

Energy Efficiency of Mixed Precision Iterative Refinement Methods using Hybrid Hardware Platforms: An Evaluation of different Solver and Hardware Configurations

Hartwig Anzt
Vincent Heuveline
Björn Rocker

No. 2010-03

Preprint Series of the Engineering Mathematics and Computing Lab (EMCL)





Preprint Series of the Engineering Mathematics and Computing Lab (EMCL)
ISSN 2191-0693
No. 2010-03

Impressum

Karlsruhe Institute of Technology (KIT)
Engineering Mathematics and Computing Lab (EMCL)

Fritz-Erler-Str. 23, building 01.86
76133 Karlsruhe
Germany

KIT – University of the State of Baden Wuerttemberg and
National Laboratory of the Helmholtz Association

Published on the Internet under the following Creative Commons License:
<http://creativecommons.org/licenses/by-nc-nd/3.0/de> .



www.emcl.kit.edu

Energy Efficiency of Mixed Precision Iterative Refinement Methods using Hybrid Hardware Platforms: An Evaluation of different Solver and Hardware Configurations

Hartwig Anzt, Björn Rocker and Vincent Heuveline

Karlsruhe Institute of Technology (KIT)
Institute for Applied and Numerical Mathematics 4
Fritz-Erler-Str. 23
76133 Karlsruhe, Germany

hartwig.anzt@kit.edu, bjoern.rocker@kit.edu,
vincent.heuveline@kit.edu

Abstract. In this paper we evaluate the possibility of using mixed precision algorithms on different hardware platforms to obtain energy-efficient solvers for linear systems of equations. Our test-cases arise in the context of computational fluid dynamics.

Therefore, we analyze the energy efficiency of common cluster nodes and a hybrid, GPU-accelerated cluster node, when applying a linear solver, that can benefit from the use of different precision formats.

We show the high potential of hardware-aware computing in terms of performance and energy efficiency.

1 Introduction

1.1 Motivation

Energy efficiency is relevant for large computing centers running high performance platforms as well as for individual desktop solutions. In the meantime, the energy cost often dominates the total expenses of scientific computing centers. Already after a few years, the energy cost for running a certain hardware platform usually exceeds the acquisition cost [1]. Therefore it is a point of major interest approaching the Exascale Computing Era [2].

While many different architectures with different properties concerning power consumption are available on the market, the architecture alone is not the governing parameter determining, whether a problem can be solved efficiently. The reason is, that algorithms have to be adapted to both, the specific problems and the specific hardware configuration, to achieve high efficiency. As an example, we show in this paper how a mixed precision error correction method [3] for

solving linear systems of equations behaves on three widespread used architectures. These so-called mixed precision solvers use single precision floating point operations instead of double precision floating point operations for large parts of the algorithm without losing accuracy of the final result. This leads to an acceleration of the solver, since in case of bandwidth limited applications, the speedup when switching from double to single precision approximates the factor of two, while for computationally intensive tasks, even higher speedups can be achieved.

1.2 Paper Organization

Since the solving process of linear equation systems often demands most of the resources when performing numerical simulations, both in time and energy, we analyze the possibility of saving energy by using more efficient linear solvers. For many problems, the idea of applying a mixed precision iterative refinement variant instead of a plain solver leads to savings concerning the computational effort and the energy need (section 2).

In a second step, we combine the idea of a mixed precision algorithm with the idea of evaluating different hardware platforms. Different systems are analyzed with respect to their specific floating point performance in single and double precision and their energy consumption (section 3). In the following (section 4), the computational effort and the energy need of different linear solver- and hardware-configurations are measured for problems occurring in the field of numerical simulations. In the end (section 5), we conclude and give a brief overview about the potential of future hardware technology in terms of energy-efficient computing.

2 Mixed Precision Iterative Refinement Methods

One basic principle in energy efficient computing is the acceleration of linear solvers and the reduction of the energy consumption by cutting the corresponding overall computation time. Research has shown, that for many linear problems, an acceleration of the applied linear solvers is possible through the use of different floating point precision formats. The underlying idea is to use the linear solver as inner solver of an iterative refinement method. The acceleration is possible since the inner solver can be performed in a less complex floating point format than working precision, without losing accuracy of the final result.

