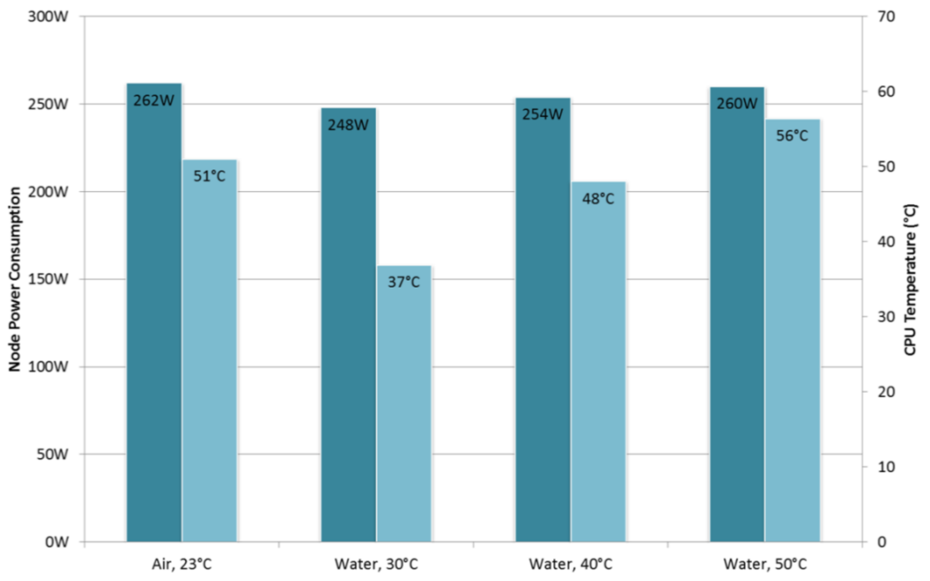


Figure 145 Comparison of node power consumption and CPU temperature of air-cooled nodes and direct liquid cooled nodes at different inlet temperatures.

Finally, Figure 146 also shows an effect that has to be considered for energy reuse: the dependence of the nodes power consumption on the water temperature. Semiconductors are known to have poor electric conductance properties at high operation temperatures, effectively resulting in an exponential increase of the power required to operate the logic circuitry. For the CooLMUC this effect is clearly visible. The increase of the power consumption is only an effect of about 0.133kW per ^{o}C, or 0.322% per ^{o}C in relation to the power consumption at 27 °C. At 50 °C the increase is 7.406%. Whether this overhead is justified by the increased ability to reuse waste heat at higher temperatures will be subject to future analysis.

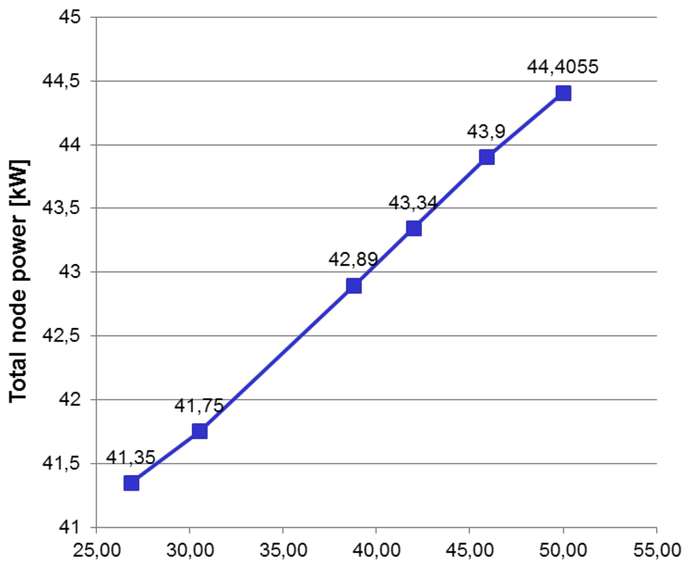
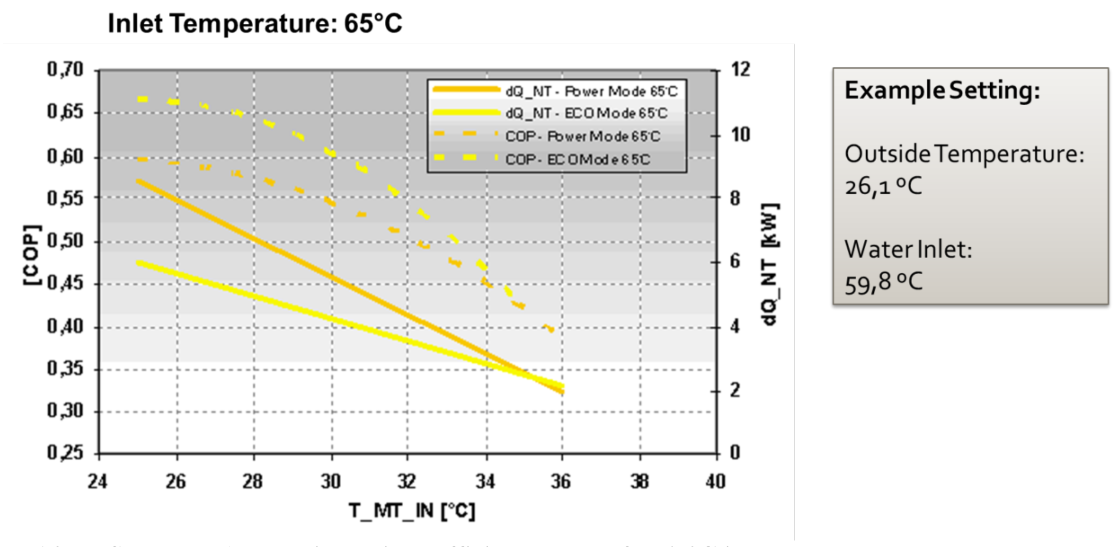


Figure 146 Node power consumption under max load in relation to water inlet temperature.

Figure 147 displays the efficiency curve for the SorTech adsorption chiller used for the CooLMUC. A CooLMUC water inlet temperature of 60 °C is required to reach an outlet temperature of 65 °C. It is used as inlet water for the adsorption chiller. The efficiency of the chiller depends on the outside air temperature. Lower outside temperature increases the efficiency. For example, with an outside air temperature from 36 °C the adsorption chiller has a Coefficient Of Performance (COP) of 0.4, meaning that only 40% of the energy removed from the hot water circuit is removed from the cooling circuit. If the air temperature decreases to 26 °C the adsorption chiller can reach a COP of 0.67 for the eco-mode and 0.60 for the power mode. Also higher water inlet temperatures will increase the COP whereas lower ones will reduce the COP. If the inlet temperature is too low during the summer, the adsorption chiller will not be able to function.



Example Setting:

Outside Temperature:
26,1 °C

Water Inlet:
59,8 °C

Figure 147 SortTech Adsorption chiller efficiency curve for 65°C inlet temperature

7 Conclusions

The objective of task 9.3 was to assess and make recommendations to the RI for joint developments with industrial partners to develop highly energy efficient HPC components and systems, power and cooling technologies. A number of prototypes were constructed or acquired and evaluated to gain the necessary insights and experiences to make the assessments and recommendations in regards to energy efficiency of systems and components and cooling technologies of interest to the PRACE RI.

Energy efficiency assessment is immature and non-trivial. What to measure and how to measure it is not commonly agreed upon and not well supported by component and platform vendors. However, with today's strong emphasis on energy efficiency of solutions there should be good grounds for collaborations with industry in developing methodologies, instrumentation and benchmarks for assessing energy efficiency and energy recovery solutions, and possibly even setting up shared testing/assessment facilities.

For nearly all benchmarks and server and cluster prototypes a direct relationship between performance and energy efficiency was observed. Therefore, an emphasis on performance optimization is very important not only for effective use of platforms but also for energy conservation. An order of magnitude difference in performance between optimized and unoptimized computational kernel codes was observed even for mature architectures and software enevironments, such x86 platforms. The highest performing platform do not, in general, minimize the energy consumption for a task. Of the protoypes, the DSP with a clock frequency of 1.25 GHz had an energy efficiency about three times higher for optimized HPL benchmarks than x86 platforms with clock frequencies in the 2.5 – 3.9 GHz range. (For individual platforms it has been observed in other investigations that for some applications, utilizing voltage and frequency control of CPUs, minimum energy consumption may occur for performance below the maxium capability of the platform.)

7.1 Node Architectures

In regards to node architectures, the prototypes covered a wide range including architectures traditionally not used for HPC applications, such as mobile CPUs (ARM) and DSPs. Prototypes also covered the emerging heterogeneous CPUs with x86 and GPU cores integrated on the same die.

