Application readiness at CSCS

Thomas C. Schulthess
Swiss Platform for High-Performance and High-Productivity Computing \textbf{HP2C}

**Overarching goal**

Prepare computational sciences to make effective use of next generation supercomputers

**Specific goal**

Emerge with several high-impact scientific applications that scale and run efficiently on leadership computing platforms in 2012/13 timeframe

Build on, and multiply the early science applications experience on Jaguar at ORNL in 2008

- DCA++: simulate models of high-temperature superconductivity first sustained petaflop/s in production runs (Gordon Bell Prize 2008)
- WL-LSMS: simulate thermodynamics properties in magnetic nanoparticles sustained petaflop/s in production runs (Gordon Bell Prize 2009)
Background HP2C

- Funded by the Swiss University Conference as a structuring project
  - strong networking component is a plus/requirement

- Approach:
  - fund HPC developers that are embedded in application development teams at the Universities
  - maintain future systems competence at CSCS as well as staff that can engage with domain science teams – avoid taking responsibility for codes
  - create interdisciplinary development projects with domain scientists, applied mathematicians, and computer scientists

- Open call for project proposals in summer 2009, peer review, project selection, and project kickoff in March 2010

- Projects will run through June 2013
  - a new platform that will succeed HP2C has been approved and will run through 2016
Projects of the HP2C platform (see www.hp2c.ch)

- Gyrokinetic Simulations of Turbulence in Fusion Plasmas (ORB5) – Laurent Villard, EPF Lausanne
- Ab initio Molecular Dynamics (CP2K) – Jürg Hutter, U. of Zurich
- Computational Cosmology on the Petascale – Geoge Lake, U. of Zurich
- Selectome, looking for Darwinian evolution in the tree of life – Marc Robinson-Rechavi, Univ. of Lausanne
- Cardiovascular Systems Simulations (LifeV) – Alfio Quarteroni, EPF Lausanne
- Large-Scale Parallel Nonlinear Optimization for High Resolution 3D-Seismic Imaging (Petaquacke) – Olaf Schenk, Univ. of Basel
- 3D Models of Stellar Explosions – Matthias Liebendörfer, Univ. of Basel
- Large Scale Electronic Structure Calculations (BigDFT) – Stefan Gödecker, Univ. of Basel
- Regional Climate & Weather Model (COSMO) – Isabelle Bey, ETH Zurich/C2SM
- Modeling humans under climate stress – Christoph Zollikhofer, U. of Zurich
Physical model

Mathematical description

\[ \text{lap}(i,j,k) = -4.0 \times \text{data}(i,j,k) + \text{data}(i+1,j,k) + \text{data}(i-1,j,k) + \text{data}(i,j+1,k) + \text{data}(i,j-1,k); \]

Discretization / algorithm

Code / implementation

“Port” serial code to supercomputers
- vectorize
- parallelize
- petascaling
- exascaling
- ...

Code compilation

A given supercomputer

Tuesday, October 9, 2012

Cray Technical Workshop on XK6 Programming, ORNL
Physical model

Mathematical description

Needs to be properly embedded

Driven by domain science

Discretization / algorithm

Libraries
DSL / DSEL

Code / implementation

Optimal algorithm
Auto tuning

Driven by vendors

Code compilation

Architectural options / design

\[ \text{lap}(i,j,k) = -4.0 \times \text{data}(i,j,k) + \text{data}(i+1,j,k) + \text{data}(i-1,j,k) + \text{data}(i,j+1,k) + \text{data}(i,j-1,k); \]
WHILE IN THIS PRESENTATION OUR FOCUS IS ON RESULTS AND DEVELOPMENTS ON CRAY’S XE6/XK6, THE HP2C PROJECTS AND APPLICATION READINESS ACTIVITIES AT CSCS HAVE BEEN COVERING A BROAD RANGE OF PLATFORMS, INCLUDING IBM’S BG/Q SYSTEM AND SYSTEMS BUILT WITH INTEL’S XEON AND XEON-PHI PROCESSORS
CPU-GPU-Hybrid study