The motivation for the error correction approach can be obtained from Newton's method. Here, f is a given function and x_i is the solution in the i -th step:

$$x_{i+1} = x_i - (\nabla f(x_i))^{-1} f(x_i). \quad (1)$$

This method can be applied to the function $f(x) = b - Ax$ with $\nabla f(x) = A$, where $Ax = b$ is the linear system that should be solved.

By defining the residual $r_i := b - Ax_i$, one obtains

$$\begin{aligned} x_{i+1} &= x_i - (\nabla f(x_i))^{-1} f(x_i) \\ &= x_i + A^{-1}(b - Ax_i) \\ &= x_i + A^{-1}r_i. \end{aligned}$$

Denoting the solution update with $c_i := A^{-1}r_i$ and using an initial guess x_0 as starting value, an iterative algorithm can be defined, where any linear solver can be used as error correction solver.

- 1: initial guess as starting vector: x_0
- 2: compute initial residual: $r_0 = b - Ax_0$
- 3: **while** ($\| Ax_i - b \|_2 > \varepsilon \| r_0 \|$) **do**
- 4: $r_i = b - Ax_i$
- 5: solve error correction equation: $Ac_i = r_i$
- 6: update solution: $x_{i+1} = x_i + c_i$
- 7: **end while**

Algorithm 1: Error Correction Method

In each iteration, the inner correction solver searches for a c_i such that $Ac_i = r_i$ and the solution approximation is updated by $x_{i+1} = x_i + c_i$.

If the inner error correction solver is performed in less complex floating point format than working precision, one obtains a mixed precision error correction method, updating the solution approximation in high precision, but computing the error correction term in a lower precision format. This approach was also suggested by [4],[5], [6], and [7].

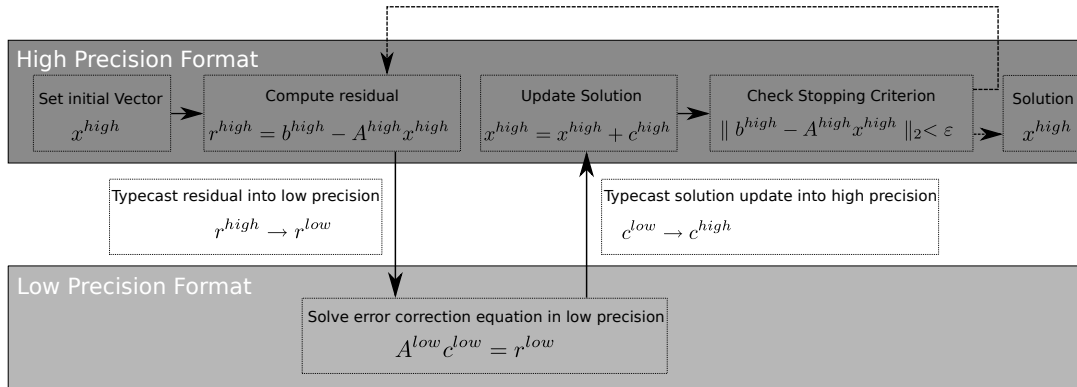


Fig. 1: Visualizing the mixed precision approach to an error correction solver

For many problems the linear solver can be accelerated by applying this mixed precision iterative refinement approach, see [8]. This acceleration leads to a reduced overall computation time. In Section 4.2, we will investigate whether the acceleration of the solver also leads to lower energy consumption when solving a linear problem.

3 Hardware Platforms

For our performance evaluation and the consideration of energy efficiency, we have chosen nodes of two HPC-cluster based on different generations of Xeon processors and one system accelerated by an Nvidia Tesla S1070. Both clusters used for our comparison are located at the Steinbuch Centre for Computing (SCC) ¹ at the Karlsruhe Institute for Technology (KIT) ².