The finding from the prototype efforts is that for matrix multiplication an FPGA implementation can achieve an energy efficiency that is about five times better than an x86 CPU software implementation, and that a DSP implementation could be about three times more efficient than an x86 implementation. This conclusion holds despite the fact that in our prototypes the FPGA and DSP used one generation older silicon technology than the x86 CPUs, something that however seems to be typical for the industry. The GPU based prototype did not have a significant advantage over x86 CPUs from an energy efficiency perspective. The FPGA, DSP and GPUs were all fabricated using 40 nm technology whereas the x86 CPUs were fabricated in 32 nm technology with one exception, the Intel 4-core Ivy Bridge produced in 22 nm technology. The integrated x86+GPU CPUs were first generation CPUs of this kind and not intended to serve the HPC market. They did not offer an energy efficiency advantage in our assessment, which in part could be a consequence of lack of time to carry out code optimization for the benchmarks executed though vendor libraries were used for some computational kernels. An order of magnitude improved performance due to code optimization cannot be ruled out, and was observed in the case of the DSP prototype for matrix multiplication and Linpack for which code optimizations was carried out and the Intel Ivy Bridge.

Code optimization is critical also for energy efficiency comparisons between different architectural approaches to HPC node designs. For the cases where optimizations were carried out for benchmarks achieved efficiencies did not vary widely for different approaches to node designs leading to the conclusion that for the type of architectures used for the prototypes, the nominal energy efficiencies determined by peak floating-point rate and TDP can serve as a guide. On this basis all approaches used in prototypes warrants further investigation. Without proper effort devoted to optimization a fair comparison of approaches is not possible. Though this conclusion is expected for new approaches, the prototyping effort showed that an “out-of-the-box” approach may not work well for established platforms even for simple common benchmarks.

7.2 I/O

The prototype efforts that focused on scalable I/O systems did validate the benefits from a tighter integration of I/O and file systems into MPP and cluster systems and the benefit of flash technology and the use MAID technology. This effort also did identify software issues to be addressed for scalability, such as handling of large amounts of small files, and assessed one promising approach to addressing this issue.

The concept of using flash and disks in a tiered file system is working in GPFS. Yet there is not much experience and documentation available to optimize this setup. Today the easiest approach is to use flash storage as a kind of fast local file system for the running jobs, which could be created as a parallel file system. The advantage of the flash cards - the high IOPS value - may not be achievable through the file system software stack (as was measured in GPFS), but compared with disk based storage clusters which are optimized for streamed I/O, the flash cards are performing quite well. Tests show that these cards can be used directly in Blue Gene/Q I/O nodes.

7.3 Cooling

In the evaluation of the prototype taking a holistic approach to energy efficiency the superior heat removal efficiency of liquid cooling was demonstrated. The power consumption of air-cooled and liquid cooled nodes was almost the same (262 W and 260 W respectively) for 23 ^{o}C air and 50 ^{o}C water inlet temperatures. With a 30 ^{o}C water inlet temperature the power consumption of the liquid cooled node was 248 W, or about 5% lower than the air-cooled node.

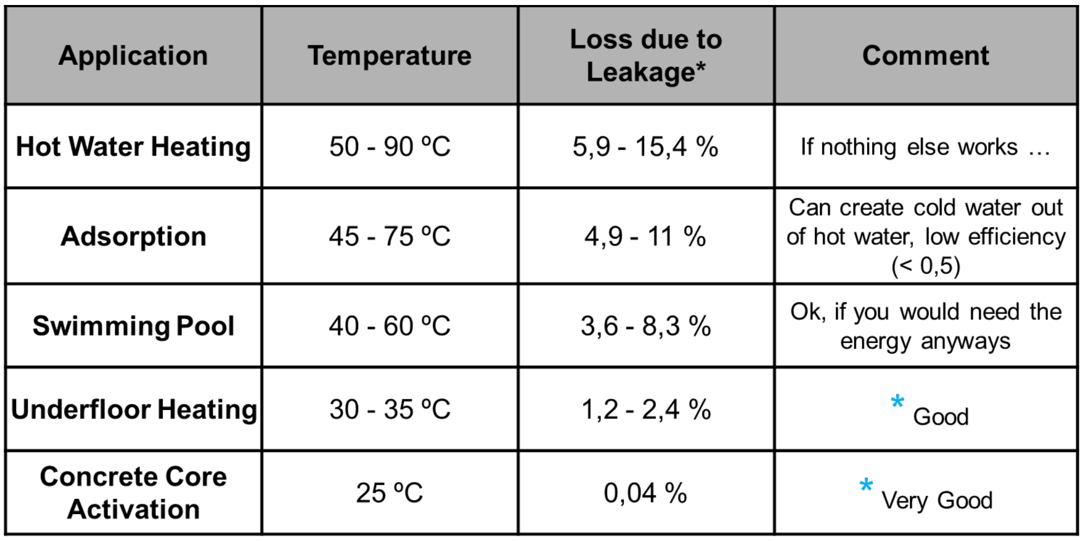
The evaluation of the prototype also validated that increased CPU temperature leads to increased power consumption. For the prototype cluster it was determined that an increase in water inlet temperature from 27 ^{o}C to 50 ^{o}C resulted in an increase in power consumption of about 7.4%.

From the tests it was also observed that the heat transfer efficiency decreased with increasing coolant temperature. At the low end of the range, 27 ^{o}C, the inlet to outlet temperature differential was about 7 ^{o}C, while at the high end, 50 ^{o}C it was 6 ^{o}C. This result confirmed well known properties of liquid cooling as a function of coolant temperature. The heat not captured by the coolant clearly is not available for energy recovery through the liquid coolant system.

Two issues in regards to direct, warm/hot water cooling should be considered carefully. First, the costs for the additional infrastructure need to be balanced against the savings that can be obtained from energy reuse. Second, at high water temperatures very good insulation of the computing equipment and the coolant pipes against heat convection is necessary to prevent the heat from escaping into the air of the computing centre (from which it would have to be removed by an air-conditioning system at additional expense).

Ideally, the waste heat from the computers could be used to generate electricity, but there does not seem to be any technology capable of doing so at reasonable efficiency given the maximum possible coolant temperatures (e.g., steam turbines need boiling water). Possible options are the use of low-temperature ammonia turbines or thermoelectrics, but these technologies are currently only options for the future.

Some of the potential applications and the corresponding requirements and issues are summarized in Figure 148.



Application	Temperature	Loss due to Leakage*	Comment
Hot Water Heating	50 - 90 °C	5,9 - 15,4 %	If nothing else works …
Adsorption	45 - 75 °C	4,9 - 11 %	Can create cold water out of hot water, low efficiency (< 0,5)
Swimming Pool	40 - 60 °C	3,6 - 8,3 %	Ok, if you would need the energy anyways
Underfloor Heating	30 - 35 °C	1,2 - 2,4 %	* Good
Concrete Core Activation	25 °C	0,04 %	* Very Good

Figure 148 Potential applications for heat reuse.

During the runtime of the system no water leakage was encountered. With current technologies this seems to be a solved problem. Because the adsorption machine cools a traditional air cooled rack CooLMUC is running with an inlet temperature of 40 °C all year around. Unfortunately during system specification the hot water cooling loop and the adsorption supply and cooling loop were not sufficiently instrumented. Therefor it is not possible to address the question of how much energy was recovered over a year. This was recently addressed but data is not available yet.

The system has an internal temperature regulation for the hot water inlet temperature. Unfortunately this set point can't be changed easily. This should be changed in future systems because the dynamic control of the inlet temperature can be crucial with regards of hot water re-use during different seasons or system load.

8 Annex

8.1 Key to benchmark result tables

In the following three sections detailed result tables show the background data behind the findings of this report. The tables follow a general layout.

The header of each column of the tables includes the physical unit of the presented quantity in square brackets (e.g. [J]). Units are usually scaled to give more convenient numbers. We attempted to use the same prefix for all the results of the same benchmark. Since some benchmark span a large range of problem sizes, some quantities are given with a specific SI prefix (m, µ) in the table overriding the column prefix.

The **Size** column contains the problem size relevant for the benchmark. In case of the *EuroBen mod2am* benchmark, this is the dimension of the square input matrices. For the *EuroBen mod2as* benchmark, the dimensions of the matrix are specified together with the fraction of non-zero elements in the **Fill** column. For the *EuroBen mod2f* benchmark, the number of complex points is given as a power of 2. For the *Linpack* benchmark, the number of unknowns is used. For *STREAM,* the length of one vector in elements (double precision numbers) is used.