- After acceptance of Cray XK6 in fall 2012, start a study of application performance on Hybrid platforms (kickoff Dec. 1, 2012)
- Selected 6 HP2C teams and three additional teams
  - CP2K – chem./mat. sci.; U. of ZH + ETH-Z
  - Astro – astro./cosmology; U. of ZH
  - COSMO – climate/weather; ETH-Z + Meteo CH
  - MAQUIS – chem./mat. sci.; U of GE + EPFL + ETH-Z
  - Petaquake – Geophysics; U of BS + ETH-Z
  - BigDFT – chem./mat. sci.; Uof BS
  - MRAG – engineering; ETH-Z
  - GROMACS – life sci.; KTH (Sweden)
  - MAGMA eigenvalue solvers – all DFT codes; UTK/ETH-Z
- Timeline:
  - Report on porting status in January 2012
  - First report on performance of XK6 vs. XE6 at ACSS workshop in March 2012
  - Continue evaluation (including Xeon-Phi*) though Q1/2013

*NDA, so no results will be shown
# 9 projects and 15 applications: status Jan. 27, 2012

<table>
<thead>
<tr>
<th>Project</th>
<th>Code</th>
<th>User community</th>
<th>Rosa (XE6)</th>
<th>Tödi (XK6)</th>
<th>HMC PE&amp;Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP2K</td>
<td>CP2K</td>
<td>very large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>Cosmology</td>
<td>RAMSES</td>
<td>large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA, OpenACC</td>
</tr>
<tr>
<td>Cosmology</td>
<td>PKDGRAV</td>
<td>medium</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>Cosmology</td>
<td>GENGA</td>
<td>medium</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>CFD</td>
<td>MRAG</td>
<td>small</td>
<td>runs</td>
<td>runs</td>
<td>CUDA, OpenCL</td>
</tr>
<tr>
<td>GROMACS</td>
<td>GROMACS</td>
<td>very large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>Comp. Materials</td>
<td>DCA++</td>
<td>small</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>Comp. Materials</td>
<td>MAGMA solvers</td>
<td>very large</td>
<td>n.a.</td>
<td>runs</td>
<td>CUDA/CuBLAS</td>
</tr>
<tr>
<td>Comp. Materials</td>
<td>ELK/EXCITING</td>
<td>very large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
</tbody>
</table>

**Remarks:**

The XK6 was an entirely new architecture, new software stack, etc.

It took only 8 weeks, which included the Christmas holidays, to get apps. up and running.

We were surprised by the level of adoption of CUDA 
(HP2C funds application developers to become experts in HPC and they do what they do)
## 9 projects and 15 applications: status Jan. 27, 2012

Target apps on which we report performance in this presentation

<table>
<thead>
<tr>
<th>Project</th>
<th>Code</th>
<th>User community</th>
<th>Rosa (XE6)</th>
<th>Tödi (XK6)</th>
<th>HMC PE&amp;Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP2K</td>
<td>CP2K</td>
<td>very large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>Cosmology</td>
<td>RAMSES</td>
<td>large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA, OpenACC</td>
</tr>
<tr>
<td>Cosmology</td>
<td>PKDGRAV</td>
<td>medium</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>Cosmology</td>
<td>GENG A</td>
<td>medium</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>C2SM</td>
<td>COS MO</td>
<td>large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA, OpenACC</td>
</tr>
<tr>
<td>MAQUIS</td>
<td>ED</td>
<td>small</td>
<td>runs</td>
<td>no attempt</td>
<td>n.a.</td>
</tr>
<tr>
<td>MAQUIS</td>
<td>DMRG</td>
<td>small</td>
<td>no attempt</td>
<td>no attempt</td>
<td>n.a.</td>
</tr>
<tr>
<td>MAQUIS</td>
<td>SE (VLI)</td>
<td>small</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>Petaquake</td>
<td>SPECFEM 3D</td>
<td>large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA, OpenACC</td>
</tr>
<tr>
<td>BigDFT</td>
<td>BigDFT</td>
<td>large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA, OpenCL</td>
</tr>
<tr>
<td>CFD</td>
<td>MRAG</td>
<td>small</td>
<td>runs</td>
<td>runs</td>
<td>CUDA, OpenCL</td>
</tr>
<tr>
<td>GROMACS</td>
<td>GROMACS</td>
<td>very large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>Comp. Materials</td>
<td>DCA++</td>
<td>small</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
<tr>
<td>Comp. Materials</td>
<td>MAGMA solvers</td>
<td>very large</td>
<td>n.a.</td>
<td>runs</td>
<td>CUDA/CuBLAS</td>
</tr>
<tr>
<td>Comp. Materials</td>
<td>ELK/EXCITING</td>
<td>very large</td>
<td>runs</td>
<td>runs</td>
<td>CUDA</td>
</tr>
</tbody>
</table>
COSMO in production for Swiss weather prediction