The first machine is the InstitutsCluster IC1. It consists of five racks containing a total of 200 computing nodes, each equipped with two Intel quad-core Xeon 5355 processors with Clovertown architecture running at 2.667 GHz. There are 16 GB of main memory available on each node.

The second machine is a HP XC3000 cluster system, called HC3. In total, the cluster has 332 computation nodes, each equipped with two Intel quad-core Xeon 5540 with Nehalem architecture. 288 nodes own 24 GB of main memory, 32 own 48 GB and the last 12 are equipped with 144 GB.

Our Tesla-based system consists of two nodes hosting two Intel Xeon 5450 processors operating at 3.0 GHz. A Tesla S1070 is connected via PCIe 2.0 16x and each node controls two Tesla T10 computing processors, both equipped with 4 GB of memory.

Due to the fact that in our experiments we are only using the computing capabilities of one node and one T10 computing processor, our Tesla-system is in terms of performance equivalent to a system equipped with one Tesla C1060. For this reason, the energy consumption is calculated for the host alone and for one Tesla C1060.

¹ www.scc.kit.edu

² www.kit.edu

Table 1: Key system characteristics of the three machines used for the tests. The first five rows give an overview for one node followed by additional information concerning the full cluster systems HC3 and IC1.

	HC3	Tesla ^a	IC1
Processors per node	2	2 CPUs / 1 GPU	2
Cores per processor	4	8 / 240	4
Theoretical comp. rate / core	10.1 GFlop/s	12 / 3.9 GFlop/s	10.7 GFlop/s
Theoretical comp. rate / node	81 GFlop/s	96 / 933 GFlop/s	85.3 GFlop/s
L2-cache per processor	8 MB	8 / - MB	8 MB
Nodes	278 / 32 / 12	1	200
Memory per node	24 / 48 / 144 GB	32 GB	16 GB
Memory full machine	10.3 TB	32 GB	32 TB
Theoretical comp. rate full machine	27 TFlop/s	1.0 TFlop/s	17.6 TFlop/s
Power consumption load full machine	80.8 kW	539 + 187,8 W	103 kW
Power consumption load per node	244 W	539 + 187,8 W	514 W

^a http://www.nvidia.com/object/product_tesla_c1060_us.html

4 Numerical Experiments

4.1 Test Configuration

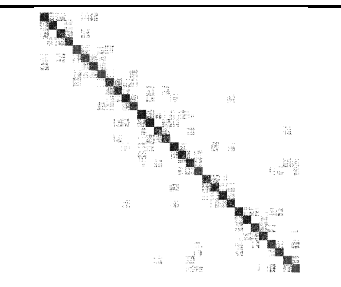
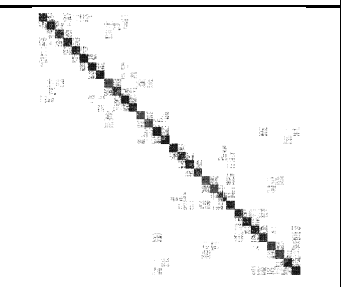
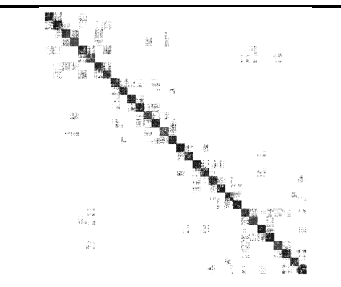
To be able to compare the performance of a plain solver and the mixed precision error correction variant on different hardware platforms, we use an implementation of the GMRES-(10) solver. In this work -(10) denotes the restart parameter for the GMRES algorithm.

Both, the plain double GMRES-(10) and the mixed precision variant use the relative residual stopping criterion of $\varepsilon = 10^{-10} \| r_0 \|_2$, while we choose $\varepsilon_{\text{inner}} = 10^{-1} \| r_0 \|_2$ as inner stopping criterion for the error correction variant. Due to the iterative residual computation in the case of the plain GMRES-(10) solvers, the mixed GMRES-(10) solvers based on the mixed precision error correction method usually iterate to a better approximation, since they compute the residual error explicitly. But as the difference is generally small, the solvers can be compared.