The **N** column contains the number of repeated runs of the same problem submitted in the dataset. All the following columns are averages of all the runs submitted.

The **Time** column gives the average wall clock time for *one iteration* of the benchmark run. Usually the benchmark executes many iterations of the same problem.

The **Power** column contains the average power consumed by the computer and possibly network and cooling equipment during benchmark execution.

The **Energy** column gives the energy to solution for *one iteration* of the benchmark run.

The **Perf.** column contains the contains the performance obtained by the benchmark in operations over time, either floating point calculations or memory accesses.

The **Eff.** column finally contains the energy efficiency in operations per unit of energy [1J = 1 Ws].

8.2 Node Benchmark Results

8.2.1 *An NVIDIA Tegra 2 mobile SoC based HPC cluster, BSC*

BSC reported data for the Euroben mod2am benchmark in Table 36, the run times for this benchmark varied from 340 – 435 seconds, resulting in 1 – 120 000 iterations of a single multiply. Data for the Euroben mod2as benchmark listed in Table 37 is based on run-times from 166 – 207 seconds and 200 – 1 000 000 iterations. Data for the Hydro benchmark listed in Table 38 is based on 1000 time steps resulting in 1060 seconds run time. Data for the Linpack benchmark in Table 39 is based on a single iteration lasting 506 seconds. The results for the single thread STREAM operations listed in Table 44, Table 45, Table 46 and Table 47 are based on 200 – 200 000 iterations taking 61 – 450 seconds. Data for the 2 thread STREAM operations displayed in Table 40, Table 41, Table 42 and Table 43 is based on the same number of iterations (200 – 200 000) taking between 41 and 329 seconds.

All the data except for the STREAM benchmarks was collected from execution of the benchmark on a single node, the STREAM benchmark data were originally run in replicated fashion on 8 nodes and total power was reported. The data was subsequently scaled down to single node data thus representing an average of 8 different nodes.

Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
100	1	4.67	7.93	37 m	428	54.0
200	1	36.3	8.34	0.30	441	52.9
300	1	137	8.42	1.15	395	46.9
400	1	468	8.78	4.11	274	31.2
500	1	1312	8.88	11.7	190	21.5
600	1	1891	9.23	17.4	228	24.8
700	1	3654	9.11	33.3	188	20.6
800	1	4337	9.32	40.4	236	25.3
900	1	7688	9.09	69.9	190	20.9
1000	1	9447	9.16	86.5	212	23.1

Table 36 Performance and energy to solution results for the Euroben mod2am kernel. (BSC)

Matrix Size	Fill [%]	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
500×500	10	1	180	6.94	1.25	278	40.1
1000×1000	10	1	1279	4.27	5.46	156	36.6
2000×2000	10	1	8287	7.35	60.9	96.5	13.1
3000×3000	10	1	19122	8.66	166	94.1	10.9
4000×4000	10	1	34748	8.67	301	92.1	10.6
5000×5000	10	1	55520	8.67	481	90.1	10.4
10000×10000	10	1	240200	7.53	1808	83.3	11.1
20000×20000	10	1	992250	7.71	7646	80.6	10.5

Table 37 Performance and energy to solution results for the Euroben mod2as kernel. (BSC)

Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
500×500	1	1.06	8.05	8.53	236	29.3

Table 38 Performance and energy to solution results for the Hydro application benchmark. (BSC)

Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
9920	1	575	7.99	4599	1.1	142

Table 39 Performance and energy to solution results for the Linpack benchmark. (BSC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
43690	1	213	9.14	1.94	3287	359
174760	1	2102	9.75	20.5	1330	136
699050	1	7727	9.74	75.3	1448	149
2796200	1	32239	9.90	319	1388	140
5592400	1	65882	9.87	650	1358	138
11184810	1	122615	9.56	1172	1460	153
33554432	1	360950	9.57	3453	1487	155

Table 40 Performance and energy to solution for the STREAM copy benchmark using 2 threads. (BSC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
43690	1	289	9.56	2.76	2422	253
174760	1	2179	9.96	21.7	1283	129
699050	1	8319	10.00	83.2	1344	134
2796200	1	31561	10.02	316	1418	141
5592400	1	67719	10.01	678	1321	132
11184810	1	131487	10.00	1315	1361	136
33554432	1	393750	9.85	3878	1363	138

Table 41 Performance and energy to solution for the STREAM scale benchmark using 2 threads. (BSC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
43690	1	525	9.59	5.04	1996	208
174760	1	4165	9.98	41.6	1007	101
699050	1	16451	10.03	165	1020	102
2796200	1	63867	10.04	641	1051	105
5592400	1	132251	10.03	1326	1015	101
11184810	1	263641	10.04	2647	1018	101
33554432	1	864100	9.88	8537	932	94.3

Table 42 Performance and energy to solution for the STREAM sum benchmark using 2 threads. (BCS)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
43690	1	614	9.57	5.87	1708	178
174760	1	4081	9.94	40.6	1028	103
699050	1	17136	10.02	172	979	97.7
2796200	1	65481	10.00	655	1025	102
5592400	1	130557	9.99	1304	1028	103
11184810	1	280538	10.00	2806	957	95.7
33554432	1	773450	9.89	7648	1041	105

Table 43 Performance and energy to solution for the STREAM triad benchmark using 2 threads. (BSC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
43690	1	363	9.23	3.35	1924	208
174760	1	2592	9.47	24.5	1079	114
699050	1	10156	9.50	96.4	1101	116
2796200	1	38517	9.62	371	1162	121
5592400	1	76931	9.57	736	1163	122
11184810	1	171436	9.35	1603	1044	112
33554432	1	542350	9.35	5069	990	106

Table 44 Performance and energy to solution for the STREAM copy benchmark using 1 thread. (BSC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
43690	1	301	9.44	2.85	2319	246
174760	1	2656	9.60	25.5	1053	110
699050	1	10098	9.67	97.6	1108	115
2796200	1	52807	9.72	513	847	87.2
5592400	1	83576	9.67	808	1071	111
11184810	1	159436	9.60	1531	1122	117
33554432	1	542200	9.49	5148	990	104

Table 45 Performance and energy to solution for the STREAM scale benchmark using 1 thread. (BCS)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
43690	1	689	9.47	6.52	1523	161
174760	1	6879	9.63	66.2	610	63.3
699050	1	20938	9.69	203	801	82.7
2796200	1	68716	9.73	668	977	100
5592400	1	218749	9.69	2119	614	63.3
11184810	1	443923	9.64	4278	605	62.7
33554432	1	1281250	9.52	12202	629	66.0

Table 46 Performance and energy to solution for the STREAM sum benchmark using 1 thread. (BSC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
43690	1	725	9.45	6.85	1446	153
174760	1	5930	9.61	57.0	707	73.6
699050	1	22905	9.69	222	732	75.6
2796200	1	90153	9.72	876	744	76.6
5592400	1	166567	9.68	1613	806	83.2
11184810	1	368590	9.61	3543	728	75.8
33554432	1	1342150	9.53	12787	600	63.0

Table 47 Performance and energy to solution for the STREAM triad benchmark using 1 thread. (BSC)

8.2.2 GPU-GPU communication over PCIe and IB, CaSToRC

CaSToRC provided data for single node runs using 2 GPUs except for the mod2f data, which is based on 1 GPU only. The Euroben mod2f results shown in Table 48 are based on 100 000 – 110 000 iterations of a single FFT that took between 122 to 135 seconds per benchmark execution. 16 replicated runs were in the dataset as base for the statistics.

The results from the Hydro benchmark in Table 49 are based on runs with 23 – 106 time steps and run times from 328 to 387 seconds.

The Linpack results shown in Table 50 were averaged from two different decompositions of the HPL benchmark that were executed in a single run. The total run time was 24 seconds. The low power readings are possibly an artifact of the short run-time, see also section 8.4.2.

The STREAM results in Table 51, Table 52, Table 53 and Table 54 were also calculated from single benchmark runs for all 4 operations. Here no number of iterations was given but bandwidth numbers for the individual operations were reported. From this data and the average power consumption during the whole run execution times for single runs and energy efficiency were calculated. The total run time for the benchmark runs was between 161 and 436 seconds. Statistics are based on 12 repeated experiments.