- **COSMO-7**
  - 3x per day 72h forecast
  - 6.6 km lateral grid, 60 layers

- **ECMWF**
  - 2x per day
  - 16 km lateral grid, 91 layers

- **COSMO-2**
  - 8x per day 24h forecast
  - 2.2 km lateral grid, 60 layers

Some of the products generated from these simulations:

- Daily weather forecast
- Forecasting for air traffic control (Sky Gide)
- Safety management in event of nuclear incidents
Performance profile of (original) COSMO-CCLM

Runtime based 2 km production model of MeteoSwiss

% Code Lines (F90)  % Runtime

Original code (with OpenACC for GPU)  Rewrite in C++ (with CUDA backend for GPU)
Analyzing the two examples – how are they different?

Physics

\[
\begin{align*}
\text{do } & j = 1, \text{ niter} \\
\text{do } & i = 1, \text{ nwork}
  \quad c(i) = a(i) \times b(i) + \sin(b(i)) \times \log(a(i)) \\
\text{end do} \\
\text{end do}
\end{align*}
\]

3 memory accesses
136 FLOPs
⇒ compute bound

Dynamics

\[
\begin{align*}
\text{do } & j = 1, \text{ niter} \\
\text{do } & i = 1, \text{ nwork}
  \quad c(i) = a(i) + b(i) \times (a(i+1) - 2.0d0 \times a(i) + a(i-1)) \\
\text{end do} \\
\text{end do}
\end{align*}
\]

3 memory accesses
5 FLOPs
⇒ memory bound

- Arithmetic throughput is a **per core resource** that scales with number of cores and frequency
- Memory bandwidth is a **shared resource between cores** on a socket
Running the simple examples on the Cray XK6

### Compute bound (physics) problem

<table>
<thead>
<tr>
<th>Machine</th>
<th>Interlagos</th>
<th>Fermi (2090)</th>
<th>GPU+transfer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>1.31 s</td>
<td>0.17 s</td>
<td>1.9 s</td>
</tr>
<tr>
<td>Speedup</td>
<td>1.0 (REF)</td>
<td>7.6</td>
<td>0.7</td>
</tr>
</tbody>
</table>

### Memory bound (dynamics) problem

<table>
<thead>
<tr>
<th>Machine</th>
<th>Interlagos</th>
<th>Fermi (2090)</th>
<th>GPU+transfer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>0.16 s</td>
<td>0.038 s</td>
<td>1.7 s</td>
</tr>
<tr>
<td>Speedup</td>
<td>1.0 (REF)</td>
<td>4.2</td>
<td>0.1</td>
</tr>
</tbody>
</table>

The simple lesson: leave data on the GPU!
Application performance of COSMO dynamical core (DyCore)

- The CPU backend is 2x-2.9x faster than standard COSMO DyCore
  - Note that we use a different storage layout in new code
  - 2.9x applied to smaller problem sizes, i.e. HPC mode (see later slide)
- The GPU backend is 2.8-4x faster than the CPU backend
- Speedup new DyCore & GPU vs. standard DyCore & CPU = 6x-7x
Solving Kohn-Sham equation is the bottleneck of most DFT based materials science codes

Kohn-Sham Eqn.
\[
\left( -\frac{\hbar^2}{2m} \nabla^2 + v_{\text{LDA}}(\vec{r}) \right) \psi_i(\vec{r}) = \epsilon_i \psi_i(\vec{r})
\]

Ansatz
\[
\psi_i(\vec{r}) = \sum_{\mu} c_{i\mu} \phi_{\mu}(\vec{r})
\]

Hermitian matrix
\[
H_{\mu\nu} = \int \phi^*_{\mu}(\vec{r}) \left( -\frac{\hbar^2}{2m} \nabla^2 + v_{\text{LDA}}(\vec{r}) \right) \phi_{\nu}(\vec{r}) d\vec{r}
\]