In case of the mixed precision GMRES-(10) implementation on the Tesla-system, the error correction solver is performed on the GPU, while the solution update is led to the CPU of the same system. This is done to be able to handle larger problems since the available memory on the GPU is limited to 4 GB.

As test problems, we use three systems of linear equations CFD1, CFD2 and CFD3 affiliated with the 2D modeling of a Venturi Nozzle in different discretization fineness. The distinct number of supporting points leads to different matrix characteristics concerning the dimension, the number of non-zeros, and the condition number. In Table 2, the nonzero structures of the obtained sparse linear systems are visualized, and the matrix properties are summarized.

Table 2: Sparsity plots showing the nonzero-structure of the CFD test-matrices.

CFD1	CFD2	CFD3
		
problem: 2D fluid flow problem size: $n = 395009$ sparsity: $nnz = 3544321$ storage format: CRS	problem: 2D fluid flow problem size: $n = 634453$ sparsity: $nnz = 5700633$ storage format: CRS	problem: 2D fluid flow problem size: $n = 1019967$ sparsity: $nnz = 9182401$ storage format: CRS

4.2 Performance Results

To ensure the high quality of the performance results, the algorithms are executed exclusively on the machines such that no effects of other jobs executed on the same device may occur.

Table 3: Computation time for problem CFD 1 based on a GMRES-(10) as inner solver for the error correction method.

		CPU-Cores			GPU
		1	4	8	
Computation time in s	HC3 double	2267.47	1245.12	776.09	
	HC3 mixed	886.46	567.51	309.61	
	IC1 double	3146.61	1656.53	1627.77	
	IC1 mixed	1378.56	712.83	659.80	
	Tesla mixed				

Table 4: Computation time for problem CFD 2 based on a GMRES-(10) as inner solver for the error correction method.

		CPU-Cores			GPU
		1	4	8	
Computation time in s	HC3 double	10765.30	4528.09	3363.44	
	HC3 mixed	4827.98	2177.19	1648.27	
	IC1 double	13204.70	6843.66	6673.07	
	IC1 mixed	5924.32	3495.09	3681.28	
	Tesla mixed				

Table 5: Computation time for problem CFD 3 based on a GMRES-(10) as inner solver for the error correction method.

		CPU-Cores			GPU
		1	4	8	
Computation time in s	HC3 double	62210.70	19954.50	16541.90	
	HC3 mixed	42919.80	9860.26	8828.28	
	IC1 double	60214.50	32875.10	32576.50	
	IC1 mixed	41927.40	19317.00	19836.80	
	Tesla mixed				

For the large problem CFD 3, we observe a speedup factor of around 1.5 when switching from the plain double implementation to the mixed precision error correction implementation on one machine (Table 5). For the smaller problems, even higher speedups can be achieved (Table 3 and Table 4).

Different effects are able to reason this decrease in speedup. On the one hand, the condition number of the systems is not determined, and a smaller condition number of the larger systems may cause a less efficient use of the mixed precision approach [3]. From the technical point of view, the memory bandwidth between the different cache and memory levels becomes the bottleneck when performing the algorithm on large data, limiting the speedup factor between mixed and plain double implementation down to a factor of smaller than two. For small problems, even higher speedups can be achieved, since, that as long as the bandwidth is not the limiting factor, and the processing units are able to perform four single precision operations instead of one double precision operation per cycle. Further investigation is necessary in this point.

4.3 Energy Efficiency

By using the values given in Table 1 for the power consumption P under load of the different architectures, we obtain 244 W per node for the HC3, 514 W per node for the IC1 and 718 W for the Tesla-system (node plus energy for one Tesla C1060). With these values, it is possible to estimate the total amount of energy E that has been spent for a computation that has taken t seconds through the relation $E = P \cdot t$. The function indicates a linear characteristic due to the fact that we use one node, assuming a constant energy consumption, not taking into account whether one or more cores were used.