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
2^{18}	16	967 ± 2.95%	296 ± 1.76%	286 ± 3.43%	142677 ± 0.33%	95.7 ± 3.43%
2^{21}	16	1999 ± 1.43%	348 ± 1.63%	696 ± 2.17%	220501 ± 0.10%	389 ± 2.17%

Table 48 Performance and energy to solution for the Euroben mod2f benchmark. (CaSToRC)

Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
12288×6144	1	4.41	357	1574	17136	48.0
12288×12288	1	6.19	386	2391	24399	63.1
13440×13440	1	16.8	327	5510	10735	32.8

Table 49 Performance and energy to solution for the Hydro application benchmark. (CaSToRC)

Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
12288	1	12.0	150	1797	103	689

Table 50 Performance and energy to solution for the Linpack benchmark. (CaSToRC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
128	12	5.37 ± 5.96%	300 ± 5.88%	1.61 ± 8.37%	382 ± 5.96%	1.3 ± 8.37%
4096	12	5.68 ± 7.91%	301 ± 5.49%	1.71 ± 9.63%	11529 ± 7.91%	38.3 ± 9.63%
131072	12	23.0 ± 0.84%	314 ± 6.16%	7.23 ± 6.21%	91003 ± 0.84%	290 ± 6.21%
4194304	12	603 ± 0.03%	341 ± 4.16%	206 ± 4.16%	111325 ± 0.03%	326 ± 4.16%
134217728	12	24257 ± 0.03%	328 ± 5.87%	7967 ± 5.87%	88530 ± 0.03%	270 ± 5.87%

Table 51 Performance and energy to solution for the STREAM copy benchmark. (CaSToRC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
128	12	5.47 ± 6.32%	300 ± 5.88%	1.64 ± 8.63%	374 ± 6.32%	1.2 ± 8.63%
4096	12	5.82 ± 7.00%	301 ± 5.49%	1.75 ± 8.90%	11267 ± 7.00%	37.4 ± 8.90%
131072	12	23.1 ± 0.73%	314 ± 6.16%	7.25 ± 6.20%	90762 ± 0.73%	289 ± 6.20%
4194304	12	604 ± 0.03%	341 ± 4.16%	206 ± 4.16%	111167 ± 0.03%	326 ± 4.16%
134217728	12	24176 ± 0.03%	328 ± 5.87%	7940 ± 5.87%	88829 ± 0.03%	270 ± 5.87%

Table 52 Performance and energy to solution for the STREAM scale benchmark. (CaSToRC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
128	12	5.49 ± 5.84%	300 ± 5.88%	1.65 ± 8.29%	560 ± 5.84%	1.9 ± 8.29%
4096	12	5.96 ± 7.05%	301 ± 5.49%	1.79 ± 8.94%	16497 ± 7.05%	54.8 ± 8.94%
131072	12	33.4 ± 0.37%	314 ± 6.16%	10.5 ± 6.17%	94324 ± 0.37%	301 ± 6.17%
4194304	12	914 ± 0.03%	341 ± 4.16%	312 ± 4.16%	110103 ± 0.03%	323 ± 4.16%
134217728	12	36079 ± 0.02%	328 ± 5.87%	11850 ± 5.87%	89283 ± 0.02%	272 ± 5.87%

Table 53 Performance and energy to solution for the STREAM sum benchmark. (CaSToRC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
128	12	5.72 ± 5.49%	300 ± 5.88%	1.71 ± 8.04%	537 ± 5.49%	1.8 ± 8.04%
4096	12	6.19 ± 6.67%	301 ± 5.49%	1.86 ± 8.64%	15890 ± 6.67%	52.8 ± 8.64%
131072	12	33.6 ± 0.70%	314 ± 6.16%	10.6 ± 6.20%	93572 ± 0.70%	298 ± 6.20%
4194304	12	915 ± 0.03%	341 ± 4.16%	312 ± 4.16%	110072 ± 0.03%	323 ± 4.16%
134217728	12	36080 ± 0.02%	328 ± 5.87%	11850 ± 5.87%	89281 ± 0.02%	272 ± 5.87%

Table 54 Performance and energy to solution for the STREAM triad benchmark. (CaSToRC)

8.2.3 *DSP based node for HPC, SNIC*

Data for the DSP node was gathered during two different measurement runs, the older run was generated using the simplified instrumentation detailed in Figure 50 that delivered about 3 samples per second. The Euroben mod2am results presented in Table 55 are based on runs containing between 140 to 130 000 000 iterations to achieve a run time of 1000 seconds per run. A similar strategy was used for the mod2as runs shown in Table 56 resulting in 120 – 120 million iterations for the 1000 second run time. For the mod2f results in Table 57 4 800 – 4 000 000 transformations were required. The Hydro benchmark shown in Table 58 took between 5 and 5 000 time steps and had run times between 781 and 1839 seconds. Each operation of the STREAM benchmark was run on its own and repeated between 7 384 and 32 billion times to fix the run time at 1000 seconds the results are presented in Table 60, Table 61, Table 62 and Table 63.

The Linpack benchmark, Table 59, was carried out with the newer faster instrumentation detailed in Figure 48 in place and used 11 – 20 runs to gather statistical errors. The shorter runs have rather large statistical variation due to the run time being in order of the time resolution of the current event tracking via serial line.

Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
10	1	8.31 µ	6.17	51 µ	249	39.0
20	1	48.1 µ	6.23	0.3 m	337	53.4
50	1	0.78	6.19	5 m	320	51.5
100	1	6.92	6.17	43 m	289	46.8
200	1	55.2	6.17	0.34	290	47.0
500	1	900	6.48	5.83	278	42.9
1000	1	7270	6.48	47.1	275	42.4

Table 55 Performance and energy to solution for the Euroben mod2am benchmark. (SNIC)

Matrix Size	Fill [%]	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
100×100	3.5	1	8.83	5.99	53 m	79.2	13.2
200×200	3.75	1	22.9	5.98	0.14	131	21.9
400×400	4.38	1	83.6	5.98	0.50	167	28.0
600×600	4.44	1	176	5.99	1.05	182	30.4
1000×1000	5	1	518	6.35	3.29	193	30.4
2000×2000	7.5	2	2964 ± 0.00%	6.49 ± 0.04%	19.2 ± 0.04%	202 ± 0.00%	31.2 ± 0.04%
10000×10000	4	2	58327 ± 0.00%	6.33 ± 0.12%	369 ± 0.12%	137 ± 0.00%	21.7 ± 0.12%
20000×20000	4	2	249505 ± 0.00%	6.30 ± 0.14%	1572 ± 0.14%	128 ± 0.00%	20.4 ± 0.14%
30000×30000	3.56	2	512139 ± 0.00%	6.30 ± 0.22%	3225 ± 0.22%	125 ± 0.00%	19.8 ± 0.22%

Table 56 Performance and energy to solution for the Euroben mod2as benchmark. (SNIC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
2^{8}	1	258	7.00	1.81	44.9	6.3
2^{9}	1	292	7.07	2.07	103	14.2
2^{10}	1	348	7.14	2.49	173	23.3
2^{11}	1	457	7.27	3.33	330	42.6
2^{12}	1	658	7.42	4.88	470	57.5
2^{13}	1	1158	7.52	8.71	643	76.4
2^{14}	1	2232	7.50	16.7	664	78.8
2^{15}	1	4512	7.54	34.0	818	89.9
2^{16}	1	9851	7.44	73.3	730	82.7
2^{17}	1	29156	7.90	230	533	59.9
2^{18}	1	59182	7.90	468	521	58.6
2^{19}	1	122092	7.89	963	573	63.8
2^{20}	1	237232	7.96	1888	589	64.7

Table 57 Performance and energy to solution for the Euroben mod2f benchmark. (SNIC)

Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
100×100	1	0.26	6.15	1.61	38.2	6.2
200×200	1	0.74	6.21	4.57	54.4	8.8
500×500	1	3.91	6.20	24.2	64.0	10.3
1000×1000	1	16.2	6.22	100.8	61.7	9.9
2000×2000	1	65.5	6.22	407	61.1	9.8
3500×3500	1	200	6.21	1242	61.3	9.9

Table 58 Performance and energy to solution for the Hydro benchmark. (SNIC)

Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
127	11	2 m ± 0.00%	4.77 ± 1.06%	10 m ± 1.06%	0.7 ± 0.00%	143 ± 1.06%
255	20	4 m ± 38.66%	5.61 ± 1.59%	21 m ± 38.69%	3.0 ± 38.66%	541 ± 38.69%
511	20	13 m ± 7.91%	7.05 ± 0.99%	93 m ± 7.98%	6.8 ± 7.91%	961 ± 7.98%
1023	20	55 m ± 0.83%	8.38 ± 0.36%	0.46 ± 0.91%	13.0 ± 0.83%	1548 ± 0.91%
2047	20	0.30 ± 0.32%	9.46 ± 0.22%	2.81 ± 0.39%	19.2 ± 0.32%	2035 ± 0.39%
4095	20	1.77 ± 0.05%	10.7 ± 0.14%	18.9 ± 0.15%	25.9 ± 0.05%	2421 ± 0.15%
8063	20	11.3 ± 0.01%	11.7 ± 0.09%	132 ± 0.09%	30.9 ± 0.01%	2644 ± 0.09%