Basis may not be orthogonal
\[
S_{\mu\nu} = \int \phi^*_{\mu}(\vec{r}) \phi_{\nu}(\vec{r}) d\vec{r}
\]

Solve generalized eigenvalue problem
\[
(H - \epsilon_i S) = 0
\]
Solving Kohn-Sham equation is the bottleneck of most DFT based materials science codes

Kohn-Sham Eqn. \[
\left( -\frac{\hbar^2}{2m} \nabla^2 + v_{LDA}(\vec{r}) \right) \psi_i(\vec{r}) = \epsilon_i \psi_i(\vec{r})
\]

Remarks:
Typical matrix size is up to 10,000
Only a few methods require high accuracy and matrix size up to 100,000
Some methods try to avoid eigensolvers (orthogonalization will be important motif instead)
When lower accuracy is an option, order(N) methods are used instead
    (this is covered by one of our HP2C projects see CP2K on hp2c.ch)

Dominant motif:
Solve generalized eigenvalue problem \[
(H - \epsilon_i S) = 0
\]

We will need between 10% and 50% of the eigenvectors
Solving the generalized eigenvalue problem

\[ Ax = \lambda Bx \]

\[ B = LL^H \]

\[ A' = L^{-1} AL^{-H} \]

\[ A'y = \lambda y \]

\[ T = Q^H A'Q \]

\[ Ty' = \lambda y' \]

\[ y = Qy' \]

Standard 1 stage solver

Most time consuming step, dominated by level 2 BLAS (memory bound)

xPOTRF

xHEGST

xHEEVx

xTRSM

T = Q^H A'Q

xHETRD

xSTExx

xUNMTR
Solving the generalized eigenvalue problem with 2 stage solver

\[ A x = \lambda B x \]
\[ B = LL^H \]
\[ A' = L^{-1} A L^{-H} \]
\[ A'y = \lambda y \]
\[ x = L^{-H} y \]

\[ A'' = Q_1^H A' Q_1 \]
\[ T = Q_2^H A'' Q_2 \]
\[ Ty' = \lambda y' \]
\[ y'' = Q_2 y' \]
\[ y = Q_1 y'' \]

reduction to banded
Most time consuming step, but dominated by BLAS-3
tri-diagonalize

needs two eigenvector transformations (but easy to parallelize)
2 stage solver maps best to hybrid CPU-GPU architecture

\[
A'y = \lambda y \\
\downarrow \\
A'' = Q_1^H A' Q_1 \\
\downarrow \\
T = Q_2^H A'' Q_2 \\
\downarrow \\
Ty' = \lambda y' \\
\downarrow \\
y'' = Q_2 y' \\
\downarrow \\
y = Q_1 y''
\]

- reduction to banded hybrid
- tri-diagonalize CPU
- divide an conquer CPU
- needs two eigenvector transformations (but easy to parallelize)

(a) xGEQRT (dark blue).
(b) xPNRFB (red).
(c) xHER2K (red).
(d) matrix structure.
Profile and performance for relevant problem size

- Results of the standard eigensolvers computing partial eigenspace

double complex, matrix size 10'000
6 threads on CPU + GPU (1x Intel X5650, 1x Nvidia M2090)
Comparison of general eigensolvers: multi-core vs. hybrid

double complex, matrix size 10000
hybrid with MAGMA (*): 6 threads + GPU (1x Intel X5650, 1x Nvidia M2090)
multi-core with MKL: 12 threads (2x Intel X5650)
multi-core with ELPA: 12 processes (2x Intel X5650)

(*) A novel hybrid CPU-GPU generalized eigensolver for electronic structure calculations based on fine grained memory aware tasks, A. Haidar, S. Tomov, J. Dongarra, R. Solcà, and TCS (under review)
(will be included in release of MAGMA late summer/fall 2012)