The energy the system needs in idle mode may be a point of interest as well. Therefore, both, a deeper analysis of dynamical power usage of modern processors and the influence of job-scheduling mechanisms is necessary. Still, we limit our analysis on the energy consumption of the evaluated mixed precision solvers on different hardware platforms.

Modern processors usually offer the possibility of automatically raising the clockspeed if sequential code is executed, and deactivating parts of the processor, if they are not needed. The power consumption is effected by such mechanisms,

and energy measurements have to be performed for every run. This becomes difficult when a machine is in production mode. On the HC3, measurements have shown an energy consumption of 243,5 W per node in case of performing complex computations using most components of the processor, and a power consumption of 225 W for less complex computations using only little resources. The system setting for the CPU-frequency is “ondemand”. A deeper analyses of the machines may give more detailed information, but we can assume, that the energy consumption will generally stay within these limits.

Table 6: Energy consumption for problem CFD 1 based on a GMRES-(10) as inner solver for the error correction method.

		CPU-Cores			GPU
		1	4	8	
Energy consumption in Wh	HC3 double	153.37	84.22	52.49	
	HC3 mixed	59.96	38.39	20.94	
	IC1 double	449.27	236.52	232.41	
	IC1 mixed	196.83	101.78	94.2	
	Tesla mixed				87.36

Table 7: Energy consumption for problem CFD 2 based on a GMRES-(10) as inner solver for the error correction method.

		CPU-Cores			GPU
		1	4	8	
Energy consumption in Wh	HC3 double	728.15	306.27	227.50	
	HC3 mixed	326.56	147.26	111.49	
	IC1 double	1885.34	977.12	952.77	
	IC1 mixed	845.86	499.02	525.60	
	Tesla mixed				417.29

Table 8: Energy consumption for problem CFD 3 based on a GMRES-(10) as inner solver for the error correction method.

		CPU-Cores			GPU
		1	4	8	
Energy consumption in Wh	HC3 double	4207.86	1349.70	1118.88	
	HC3 mixed	2903.05	666.94	597.14	
	IC1 double	8597.29	4693.83	4651.20	
	IC1 mixed	5986.30	2758.04	2832.25	
	Tesla mixed				2057.04

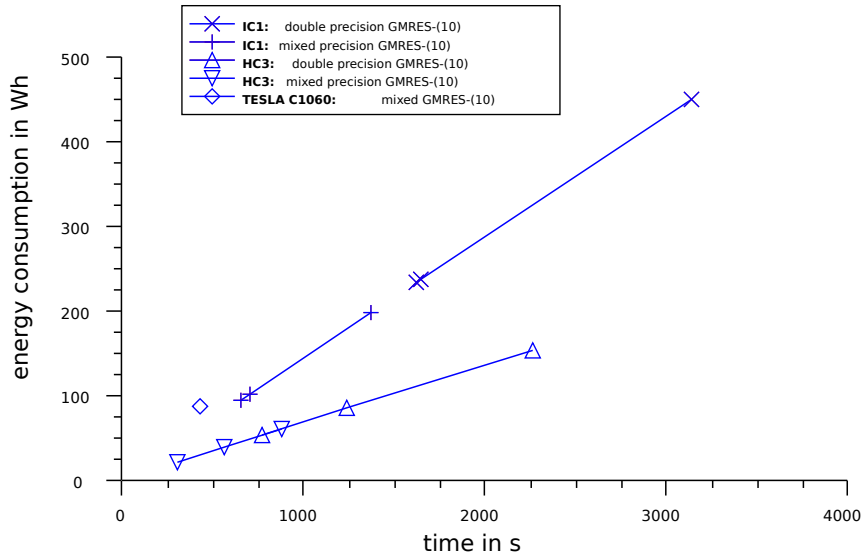


Fig. 2: Energy consumption as a function of time for solving the CFD1 test-case on HC3, IC1 and Tesla. The inner solver is a GMRES-(10). In case of the IC1 and HC3, the results for the double and mixed precision implementations using 1/4/8 cpu-cores are plotted, in case of the Tesla, the mixed precision approach is executed.