Table 59 Performance and energy to solution for the Linpack benchmark. (SNIC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
1024	1	0.16	8.56	1 m	103052	12037
2048	1	0.29	8.74	3 m	114183	13067
4096	1	0.54	8.84	5 m	120702	13649
8192	1	1.05	8.92	9 m	124248	13928
16384	1	2.08	8.95	19 m	126102	14089
32768	1	4.13	8.96	37 m	127049	14173
65536	1	22.5	7.12	0.16	46502	6534
131072	1	45.1	7.12	0.32	46525	6538
262144	1	1407	7.57	10.6	2982	394
524288	1	2791	7.58	21.1	3005	397
1048576	1	5561	7.58	42.1	3017	398
2097152	1	11106	7.58	84.2	3021	399
4194304	1	22199	7.59	168	3023	398
8388608	1	44327	7.59	336	3028	399
16777216	1	88660	7.59	673	3028	399

Table 60 Performance and energy to solution for the STREAM copy benchmark. (SNIC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
1024	1	0.16	8.89	1 m	99910	11237
2048	1	0.29	9.17	3 m	112228	12238
4096	1	0.55	9.34	5 m	119601	12808
8192	1	1.06	9.43	10 m	123662	13118
16384	1	2.08	9.47	20 m	125799	13280
32768	1	4.13	9.50	39 m	126895	13364
65536	1	24.1	7.43	0.18	43533	5859
131072	1	48.1	7.42	0.36	43555	5868
262144	1	1463	7.56	11.1	2867	379
524288	1	2903	7.56	21.9	2889	382
1048576	1	5791	7.56	43.8	2897	383
2097152	1	11559	7.56	87.4	2903	384
4194304	1	23043	7.56	174	2912	385
8388608	1	46124	7.56	349	2910	385
16777216	1	92244	7.56	698	2910	385

Table 61 Performance and energy to solution for the STREAM scale benchmark. (SNIC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
1024	1	0.27	8.01	2 m	92398	11533
2048	1	0.51	8.04	4 m	97146	12079
4096	1	0.99	8.06	8 m	99708	12370
8192	1	1.95	8.07	16 m	101040	12518
16384	1	3.87	8.07	31 m	101719	12599
32768	1	16.9	7.17	0.12	46500	6488
65536	1	33.8	7.17	0.24	46524	6491
131072	1	901	7.67	6.91	3492	455
262144	1	2182	7.34	16.0	2883	393
524288	1	4294	7.36	31.6	2930	398
1048576	1	8511	7.37	62.7	2957	401
2097152	1	16965	7.37	125	2967	402
4194304	1	33880	7.38	250	2971	403
8388608	1	67699	7.38	499	2974	403
16777216	1	135178	7.38	997	2979	404

Table 62 Performance and energy to solution for the STREAM sum benchmark. (SNIC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
1024	1	0.30	8.00	2 m	80848	10105
2048	1	0.58	8.02	5 m	85340	10638
4096	1	1.12	8.06	9 m	87778	10897
8192	1	2.21	8.06	18 m	89051	11042
16384	1	4.38	8.07	35 m	89701	11116
32768	1	23.1	6.86	0.16	34117	4976
65536	1	46.1	6.86	0.32	34126	4978
131072	1	883	7.70	6.80	3563	463
262144	1	1739	7.58	13.2	3618	477
524288	1	3466	7.58	26.3	3630	479
1048576	1	6919	7.58	52.5	3637	480
2097152	1	13821	7.59	105	3642	480
4194304	1	27643	7.59	210	3642	480
8388608	1	55284	7.59	419	3642	480
16777216	1	110559	7.58	838	3642	480

Table 63 Performance and energy to solution for the STREAM triad benchmark. (SNIC)

8.2.4 *Holistic approach to energy efficiency, LRZ*

LRZ delivered single node data for the mod2am benchmark shown in Table 64, unfortunately power data was only available for matrix sizes between 200^2 and 576^2. Data for mod2as is shown in Table 65 also here only matrices with 57 671 to 1 000 000 non-zero elements have power measurement data. For the mod2f benchmark a full data set is available in Table 66. The data for the Hydro benchmark shown in Table 67 is also complete. The Linpack benchmark data in Table 68 contains also power data for the whole sweep.

For the STREAM operations shown in Table 69, Table 70, Table 71 and Table 72 a single average power value was reported along with bandwidth results for all for operations. Per iteration run time and energy to solution and energy efficiency were calculated from these values.

In general LRZ power instrumentation required run times of about 2400 seconds to collect enough samples for accuracy better than 5% due to the low sample rates of some of the cooling infrastructure instrumentation.

Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
10	1	50.6 µ	-	-	39.6	-
20	1	150 µ	-	-	106	-
50	1	99.3 µ	-	-	2517	-
100	1	0.31	-	-	6505	-
192	1	1.61	-	-	8806	-
200	1	1.81	243	0.44	9046	37.3
500	1	25.4	242	6.15	9806	40.5
512	1	27.5	246	6.77	9685	39.4
576	1	40.6	247	10.02	9554	38.7
1000	1	621	-	-	3219	-
1024	1	680	-	-	3160	-
2000	1	7428	-	-	2154	-
2048	1	8145	-	-	2109	-
5000	1	132700	-	-	1883	-
10240	1	1181000	-	-	1818	-
12096	1	1977000	-	-	1790	-
20000	1	7696000	-	-	2079	-

Table 64 Performance and energy to solution for the Euroben mod2am benchmark. (LRZ)

Matrix Size	Fill [%]	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
100×100	3.5	1	23.9	-	-	29.2	-
200×200	3.75	1	24.4	-	-	123	-
256×256	5	1	25.3	-	-	259	-
400×400	4.38	1	27.7	-	-	505	-
500×500	5	1	29.9	-	-	701	-
512×512	4	1	31.3	-	-	799	-
960×960	4.5	1	48.9	-	-	1698	-
1000×1000	5	1	54.2	-	-	1846	-
1024×1024	5.5	1	59.0	236	13.9	1947	8.3
2000×2000	7.5	1	199	237	47.2	3011	12.7
4096×4096	3.5	1	393	237	93.3	2985	12.6
4992×4992	4	1	839	236	198	2375	10.1
5000×5000	4	1	676	238	161	2958	12.4
9984×9984	4.5	1	15400	-	-	584	-
10000×10000	5.0	1	17500	-	-	571	-
10240×10240	5.72	1	21000	-	-	571	-
20000×20000	5	1	70300	-	-	569	-
40000×40000	5	1	285000	-	-	561	-

Table 65 Performance and energy to solution for the Euroben mod2as benchmark. (LRZ)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
2^{15}	1	248	277	68.7	12359	44.5
2^{16}	1	479	280	134	12664	45.2
2^{17}	1	1038	286	297	13295	46.5
2^{18}	1	1981	287	568	13829	48.2
2^{19}	1	4471	293	1310	13748	46.9
2^{20}	1	10443	297	3103	11698	39.4
2^{21}	1	21570	363	7823	12567	34.6
2^{22}	1	47896	291	13914	11253	38.7
2^{23}	1	102170	289	29556	11597	40.1
2^{24}	1	205310	290	59613	11481	39.5
2^{25}	1	469420	293	137502	10954	37.4
2^{26}	1	1004500	289	290617	10189	35.2

Table 66 Performance and energy to solution for the Euroben mod2f benchmark. (LRZ)

Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
250×250	1	4 m	258	0.97	16640	64.6
500×500	1	15 m	251	3.75	16692	66.6
1000×1000	1	60 m	262	15.6	16795	64.1
1920×1920	1	0.25	257	63.3	14961	58.3
3840×3840	1	1.02	255	260	14456	56.7
5760×5760	1	2.38	256	608	13964	54.6
7680×7680	1	4.46	256	1143	13227	51.6
9600×9600	1	7.74	267	2071	11903	44.5
11520×11520	1	12.3	270	3312	10811	40.1
13440×13440	1	18.5	265	4915	9753	36.8
15360×15360	1	26.9	268	7229	8761	32.6
17280×17280	1	35.8	270	9662	8344	30.9
19200×19200	1	51.6	270	13944	7146	26.4

Table 67 Performance and energy to solution for the Hydro benchmark. (LRZ)

Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
336	1	743 µ	221	0.16	2.6	12.0
840	1	3 m	261	0.77	12.5	48.0
2520	1	0.33	313	101.9	45.6	146
5040	1	2.05	272	558	68.6	252
7224	1	3.19	331	1056	79.1	239
10248	1	8.44	335	2828	86.3	257
13608	1	19.0	343	6530	93.2	272
19320	1	50.4	323	16305	99.7	308
27384	1	134	332	44532	104	314
38640	1	355	331	117409	108	326

Table 68 Performance and energy to solution for the Linpack benchmark. (LRZ)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
1365	1	5.96	231	1.38	3664	15.9
2731	1	5.96	234	1.39	7331	31.4
5461	1	5.96	233	1.39	14659	63.0
10923	1	5.96	227	1.35	29321	129
21845	1	6.91	227	1.57	50551	223
43691	1	7.87	238	1.87	88850	373
87381	1	12.9	222	2.85	108593	490
174763	1	21.0	242	5.08	133274	550
349525	1	39.8	234	9.30	140456	601
699051	1	101.8	214	21.8	109865	514
1398101	1	286	233	66.6	78253	336
2796203	1	1132	221	250	39522	179
5592405	1	2039	229	467	43884	192
11184811	1	4506	222	1002	39714	179
22369621	1	7382	226	1666	48485	215
44739243	1	14666	226	3321	48808	216
89478485	1	28729	231	6632	49833	216
178956971	1	57232	226	12922	50030	222
357913941	1	115393	217	25065	49627	228

Table 69 Performance and energy to solution for the STREAM copy benchmark. (LRZ)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
1365	1	5.96	231	1.38	3664	15.9
2731	1	5.96	234	1.39	7331	31.4
5461	1	5.96	233	1.39	14659	63.0
10923	1	5.96	227	1.35	29321	129
21845	1	6.91	227	1.57	50551	223
43691	1	7.87	238	1.87	88850	373
87381	1	12.9	222	2.85	108593	490
174763	1	17.9	242	4.33	156375	646
349525	1	37.0	234	8.64	151330	648
699051	1	99.9	214	21.4	111963	524
1398101	1	289	233	67.4	77350	332
2796203	1	964	221	213	46414	210
5592405	1	2270	229	520	39418	172
11184811	1	6033	222	1341	29663	133
22369621	1	9745	226	2200	36728	163
44739243	1	20181	226	4570	35470	157
89478485	1	39721	231	9169	36043	156
178956971	1	79622	226	17978	35961	159
357913941	1	161506	217	35081	35458	163

Table 70 Performance and energy to solution for the STREAM scale benchmark. (LRZ)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
1365	1	5.96	231	1.38	5496	23.8
2731	1	5.96	234	1.39	10996	47.0
5461	1	5.96	233	1.39	21989	94.4
10923	1	5.96	227	1.35	43982	194
21845	1	6.91	227	1.57	75827	334
43691	1	8.82	238	2.10	118867	499
87381	1	16.0	222	3.54	131284	592
174763	1	23.8	242	5.78	175922	726
349525	1	49.8	234	11.6	168346	720
699051	1	149	214	31.8	112770	527
1398101	1	472	233	110	71115	305
2796203	1	1826	221	404	36756	166
5592405	1	3490	229	799	38458	168
11184811	1	9192	222	2044	29203	131
22369621	1	13628	226	3076	39395	175
44739243	1	27954	226	6330	38411	170
89478485	1	54365	231	12549	39501	171
178956971	1	108501	226	24498	39585	175
357913941	1	220050	217	47798	39036	180

Table 71 Performance and energy to solution for the STREAM sum benchmark. (LRZ)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
1365	1	5.96	231	1.38	5496	23.8
2731	1	5.96	234	1.39	10996	47.0
5461	1	5.96	233	1.39	21989	94.4
10923	1	5.96	227	1.35	43982	194
21845	1	6.91	227	1.57	75827	334
43691	1	8.82	238	2.10	118867	499
87381	1	16.0	222	3.54	131284	592
174763	1	23.8	242	5.78	175922	726
349525	1	54.8	234	12.8	152975	655
699051	1	151	214	32.3	111167	520
1398101	1	526	233	123	63798	274
2796203	1	1705	221	377	39362	178
5592405	1	3405	229	780	39417	172
11184811	1	9057	222	2014	29638	133
22369621	1	13569	226	3063	39566	175
44739243	1	27905	226	6318	38478	170
89478485	1	54354	231	12547	39509	171
178956971	1	108493	226	24496	39587	175
357913941	1	220200	217	47831	39010	180

Table 72 Performance and energy to solution for the STREAM triad benchmark. (LRZ)

8.2.5 *On die integrated CPU and GPU, PSNC*

PSNC has submitted data using two different sets of instrumentation. Unfortunately the first type of instrumentation reporting AC side measurements proved to give unreliable results, so all the power data from this submission was considered erroneous.

In this submission there were data for STREAM using a single thread on the AMD E-350 APU in Table 73, Table 74, Table 75 and Table 76, as well as using two threads in Table 80, Table 81, Table 82 and Table 83. Euroben mod2am data is presented in Table 77, mod2as in Table 78 and mod2f in Table 79.

Performance only data for the Euroben mod2as benchmark on the AMD A8-3870 APU is shown in Table 85 and for the Euroben mod2f benchmark in Table 86. Using the newer DC side instrumentation, PSNC was able to provide reliable power measurement data for the Euroben mod2am benchmark in Table 84, for the Hydro benchmark in Table 87 and for the Linpack benchmark in Table 88.

PSNC also provided complete Euroben mod2am benchmark results for the A10-5800k APU in Table 89 along with data for the Linpack benchmark in Table 92. For the A10-5800k APU data for the Hydro benchmark executing on the CPU cores is presented in Table 90 to compare with the OpenCL version of Hydro executing on the GPU cores in Table 91.

Performance data for the Intel i7-2600 CPU was submitted for the Euroben mod2am benchmark using the Intel MKL library in Table 93 to compare with the results using ACML in Table 94. Furthermore performance data for the Euroben mod2as benchmark is presented in Table 95 and for mod2f in Table 96.

Finally PSNC also provided complete a dataset including power for the Intel i7-3770 CPU for the Euroben mod2am benchmark in Table 97. Data for Hydro on the same CPU is shown in Table 98 and Linpack results are presented in Table 99.

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
100	1	0.57	-	-	2826	-
200	1	0.89	-	-	3579	-
500	1	1.31	-	-	6101	-
1000	1	2.38	-	-	6711	-
10000	1	32.4	-	-	4934	-
100000	1	916	-	-	1748	-
1000000	1	8789	-	-	1820	-
10000000	1	82031	-	-	1950	-

Table 73 Performance for the STREAM copy benchmark on the AMD E-350 APU for 1 thread. (PSNC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
100	1	0.69	-	-	2334	-
200	1	1.07	-	-	2983	-
500	1	1.91	-	-	4194	-
1000	1	3.58	-	-	4474	-
10000	1	40.1	-	-	3995	-
100000	1	916	-	-	1748	-
1000000	1	8301	-	-	1928	-
10000000	1	82031	-	-	1950	-

Table 74 Performance for the STREAM scale benchmark on the AMD E-350 APU for 1 thread. (PSNC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
100	1	0.69	-	-	3501	-
200	1	1.07	-	-	4474	-
500	1	1.91	-	-	6291	-
1000	1	3.46	-	-	6942	-
10000	1	43.9	-	-	5471	-
100000	1	1221	-	-	1966	-
1000000	1	11230	-	-	2137	-
10000000	1	109375	-	-	2194	-

Table 75 Performance for the STREAM sum benchmark on the AMD E-350 APU for 1 thread. (PSNC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
100	1	0.66	-	-	3660	-
200	1	1.01	-	-	4737	-
500	1	1.85	-	-	6494	-
1000	1	3.22	-	-	7457	-
10000	1	45.8	-	-	5243	-
100000	1	1160	-	-	2070	-
1000000	1	11719	-	-	2048	-
10000000	1	109375	-	-	2194	-

Table 76 Performance for the STREAM triad benchmark on the AMD E-350 APU for 1 thread. (PSNC)

Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
10	1	6 m	-	-	323	-
20	1	19 m	-	-	839	-
500	1	125	-	-	2000	-
1000	1	938	-	-	2133	-

Table 77 Performance for the Euroben mod2am benchmark on the AMD E-350 APU. (PSNC)

Matrix Size	Fill [%]	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
100×100	3.5	1	2.62	-	-	267	-
200×200	3.75	1	9.54	-	-	315	-
600×600	4.44	1	91.6	-	-	350	-