<table>
<thead>
<tr>
<th>Code</th>
<th>measure</th>
<th>problem size</th>
<th># nodes</th>
<th>IL-IL (Rosa)</th>
<th>IL-Fermi (Tödi)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP2K</td>
<td>time</td>
<td>864 H\textsubscript{2}O</td>
<td>16</td>
<td>50</td>
<td>33</td>
<td>1.51</td>
</tr>
<tr>
<td>COSMO DyCore</td>
<td>time</td>
<td>128x96</td>
<td>1</td>
<td>2.15</td>
<td>1.19</td>
<td>1.81</td>
</tr>
<tr>
<td>Gromacs</td>
<td>ns/day</td>
<td>512k H\textsubscript{2}O</td>
<td>1</td>
<td>0.659</td>
<td>1.025</td>
<td>1.56</td>
</tr>
<tr>
<td>Eigen Solver</td>
<td>time</td>
<td>8100x8100</td>
<td>1</td>
<td>288</td>
<td>94.3</td>
<td>3.05</td>
</tr>
<tr>
<td>SPECFEM3D</td>
<td>time</td>
<td>300k Eur.</td>
<td>16</td>
<td>20.78</td>
<td>21.13</td>
<td>0.98</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>394.37</td>
<td>304.90</td>
<td>1.29</td>
</tr>
</tbody>
</table>

### Remarks
- Snapshot for typical workloads of the codes (this is work in progress!)
- SPECFEM3D 300k Europe is relatively small and reaches scaling limits on 16 GPUs
- Apples-to-apples comparison across several codes is hard and not always meaningful (need to consider codes individually)
- No Intel results shown here, since we don’t have SandyBridge+GPU with PCIe3 yet
- note SandyBridge performs significantly better than AMD/Interlagos
- COSMO DyCore covers only about 60% of production run
Computer performance and application performance increase $\sim 10^3$ every decade

- 1988: 1 Gigaflop/s, Cray YMP, 8 processors
- 1998: 1.02 Teraflop/s, Cray T3E, 1'500 processors
- 2008: 1.35 Petaflop/s, Cray XT5, 150'000 processors
- 2018: ~1 Exaflop/s, 100 million or billion processing cores (!)

- If HPC community continues to be in denial this NewPM will be CUDA

<table>
<thead>
<tr>
<th>Year</th>
<th>Technology</th>
<th>GFlop/s</th>
<th>CPU Cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>1988</td>
<td>Cray YMP</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>1998</td>
<td>Cray T3E</td>
<td>1.02</td>
<td>1'500</td>
</tr>
<tr>
<td>2008</td>
<td>Cray XT5</td>
<td>1.35</td>
<td>150'000</td>
</tr>
<tr>
<td>2018</td>
<td></td>
<td>~1</td>
<td>~1 Exaflop</td>
</tr>
</tbody>
</table>

- First sustained GFlop/s Gordon Bell Prize 1988
- First sustained TFlop/s Gordon Bell Prize 1998
- First sustained PFlop/s Gordon Bell Prize 2008
- Another 1,000x increase in sustained performance

- MPI
- MPP
- MPI + OpenMP
- MPP + Multi Core
- X + NewPM
- X + massive threading

~100 Kilowatts → ~5 Megawatts → 20-30 MW

~1 Exaflop/s

~20-30 MW

~100 million or billion processing cores (!)
Acknowledgements

- Team working on COSMO refactoring
  - Oliver Fuhrer (Meteo Swiss)
  - Tobias Gysi and Daniel Müller (Supercomputing Systems)
  - Xavier Lapillone and Carlos Osuna (C2SM@ETH)
  - Will Sawyer, Mauro Bianco, Ugo Vareto, Ben Cumming, and TCS (CSCS)
  - Tim Schröder, and Peter Messmer (NVIDIA)
  - Ulli Schättler, and Michael Baldauf (DWD)

- Team working on eigensolvers for hybrid architectures (MAGMA)
  - Stan Tomov, Azam Haidar, Jack Dongara (UTK), Raffaele Solca, and TCS (ETH)

- Future Systems team working on architecture evaluations at CSCS
  - Sadaf Alam, Gilles Fourestey, Ben Cumming, Jeff Poznanovic, Ugo Vareto, and Mauro Bianco

- Many researchers from the HP2C platform that develop the applications

- Funding for much of the HPC related work
  - Swiss University Conference through HP2C Platform (www.hp2c.ch)
  - PRACE 2IP WP8 and all institutions that had to provide matching funds
  - ETH Board through ETH Zurich
THANK YOU!