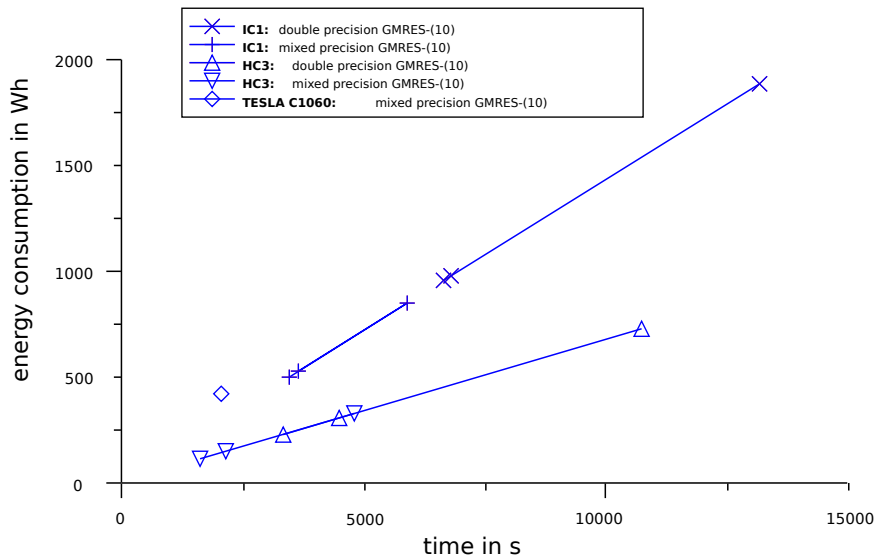


Fig. 3: Energy consumption as a function of time for solving the CFD2 test-case on HC3, IC1 and Tesla. The inner solver is a GMRES-(10). In case of the IC1 and HC3, the results for the double and mixed precision implementations using 1/4/8 cpu-cores are plotted, in case of the Tesla, the mixed precision approach is executed.

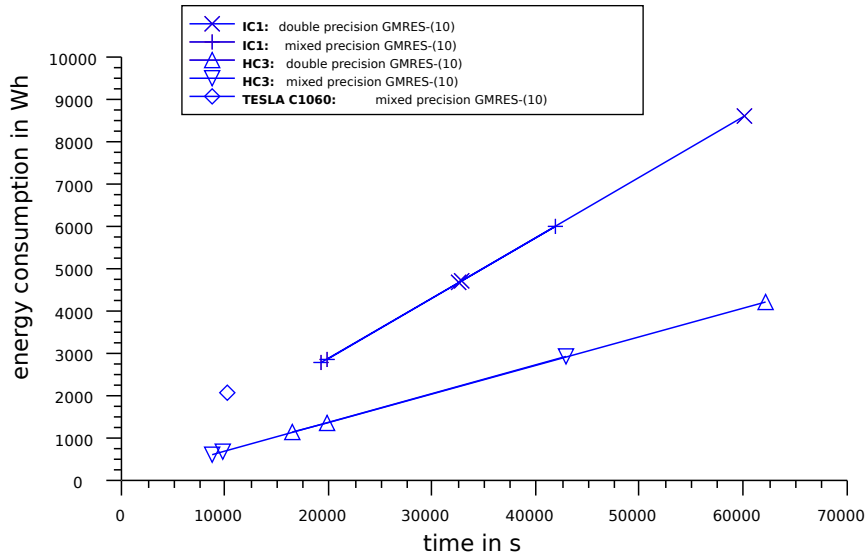


Fig. 4: Energy consumption as a function of time for solving the CFD3 test-case on HC3, IC1 and Tesla. The inner solver is a GMRES-(10). In case of the IC1 and HC3, the results for the double and mixed precision implementations using 1/4/8 cpu-cores are plotted, in case of the Tesla, the mixed precision approach is executed.

The Tesla-system is based on an old CPU-architecture, comparable to the IC1-cluster. Our experiments have shown that even older architectures can greatly benefit by adding accelerators like GPUs, both in terms of performance and energy efficiency. The GPU used in the Tesla-system is based on the old T10 computing chip, and with the availability of the new Fermi-based GPUs, even the newest CPU-architectures will be outperformed.