Table 78 Performance for the Euroben mod2as benchmark on the AMD E-350 APU. (PSNC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
2^8	1	28.6	-	-	400	-
2^9	1	80.1	-	-	366	-
2^{10}	1	130	-	-	446	-
2^{11}	1	458	-	-	310	-
2^{12}	1	702	-	-	400	-
2^{13}	1	2075	-	-	321	-
2^{14}	1	4639	-	-	284	-
2^{15}	1	16602	-	-	184	-
2^{16}	1	31250	-	-	194	-
2^{17}	1	78125	-	-	177	-
2^{18}	1	140625	-	-	195	-
2^{19}	1	312500	-	-	197	-
2^{20}	1	625000	-	-	195	-

Table 79 Performance for the Euroben mod2f benchmark on the AMD E-350 APU. (PSNC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
100	1	0.60	-	-	2684	-
200	1	0.83	-	-	3835	-
500	1	1.37	-	-	5836	-
1000	1	2.38	-	-	6711	-
10000	1	32.4	-	-	4934	-
100000	1	916	-	-	1748	-
1000000	1	8301	-	-	1928	-
10000000	1	82031	-	-	1950	-

Table 80 Performance for the STREAM copy benchmark on the AMD E-350 APU using 2 threads. (PSNC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
100	1	0.69	-	-	2334	-
200	1	1.07	-	-	2983	-
500	1	1.91	-	-	4194	-
1000	1	3.58	-	-	4474	-
10000	1	38.1	-	-	4194	-
100000	1	916	-	-	1748	-
1000000	1	8301	-	-	1928	-
10000000	1	85938	-	-	1862	-

Table 81 Performance for the STREAM scale benchmark on the AMD E-350 APU using 2 threads. (PSNC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
100	1	0.69	-	-	3501	-
200	1	1.07	-	-	4474	-
500	1	1.91	-	-	6291	-
1000	1	3.46	-	-	6942	-
10000	1	43.9	-	-	5471	-
100000	1	1221	-	-	1966	-
1000000	1	11230	-	-	2137	-
10000000	1	109375	-	-	2194	-

Table 82 Performance for the STREAM sum benchmark on the AMD E-350 APU using 2 threads. (PSNC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MB/s]	Eff. [MB/J]
100	1	0.66	-	-	3660	-
200	1	1.07	-	-	4474	-
500	1	1.79	-	-	6711	-
1000	1	3.34	-	-	7190	-
10000	1	45.8	-	-	5243	-
100000	1	1160	-	-	2070	-
1000000	1	11719	-	-	2048	-
10000000	1	109375	-	-	2194	-

Table 83 Performance for the STREAM triad benchmark on the AMD E-350 APU using 2 threads. (PSNC)

Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
10	1	1.67 m	119	199 µ	1198	10.1
20	1	5.96 m	124	737 µ	2684	21.7
50	1	53.4 m	127	6.77 m	4681	36.9
100	1	0.21	201	43.0 m	9362	46.5
500	1	15.1	243	3.68	16516	67.9
1000	1	101.6	207	21.0	19692	95.1

Table 84 Performance and energy to solution for the Euroben mod2am benchmark on the AMD A8-3870 APU. (PSNC)

Size	Fill [%]	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
100×100	3.5	1	1.19	-	-	587	-
200×200	3.75	1	4.77	-	-	629	-
400×400	4.38	1	21.0	-	-	667	-
600×600	4.44	1	45.8	-	-	699	-
1000×1000	5	1	145	-	-	690	-
2000×2000	7.5	1	1038	-	-	578	-

Table 85 Performance for the Euroben mod2as benchmark on the AMD A8-3870 APU. (PSNC)

Size	N	Time [µs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
2^8	1	10.01	-	-	1142	-
2^9	1	32.4	-	-	905	-
2^{10}	1	43.9	-	-	1320	-
2^{11}	1	145	-	-	979	-
2^{12}	1	259	-	-	1082	-
2^{13}	1	1221	-	-	545	-
2^{14}	1	2197	-	-	600	-
2^{15}	1	6836	-	-	447	-
2^{16}	1	16602	-	-	365	-
2^{17}	1	39062	-	-	353	-
2^{18}	1	74219	-	-	369	-
2^{19}	1	156250	-	-	393	-
2^{20}	1	312500	-	-	391	-

Table 86 Performance of the Euroben mod2f benchmark on the AMD A8-3870 APU. (PSNC)

Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
500×500	1	0.35	75.7	26.5	714	9.4
1000×1000	1	1.02	109	111	980	9.0
1500×1500	1	2.42	125	303	930	7.4
2000×2000	1	3.87	134	518	1034	7.7
2500×2500	1	6.57	135	884	951	7.1

Table 87 Performance and energy to solution of the Hydro benchmark on the AMD A8-3870 APU. (PSNC)

Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
18000	1	157	131	20507	24.8	190

Table 88 Performance and energy to solution of the Linpack benchmark on the AMD A8-3870 APU. (PSNC)

Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
10	1	1.37 µ	52.5	71.9 µ	1459	27.8
20	1	3.58 µ	58.7	209 µ	4474	76.2
50	1	30.0 µ	59.8	1.77 m	8456	141
100	1	0.17	68.1	11.4 m	11916	175
500	1	13.7	78.6	1.08	18286	233
1000	1	97.7	74.2	7.25	20480	276

Table 89 Performance and energy to solution for the Euroben mod2am benchmark on the AMD A10-5800k APU. (PSNC)

Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
500×500	1	0.28	66.0	18.5	893	13.5
1000×1000	1	0.77	98.6	75.9	1299	13.2
1500×1500	1	1.85	97.1	180	1216	12.5
2000×2000	1	2.89	101.8	294	1384	13.6
2500×2500	1	4.52	103	465	1383	13.4
3000×3000	1	7.12	100.3	714	1264	12.6
3500×3500	1	9.82	100.8	989	1247	12.4
4000×4000	1	14.4	99.7	1437	1110	11.1

Table 90 Performance and energy to solution for the Hydro benchmark on the AMD A10-5800k APU. (PSNC)

Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
500×500	1	0.23	42.1	9.67	1087	25.8
1000×1000	1	0.54	55.5	29.9	1852	33.4
1500×1500	1	1.11	56.6	62.8	2027	35.8
2000×2000	1	3.56	57.2	203	1124	19.7

Table 91 Performance and energy to solution for the Hydro benchmark on the A10-5800k APU using the GPU. (PSNC)

Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
28000	1	422	111	46929	34.7	312

Table 92 Performance and energy to solution for the Linpack benchmark on the AMD A10-5800k APU. (PSNC)

Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
10	1	715 n	-	-	2796	-
20	1	2.38 μ	-	-	6711	-
50	1	11.0 μ	-	-	22795	-
100	1	57.2 μ	-	-	34952	-
500	1	3.91	-	-	64000	-
1000	1	29.3	-	-	68267	-

Table 93 Performance of the Euroben mod2am benchmark on the Intel Sandy Bridge CPU using the MKL library. (PSNC)

Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
10	1	6.20 μ	-	-	323	-
20	1	19.1 μ	-	-	839	-
50	1	260 μ	-	-	964	-
100	1	1.10	-	-	1820	-
500	1	125	-	-	2000	-
1000	1	938	-	-	2133	-

Table 94 Performance for the Euroben mod2am benchmark on the Intel Sandy Bridge CPU using the ACML library. (PSNC)

Matrix Size	Fill [%]	N	Time [μs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
100×100	3.5	1	0.75	-	-	940	-
200×200	3.75	1	2.03	-	-	1480	-
400×400	4.38	1	8.11	-	-	1727	-
600×600	4.44	1	19.1	-	-	1678	-
1000×1000	5	1	61.0	-	-	1638	-
2000×2000	7.5	1	397	-	-	1512	-

Table 95 Performance of the Euroben mod2as benchmark on the Intel Sandy Bridge CPU. (PSNC)

Size	N	Time [μs]	Power [W]	Energy [mJ]	Perf. [MF/s]	Eff. [MF/J]
2^8	1	7.63	-	-	1499	-
2^9	1	26.7	-	-	1099	-
2^{10}	1	32.4	-	-	1786	-
2^{11}	1	122	-	-	1162	-
2^{12}	1	160	-	-	1752	-
2^{13}	1	549	-	-	1212	-
2^{14}	1	732	-	-	1801	-
2^{15}	1	2319	-	-	1317	-
2^{16}	1	3662	-	-	1655	-
2^{17}	1	10254	-	-	1345	-
2^{18}	1	17578	-	-	1558	-
2^{19}	1	46875	-	-	1311	-
2^{20}	1	85938	-	-	1421	-

Table 96 Performance of the Euroben mod2f benchmark on the Intel Sandy Bridge CPU. (PSNC)

Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
10	1	954 n	52.6	50.1 μ	2097	39.9
20	1	5.72 μ	50.2	287 μ	2796	55.7
50	1	55.3 μ	53.6	2.96 m	4520	84.3
100	1	114 μ	65.8	7.53 m	17476	266
500	1	14.6	67.8	0.99	17067	252
1000	1	109	64.2	7.02	18286	285

Table 97 Performance and energy to solution for the Euroben mod2am benchmark on the Intel i7-3770 CPU. (PSNC)

Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
500×500	1	0.20	46.7	9.33	1250	26.8
1000×1000	1	0.49	78.1	38.3	2041	26.1
1500×1500	1	0.98	81.0	79.4	2296	28.4
2000×2000	1	1.65	81.6	135	2424	29.7
2500×2500	1	2.58	81.4	210	2422	29.8
3000×3000	1	3.74	81.5	305	2406	29.5
3500×3500	1	5.33	81.9	436	2298	28.1
4000×4000	1	6.99	81.2	568	2289	28.2

Table 98 Performance and energy to solution for the Hydro benchmark on the Intel i7-3770 CPU. (PSNC)

Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
28000	1	201	80.9	16263	72.8	900

Table 99 Performance and energy to solution for the Linpack benchmark on the Intel i7-3770 CPU. (PSNC)

8.3 Shared Memory Scaling Benchmark Results

8.3.1 *Shared memory through a cache-coherency add-in card (NUMA-CIC), UiO*

UiO submitted strong scaling results for the STREAM operations shown in Table 100, Table 101, Table 102 and Table 103. The results were calculated from average power values captured during all 4 STREAM operations of a given size and the bandwidth information for each operation. Each operation was repeated 10 times.

Threads	Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MB/s]	Eff. [MB/J]
23	$60 \cdot 10^9$	1	15085	4136	62392	63639	15.4
46		1	5361	4208	22558	179082	42.6
69		1	4528	4245	19223	212000	49.9
138		1	2390	4351	10399	401678	92.3
276		1	1593	4538	7229	602625	133
552		1	2019	5020	10131	475609	94.7

Table 100 Performance and energy to solution for the STREAM copy benchmark under strong scaling. (UiO)

Threads	Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MB/s]	Eff. [MB/J]
23	$60 \cdot 10^9$	1	15188	4136	62818	63207	15.3
46		1	5428	4208	22841	176864	42.0
69		1	4534	4245	19249	211713	49.9
138		1	2417	4351	10516	397215	91.3
276		1	1618	4538	7342	593412	131
552		1	2044	5020	10262	469614	93.5

Table 101 Performance and energy efficiency of the STREAM scale benchmark under strong scaling. (UiO)

Threads	Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MB/s]	Eff. [MB/J]
23	$60 \cdot 10^9$	1	22745	4136	94071	63312	15.3
46		1	7796	4208	32804	184720	43.9
69		1	6683	4245	28368	215484	50.8
138		1	3413	4351	14850	421903	97.0
276		1	2274	4538	10320	633213	140
552		1	3018	5020	15152	477088	95.0

Table 102 Performance and energy to solution of the STREAM sum benchmark under strong scaling. (UiO)

Threads	Size	N	Time [ms]	Power [W]	Energy [J]	Perf. [MB/s]	Eff. [MB/J]
23	$60 \cdot 10^9$	1	22694	4136	93864	63452	15.3
46		1	7728	4208	32518	186343	44.3
69		1	6654	4245	28248	216401	51.0
138		1	3392	4351	14760	424508	97.6
276		1	2265	4538	10278	635794	140
552		1	3045	5020	15284	472961	94.2

Table 103 Performance and energy to solution of the STREAM triad benchmark under strong scaling. (UiO)

8.4 Distributed Memory Scaling Benchmark Results

8.4.1 *An NVIDIA Tegra 2 mobile SoC based HPC cluster, BSC*

Alongside the node-centric results in the previous section, BSC also provided scaling studies for Euroben mod2am in Table 104 for Hydro in Table 105 and for Linpack in Table 106.

Nodes	Size	N	Time [s]	Power [W]	Energy [J]	Perf. [MF/s]	Eff. [MF/J]
1	1000	1	9.45	9.16	86.5	212	23.1
8	4000	1	104	71.5	7426	1232	17.2
16	8000	1	449	142	63701	2282	16.1

Table 104 Performance and energy to solution of the Euroben mod2am benchmark scaling across nodes. (BSC)

Nodes	Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
1	500×500	1	1.06	8.05	8.53	236	29.3
2	1000×1000	1	2.45	16.5	40.3	409	24.8
4	2000×2000	1	4.51	34.7	156	886	25.6
8	2500×2500	1	3.97	75.1	298	1574	21.0
32	5000×5000	1	3.88	300	1164	6442	21.5
128	10000×10000	1	3.98	1201	4785	25102	20.9
128	12000×12000	1	5.97	1197	7145	24120	20.2
192	12000×12000	1	5.45	1735	9453	26432	15.2

Table 105 Performance and energy to solution of the Hydro benchmark scaling across nodes. (BSC)

Nodes	Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
1	9920	1	575	7.99	4599	1.1	142
2	13920	1	834	16.2	13548	2.2	133
4	19840	1	1199	33.4	40010	4.3	130
8	28160	1	1754	71.3	125050	8.5	119
16	39680	1	2406	141	339734	17.3	123
32	56000	1	3506	283	993543	33.4	118
64	57600	1	2022	562	1137010	63.0	112
96	97120	1	6166	842	5192070	99.0	118
128	112160	1	7423	1127	8363050	127	112
160	125440	1	8676	1405	12191500	152	108
192	137440	1	9892	1656	16380100	175	106
224	147840	1	10798	1966	21226700	200	101

Table 106 Performance and energy to solution for the Linpack benchmark scaling across nodes. (BSC)

8.4.2 *GPU-GPU communication over PCIe and IB, CaSToRC*

CaSToRC provided a strong scaling study for Hydro on GPUs in Table 107. Scaling results for the Linpack benchmark are presented in Table 108; the values in this data set are averaged over 3 – 4 different blocking factors for each problem size. The exceptional low power reading for the 1 node Linpack benchmark is most likely an artefact of the short benchmark run-time and is currently under investigation.

Nodes	Size [x×y]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
1	12288×12288	1	6.19	386	2391	24399	63.1
2		2	3.94 ± 35.40%	738 ± 7.49%	2907 ± 36.18%	38336 ± 35.40%	51.9 ± 36.18%
3		1	2.57	1159	2974	58844	50.8
4		1	1.65	1561	2577	91460	58.6
5		1	1.46	1978	2881	103663	52.4
6		1	1.42	2310	3269	106703	46.2
8		1	0.75	3146	2360	201327	64.0

Table 107 Performance and energy to solution for the Hydro benchmark under strong scaling. (CaSToRC)

P	Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
1	12288	1	12.0	150	1797	103	689
2	24576	1	27.5	751	20662	360	479
3	36864	1	44.2	1298	57448	755	581
4	49152	1	70.0	1894	132587	1131	597
5	61440	1	97.2	2573	250198	1590	618
6	73728	1	126	3248	408446	2125	654
7	86016	1	161	3895	628120	2631	675
8	98304	1	209	4490	936884	3035	676

Table 108 Performance and energy to solution for the Linpack benchmark scaling across nodes. (CaSToRC)

8.4.3 *Holistic approach to energy efficiency, LRZ*

LRZ provided a scaling study for the Hydro benchmark shown in Table 109 as well as results for the weak scaling of Linpack in Table 110.

Node	Size [kU]	N	Time [s]	Power [W]	Energy [J]	Perf. [kU/s]	Eff. [kU/J]
1	5000×5000	1	1.76	256	451	14204	55.4
2	10000×10000	1	3.89	547	2130	25688	47.0
4	20000×20000	1	8.63	1092	9430	46334	42.4
8	40000×40000	1	20.7	2182	45227	77181	35.4
16	80000×80000	1	52.4	4296	225119	122141	28.4

Table 109 Performance and energy to solution for the Hydro benchmark scaling across nodes. (LRZ)

Nodes	Size	N	Time [s]	Power [W]	Energy [J]	Perf. [GF/s]	Eff. [MF/J]
1	41328	1	434	333	297000	108	326
2	58464	1	668	696	965880	200	286
4	82824	1	912	1332	2468880	415	312
8	117096	1	1306	2602	6872760	820	315
16	165648	1	2376	5114	24924600	1275	249
32	234360	1	3857	10209	80038800	2225	218

Table 110 Performance and energy to solution for the Linpack benchmark under weak scaling across nodes. (LRZ)