5 Conclusion and Future Work

The iterative refinement method implemented with mixed precision techniques, and combined with GPU coprocessor technology, shows very good results for our test-cases, arising in the field of computational fluid dynamics. For this reason, for problems similar to those, a mixed precision error correction implementation is advisable to achieve better performance, both in energy efficiency and computation time.

This also reveals the high potential of hardware-aware computing. When numerical procedures are developed and implemented with respect to the available hardware resources, we can expect advantages in terms of performance and energy efficiency. Since, depending on the problem, the time and energy savings can become quite large, hardware-aware computing should be taken into account for individual solutions as well as for large-scale simulations in computing centers.

Future work would include a larger set of experiments concerning the linear systems, the used solvers, and the evaluated hardware platforms. One focus could also be to monitor the energy saving techniques of modern processors.

Acknowledgements

We would like to thank Frauke Bösert from the SCC³ for supporting us with measurements for the energy consumption of both cluster systems HC3 and IC1.

References

1. Frank Lampe. *Mit neuen IT-Technologien Energieeffizienz erreichen, die Umwelt schonen und Kosten sparen*. Vieweg+Teubner | GWV Fachverlage GmbH, Wiesbaden, 2010.
2. DARPA/IPTO. Exascale computing study: Technology challenges in achieving exascale systems. Technical report, 2008.
3. Hartwig Anzt, Björn Rocker, and Vincent Heuveline. Mixed precision error correction methods for linear systems: Convergence analysis based on krylov subspace methods. Proceedings of PARA 2010 State of the Art in Scientific and Parallel Computing, 2010.
4. D. Göttsche, R. Strzodka, and S. Turek. Performance and accuracy of hardware-oriented native-, emulated- and mixed-precision solvers in fem simulations. *International Journal of Parallel, Emergent and Distributed Systems (IJPEDS), Special issue: Applied parallel computing, Volume 22*.
5. D. Göttsche and R. Strzodka. Performance and accuracy of hardware-oriented native-, emulated- and mixed-precision solvers in FEM simulations (part 2: Double precision gpus). Technical report, Fakultät für Mathematik, TU Dortmund, 2008.
6. Alfredo Buttari, Jack J. Dongarra, Julie Langou, Julien Langou, Piotr Luszczek, and Jakub Kurzak. Mixed precision iterative refinement techniques for the solution of dense linear systems. *The International Journal of High Performance Computing Applications, Volume 21, No. 4*, 2007.
7. Marc Baboulin, Alfredo Buttari, Jack J. Dongarra, Julie Langou, Julien Langou, Piotr Luszczek, Jakub Kurzak, and Stanimire Tomov. Accelerating scientific computations with mixed precision algorithms. *Computer Physics Communications Volume 180*, 2008.
8. Hartwig Anzt, Björn Rocker, and Vincent Heuveline. An error correction solver for linear systems: Evaluation of mixed precision implementations. Proceedings of VECPAR 2010 - 9th International Meeting on High Performance Computing for Computational Science, 2010.

³ www.scc.kit.edu

Preprint Series of the Engineering Mathematics and Computing Lab

recent issues

- No. 2010-02 Hartwig Anzt, Vincent Heuveline, Björn Rucker: Mixed Precision Error Correction Methods for Linear Systems: Convergence Analysis based on Krylov Subspace Methods
- No. 2010-01 Hartwig Anzt, Vincent Heuveline, Björn Rucker: An Error Correction Solver for Linear Systems: Evaluation of Mixed Precision Implementations
- No. 2009-02 Rainer Buchty, Vincent Heuveline, Wolfgang Karl, Jan-Philipp Weiß: A Survey on Hardware-aware and Heterogeneous Computing on Multicore Processors and Accelerators
- No. 2009-01 Vincent Heuveline, Björn Rucker, Staffan Ronnas: Numerical Simulation on the SiCortex Supercomputer Platform: a Preliminary Evaluation